Lea Bogensperger

Model-augmented Deep Learning for Variable Flip Angle T1-mapping

Master's Thesis

to achieve the university degree of

Master of Science

Master's degree programme: Biomedical Engineering

submitted to

Graz University of Technology

Supervisors

Dipl.-Ing. Oliver Maier

Univ.-Prof. Dipl.-Ing. Dr.techn. Rudolf Stollberger

Institute for Medical Engineering
Head: Univ.-Prof. Dipl-Ing. Dr.techn. Rudolf Stollberger

Graz, December 2019

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

Datum	Unterschrift

Acknowledgements

First and foremost I would like to thank my thesis supervisor, Oliver Maier, for all his valuable support and the helpful discussions. He encouraged me to work independently on the topic and always steered me in the right direction whenever necessary.

I would also like to thank Prof. Stollberger, for his competent guidance and expertise in the field and Prof. Scharfetter for encouraging me to pursue my master's thesis at this Institute in the first place. A big thanks goes to the whole Institute of Medical Engineering, I have truly enjoyed my time here with you.

I want to express my deepest gratitude to my parents for always believing in me and for supporting me in my decision to study at Graz University of Technology. Thank you to my brothers, Jonathan and Moritz, to my whole extended family and my dear friends. A special thanks goes to Julia for proof-reading this thesis.

Finally, I would like to thank Martin for being a constant source of energy and joy to me and for providing me with continuous motivation throughout the course of my studies and my master's thesis.

Abstract

The imaging of quantitative MRI parameters enables an objective comparison of the investigated tissues on the basis of physical properties and is therefore considered an invaluable component of precision medicine. Recent work shows that long scan times for qMRI can be reduced to a fraction through data subsampling and model-based reconstructions. However, reconstruction with nonlinear models can be time-consuming and thus makes them an ideal candidate for deep learning methods. There have only been very few approaches of mapping MRI parameters by means of deep learning, and only one, where a model-augmented neural network is used to estimate M_0 and T_2 maps. In this master's thesis, a U-Net is proposed that estimates M_0 and T_1 maps from a corresponding set of subsampled Variable Flip Angle (VFA) images, comprising the physical model consistency term that incorporates the signal model into the objective function. The acceleration potential is shown on numerical brain phantoms and on retrospectively subsampled in vivo measurements via transfer-learning for cartesian subsampling of $R = 1.89$, $R = 3.43$ and $R = 5.84$. It is further shown that prior knowledge of B_1 can be included in the signal model but it is even possible to estimate B_1 inhomogeneity maps in a separate output channel of the neural network. A comparison between learning the parameter maps in image domain or in k-space was performed, showing that training in image domain yields significantly better results without any backfolding artifacts.

Keywords: quantitative MRI, Deep Learning, U-Net, VFA, accelerated MRI

Kurzfassung

Die Bildgebung quantitativer MR Parameter gewährleistet einen objektiven Vergleich der untersuchten Gewebe auf Basis von physikalischer Eigenschaften. Neuere Forschungsarbeiten konnten zeigen, dass lange Scanzeiten für quantitative Bildgebung mittels Unterabtastung der gemessenen Daten und modellbasierter Rekonstruktion auf einen Bruchteil reduziert werden können. Die Rekonstruktion mithilfe nichtlinearer Modelle ist jedoch zeitintensiv und ist daher ein idealer Kandidat für Deep Learning basierte Fitting Methoden. Bisher gibt es nur wenige publizierte Methoden, um MR Parameter mittels Deep Learning zu bestimmen und darunter nur einen Ansatz, welcher das neuronale Netzwerk mit dem Modell der Signalgleichung erweitert, um M_0 und T_2 zu quantifizieren. Im Zuge dieser Masterarbeit wurde ein U-Net implementiert, welches M_0 und T_1 aus einer Serie von unterabgetasteten Gradientenecho Bildern mit variablen Kippwinkeln bestimmt, wobei physikalische Information über die Signalgleichung in der Kostenfunktion enthalten ist. Anhand von numerischen Gehirnphantomen und retrospektiv unterabgetasteten in vivo Messdaten wird das Potenzial der Methode im Bezug auf beschleunigte Messdaten gezeigt. Weiters ist es möglich, bereits bekanntes Wissen über B_1 in das Signalmodell zu integrieren, oder B_1 direkt in einem separaten Ausgangskanal des Netzwerkes zu lernen. Der abschließende Vergleich zwischen Lernprozess im Bildbereich und im k-Raum lieferte deutlich bessere Ergebnisse im Bildbereich ohne verbleibende Artefakte durch Aliasing.

Schlüsselwörter: quantitative MRT, Deep Learning, U-Net, VFA, beschleunigte MRT

Contents

Acronyms

AI Artificial Intelligence
B_0 Main Magnetic Field
B_1 Radiofrequency Field
CNN Convolutional Neural Network
CT Computed Tomography
EPI Echo Planar Imaging
FOV Field of View
HU Hounsfield Units
IR Inversion Recovery
LL Look-Locker
M_0 Pseudo Proton Density
MLP Multilayer Perceptron
MRI Magnetic Resoncance Imaging
nRMSE normalized Root Mean Square Error
PD Proton Density
qMRI Quantitative Magnetic Resoncance Imaging
ReLu Rectified Linear Unit
RF Radiofrequency
SAR Specific Absorption Rate
SNR Signal-to-Noise Ratio
SPGR Spoiled Gradient Recalled Echo
SSIM Structural Similarity Index
T_1 Longitudinal Relaxation Time
T_2 Transversal Relaxation Time
TE Echo Time
TI Inversion Time
TR Repetition Time
TV Total Variation
VFA Variable Flip Angle

1 Introduction

1.1 Preface

Over the last decades, Magnetic Resoncance Imaging (MRI) has become a very important imaging technique in medicine as it enables wide-spread applications in diagnostics and research while being non-invasive. Unlike Computed Tomography (CT), MRI does not require X-rays and offers a significant advantage when imaging soft-tissue structures, but it also has some major drawbacks including longer acquisition times and higher cost. A major benefit of CT is that the resulting images are expressed in Hounsfield Units (HU), where the attenuation coefficients are scaled to contain quantitative information on the X-ray attenuation by the tissue [21].

Quantitative Magnetic Resoncance Imaging (qMRI), an important method that has emerged out of conventional MRI, may provide useful insight into the progression of certain diseases, where degradation might not be visible in a conventionally weighted MRI scan. Especially diagnosis and progression of neuro-degenerative diseases, such as multiple sclerosis and Alzheimer's disease, could be improved with quantitative MRI [11]. This will become even more important as these diseases show an increasing prevalence. As qMRI is based on directly estimating tissue properties, the pseudo proton density M_0 and the longitudinal and the transversal relaxation times, T_1 and T_2, it mainly displays physical tissue parameters and does not account for differences

amongst scanners, allowing thus a more objective comparison and evaluation of tissue. There are various well-established methods that can be used to estimate these parameters, such as the conventional Look-Locker Method [28] based on inversion-recovery [37] or saturation-recovery spin echo sequences [7]. A different method, the variable flip angle (VFA) technique [10], relies on two or more spoiled gradient recalled echo (SPGR) images that are measured with different flip angles. After acquiring the data, an optimization algorithm, typically based on nonlinear least-squares or on a linearization procedure, is crucial to recover the sought parameter maps. Two major disadvantages of qMRI are the longer imaging and reconstruction times needed to recover parameter maps from acquired images.

These are among the reasons that make qMRI an ideal candidate for deep learning approaches. In the recent years, machine learning has started to spark a lot of interest in the field of medical imaging [18, 2, 20, 47], which has mainly been credited to the increase of computational power of graphical processing units. This massive improvement enables researchers to construct neural networks that are used in image segmentation, image reconstruction and denoising applications, to name but a few. Another reason for the hightened popularity of deep learning is due to the availability of big data and the development of user-friendly machine learning libraries such as TensorFlow [30], Keras [6] and PyTorch [33].

With this enormous upsurge of artificial intelligence (AI) in the domain of medical imaging and the above-mentioned obstacles of long reconstruction times in qMRI, the question naturally arises, whether it is possible to combine these two techniques to develop a faster method of predicting paramter maps than the technical and physical limits until now would allow it.

1.2 Aim of this Thesis

The aim of this master's thesis is to investigate whether the parameter maps for a VFA sequence can be predicted by means of a deep learning approach as opposed to conventional fitting routines. A U-Net, which is a special convolutional neural network with an encoder-decoder architecture, is implemented to estimate pixelwise M_0 and T_1 maps. The network is targeted to learn the relation between an input set of variable flip angle (VFA) images and the corresponding parameter maps for anatomical brain slices. The neural network is trained on generic phantom data and is then fine-tuned with measured in vivo data according to the method of transfer-learning. Transfer-learning is a commonly used tool for small training datasets where a larger, similar dataset is taken to pre-train the neural network. Further, a comparison between training the U-Net in image space versus training it in k-space is made to investigate whether this has any major effects on the final results in the subsequent inference phase.

2 Background

2.1 Inverse Problems in Medical Imaging

Inverse problems are prevalent in all fields of sciene and can in its most basic form be formulated as

$$y = A(x) \tag{2.1}$$

The aim of solving an inverse problem is to estimate underlying parameters of a given measurement or observation. It is therefore sought to recover the quantity of interest x from an observation y which are related by a potentially nonlinear operator A. In the domain of MRI, the most basic example of an inverse problem is the recovery of an image given measured data in k-space. However, this also includes further applications such as image denoising and quantitative MRI. For qMRI, this reconstruction process is mathematically expressed as

$$x^* \in \arg\min_x \frac{1}{2}||P\mathcal{F}CS(x) - d||_2^2 \tag{2.2}$$

where d is the measured k-space data, x are the sought parameter maps, C models the coil sensitivities, $\mathcal{F}$ is the fourier transform, P is a sampling pattern and S is a specified signal equation depending on the sequence which was used. The usage of the ℓ_2-norm is justified by the fact that Gaussian noise is assumed for k-space data. For a spoiled gradient echo sequence (SPGR) the signal equation relates the signal to flip angles and parameters and reads as

follows

$$S = M_0 \frac{\sin(\alpha)(1 - e^{-TR/T_1})}{1 - \cos(\alpha)e^{-TR/T_1}} \tag{2.3}$$

If one would like to map T_2, the signal equation would be a different one than for mapping T_1 and could be modelled as a monoexponential decay with Echo Time (TE) such as

$$S(T_2) = M_0 e^{-TE/T_2} \tag{2.4}$$

2.2 qMRI and Relaxation Times

In a very broad sense, the miracle of MRI leads back to the detection of protons H^1 in tissue whereof there is a large abundance in the human body. Applying an external main magnetic field B_0 leads to the proton spins aligning along B_0. Spin is a fundamental property of particles (atomic and subatomic), but will not be dealt with further in this master's thesis. When a radiofrequency (RF) pulse is switched on, the magnetization is flipped according to the flip angle α and the spins dephase. This introduces the fundamental concept of relaxation. The nuclear spins absorb the deposited RF energy and get into an excited state thus changing the orientation of the net magnetization vector M. The spins are then located in the transversal plane. They precess about the longitudinal axis at the Larmor frequency which is governed by the gyromagnetic ratio γ ($42.58 \frac{MHz}{T}$ for water protons) and the main magnetic field strength B_0 according to

$$\omega = \gamma B_0 \tag{2.5}$$

As soon as the RF pulse is turned off, the process of relaxation sets in, which consists of simultaneous longitudinal and transversal relaxation.

Longitudinal or spin-lattice relaxation is guided by the longitudinal relaxation time constant T_1 and accounts for the recovery of the initial value M_0 of the net magnetization. The energy that has been given to the system must leave

it again in the course of T_1 relaxation by being transferred to the surrounding lattice such as neighbouring nuclei, atoms and molecules. This recovery process can be described mathematically by the following equation and is depicted in Figure 2.1.

$$M_z(t) = M_0(1 - e^{-t/T_1}) \tag{2.6}$$

One can see from the above equation and in Figure 2.1 that T_1 is a time constant indicating a recovery of 63.2% of the original net magnetization after one timespan of T_1.

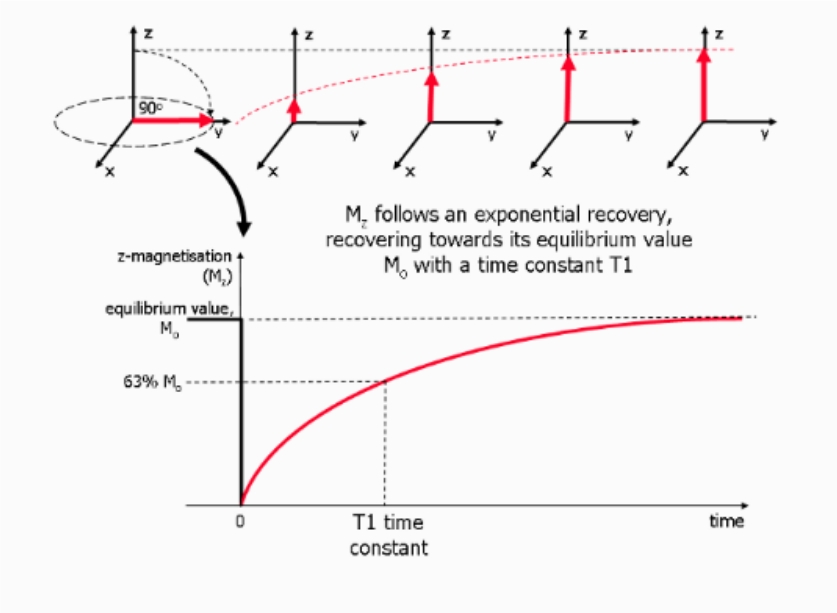


Figure 2.1: Process of T_1 relaxation (Figure taken from Ridgway [35]).

The second relaxation process is transversal or spin-spin relaxation which brings about the decay of the transversal magnetization. Mathematically, it is also based upon a simple exponential decay.

$$M_{xy}(t) = M_0 e^{-t/T_2} \tag{2.7}$$

T_2 relaxation is a very complex process due to a loss in phase coherence of the spins that were tipped into the xy-plane. According to the exponential decay of transverse magnetization, it is solely this interaction between the spins that leads to dephasing.

In practice, since the main magnetic field can never be completely homogeneous, B_0 varies slightly with ΔB_0 depending on the spatial location which directly influences the Larmor frequency of the spins. Due to these small differences in resonant frequencies of neighbouring spins, additional dephasing is brought about which is summarized by the time constant T_2^*. The relation between T_2 and T_2^* is shown in Equation 2.8. In Figure 2.2 the process of dephasing spins is illustrated and the curves of T_2 and T_2^* are shown where it can be clearly seen that T_2^*-decay is faster due to additional dephasing from B_0 inhomogeneities. T_2^*-decay is a reversible process since the inhomogeneities are constant in time and space. Therefore, applying a 180° RF pulse after dephasing leads to a resulting spin echo.

$$\frac{1}{T_2^*} = \frac{1}{T_2} + \frac{1}{T_{2'}} = \frac{1}{T_2} + \gamma \Delta B_0 \tag{2.8}$$

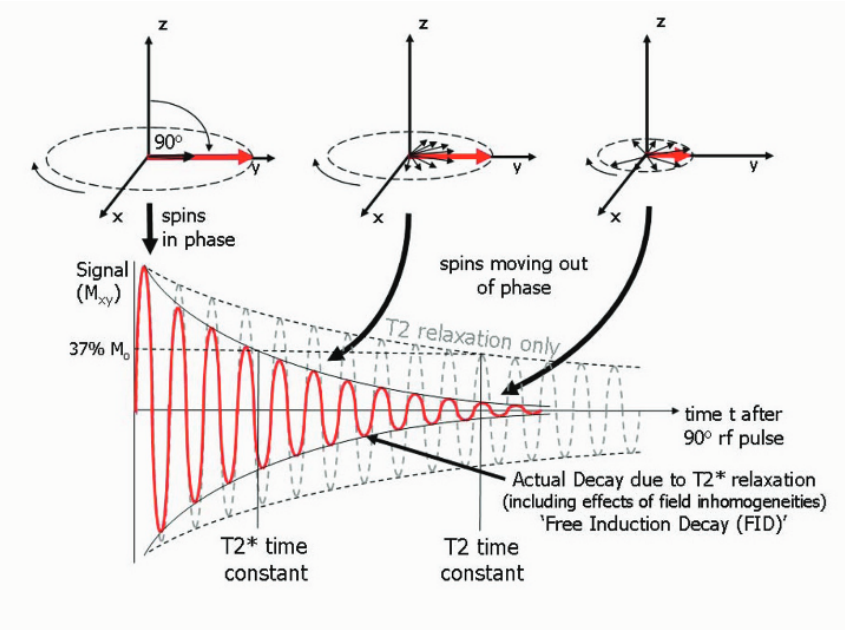


Figure 2.2: Process of T_2 relaxation (Figure taken from Ridgway [35]).

Another parameter that is associated with qMRI is the proton density PD. According to Leroi [24], it is defined by the concentration of protons that undergo resonance and thus are responsible for the signal in MRI. This emphasizes the fact that all images have intensities which are proportional to the proton density. Since this is a proportionality factor, it is sometimes combined with other factors such as coil sensitivities and field inhomogeneities and it is then referred to as pseudo proton density M_0 as it is not solely a physical density anymore.

2.3 Conventional Methods in qMRI

There exist a lot of different T_1 mapping techniques, where three very well-known and established methods are inversion recovery (IR), Look-Locker (LL)

and the variable flip angle (VFA) method. As stated by Taylor et al. [41], the general principle in T_1 mapping requires several images with different parameter settings, such as different T_1 weighting by variable flip angles α, and to then fit these images to a predefined signal equation. Referring to the work by Stikov et al. [40], the inversion recovery and the variable flip angle techniques will be discussed in more detail in the following sections.

2.3.1 Inversion Recovery T_1 Mapping

IR T_1 mapping is the oldest method based on a spin-echo sequence and is considered to be the gold standard in T_1 mapping [37]. The sequence, which is illustrated in Figure 2.3, consists of two RF pulses, where the first is an IR pulse that flips the equilibrium magnetization M_0 by 180°. After waiting for an inversion time (TI) during which the flipped magnetization recovers, a second pulse of 90° is applied to flip the magnetization into the transverse plane. This is followed by the readout phase.

Figure 2.3: IR T_1 mapping (Figure taken from [40]).

The acquired signal S in its most general form results from the following relation shown in Equation 2.9. The parameters a and b relate to proton density and

contain information on the deviation from an ideal 180° pulse. According to Taylor et al. [41], approximating the course of the longitudinal magnetization is approximated surprisingly well using this simple model with one exponential function.

$$S = a + b \exp^{-\frac{TI}{T_1}} \tag{2.9}$$

2.3.2 Variable Flip Angle T_1 Mapping

The variable flip angle (VFA) technique is based on a spoiled gradient-echo sequence and can also be used to acquire 3D T_1 maps in a clinically acceptable time frame (< 30 min according to Cheng and Wright in [5]). The authors highlight its advantages as a low power deposition in comparison to spin echo techniques and low spatial distortion compared to EPI sequences. As described by Stikov et al. [40], at least two gradient-echo measurements with constant TR and TE and varying flip angles α_i are required. The relation between signal intensity, parameters, and flip angles is described by Equation 2.10. The factor k is a proportionality constant incorporating effects such as coil sensitivity and T_2^* relaxation and it is usually combined with the proton density into one parameter that will be fitted, i.e. the pseudo proton density.

$$S = M_0 \sin(\alpha) \frac{1 - e^{-TR/T_1}}{1 - \cos(\alpha) e^{-TR/T_1}} \tag{2.10}$$

Choosing n different flip angles leads to n measurements that allow for a fitting procedure of this equation. The sequence diagramm can be seen in Figure 2.4.

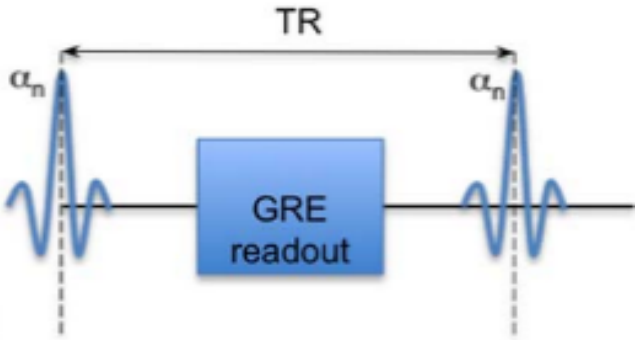


Figure 2.4: VFA sequence for T_1 mapping (Figure taken from Stikov et al. [40]).

Two important issues discussed by Stikov et al. [40] have yet to be mentioned. This is first and foremost the fact that this relation assumes perfect flip angles, which in practice can not be achieved due to B_1 inhomogeneity. Thus, a B_1 field map is required to be able to account for the spatial inhomogeneity of the radiofrequency field. Especially at higher field strengths ($\geq 3T$), this becomes an urgent issue. Furthermore, the above equation also assumes perfect spoiling i.e. the complete disruption of residual transversal phase coherences that might still persist at the end of TR. Since most sequences have a rather short TR to render them clinically feasible, this demands for a good spoiling technique apart from choosing a long TR. This could theoretically be achieved by gradient spoiling, where the amplitude of the gradient is varied randomly and, more important and effective, RF spoiling. This means that the phase of the RF pulse is incremented in a predefined relation.

2.4 Fitting Methods

2.4.1 Iterative Fitting Methods

Based on the formulated inverse problem for qMRI, which consists of a model-based data fidelity term and a regularization term and is shown below in Equation 2.11, an optimization algorithm has to be chosen to find the optimal parameters x.

$$x^* \in \arg\min_x \frac{1}{2}|| \underbrace{P\mathcal{F}CS}_{A}(x) - d||_2^2 + \lambda\mathcal{R}(x) \tag{2.11}$$

These problems are often ill-posed. According to Jacques Hadamard in 1902 [14], a well-posed problem must have the following three properties:

- Existence: a solution exists
- Uniqueness: the solution is unique
- Stability: the solution's behaviour changes continuously with the data

As soon as any of the above three properties is violated, a problem becomes ill-posed. The forward operator A is nonlinear due to the nonlinear signal equation and there are several possible choices for the regularization term, such as the ℓ_2 norm or Total Variation (TV). The undersampling pattern enhances the ill-posedness of the problem and therefore does not allow for a simple analytic solution with the inverse operator A^{-1}.

This calls for iterative algorithms and there are many different algorithms available depending on the specific problem. The most basic first-order method is gradient descent, which can be used if the underlying problem is differentiable. It is based on the fact that for a differentiable function, the gradient at any point will always yield the direction of biggest increase. Therefore, the idea is to take a small step into the negative direction of the gradient

and to iteratively repeat this procedure to find the minimum of a function.

Algorithm 1: Gradient Descent

1 Initialize stepsize t, starting point f^0
2 Define threshold ϵ for stopping criterion
3 **while** *stopping criterion is not yet met* **do**
4 $\quad f^{n+1}(x) = f^n(x) - t\nabla f^n(x)$

Gradient Descent can be used to solve simple and differentiable problems. There are many other algorithms which are based on the principle of gradient descent. Some examples are conjugate gradient descent, which additionally requires the search directions to be orthogonal to each other, and Newton's method, which uses second order derivative information and fits a quadratic function in every iteration, both methods yielding faster convergence than plain gradient descent. There are also more sophisticated methods to select the stepsize t than setting it to a fixed scalar such as diminishing stepsizes and backtracking line search procedures.

2.4.2 Analytic Methods

The analytic method to map T_1 based on three different flip angles from a VFA sequence was first proposed by Blüml et al. [4] in 1993 and it is also described very detailed by Deoni et al. [10] and Cheng and Wright [5]. A correction scheme for T_1 is included in [5] to compensate for inaccurate knowledge of the flip angles due to B_1 field inhomogeneities. The authors start by transforming the equation of a spoiled gradient echo sequence into a linear form of $Y_i = mX_i + b$

with slope m and bias b.

$$\begin{aligned} S_i &= M_0 \sin(\alpha_i) \frac{1 - E_1}{1 - \cos(\alpha_i) E_1} \\ S_i - S_i \cos(\alpha_i) E_1 &= M_0 \sin(\alpha_i) - M_0 \sin(\alpha_i) E_1 \\ \frac{S_i}{\sin(\alpha_i)} - \underbrace{\frac{S_i \cos(\alpha_i) E_1}{\sin(\alpha_i)}}_{\frac{S_i E_1}{\tan(\alpha_i)}} &= \underbrace{M_0 - M_0 E_1}_{M_0(1-E_1)} \\ \frac{S_i}{\sin(\alpha_i)} &= \frac{S_i E_1}{\tan(\alpha_i)} + M_0(1 - E_1) \end{aligned} \tag{2.12}$$

The result of rearranging the SPGR equation has a linear form, where $Y_i = \frac{S_i}{\sin(\alpha_i)}$, $X_i = \frac{S_i}{\tan(\alpha_i)}$, $m = E_1$, and $b = M_0(1 - E_1)$. Therefore, a linear equation is obtained, where a set of 3 input-output pairs $\{X_i, Y_i\}_{i=1}^3$ that were measured with the three flip angles α_i can be used for fitting. A linear regression on this equation yields values of bias b and slope m. M_0 can directly be calculated as

$$M_0 = \frac{b}{1 - m} \tag{2.13}$$

T_1 can be obtained by rewriting the expression for the slope m as

$$T_1 = -\frac{TR}{\ln(m)} \tag{2.14}$$

Figure 2.5 shows the errors that can occur in the linear regression on a low and high angle (α_L and α_H) due to random noise. This leads to shifted data points, whereas the measured magnitude signal will be larger, and thus the true slope gets over- or underestimated as indicated by the dashed and dotted line, depending on whether the noise affects the low or the high angle. This in turn introduces errors in T_1. If high SNR points are taken into consideration and more than two angles are measured, the random noise will have less influence. Solving for T_1 analytically as shown above is an accurate method but might bring about problems related to SNR if smaller flip angles are used. It should be noted that when imaging over a large range of T_1 values, multiple angles α_i deliver better results in a more uniform precision [5].

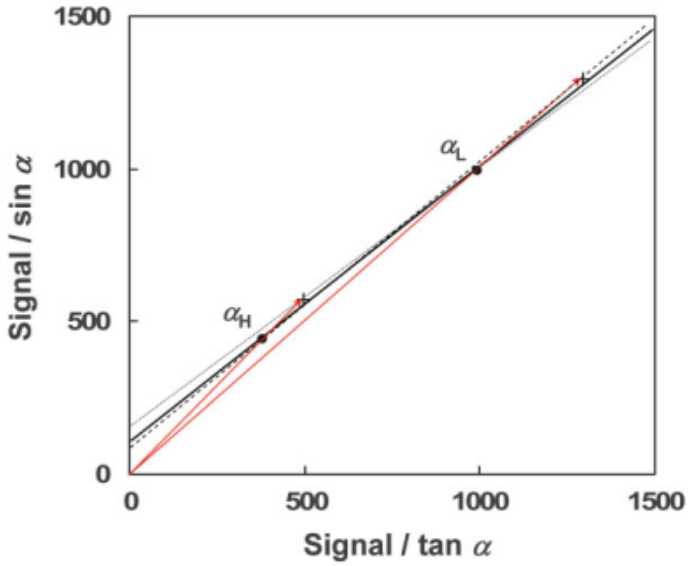


Figure 2.5: T_1 estimation from linear regression on two different angles α_L, α_H (Figure taken from Cheng and Wright [5]).

2.5 Neural Networks

2.5.1 Idea

The basic idea of neural networks was inspired by information flow within the human brain. Electrical signals are transmitted from dendrites via chemical synapses to neurons and are carried on along the axon. This mechanism is the principle for a neural network, which receives multiple inputs, combines them and passes them through some activation function. This process is depicted in Figure 2.6.

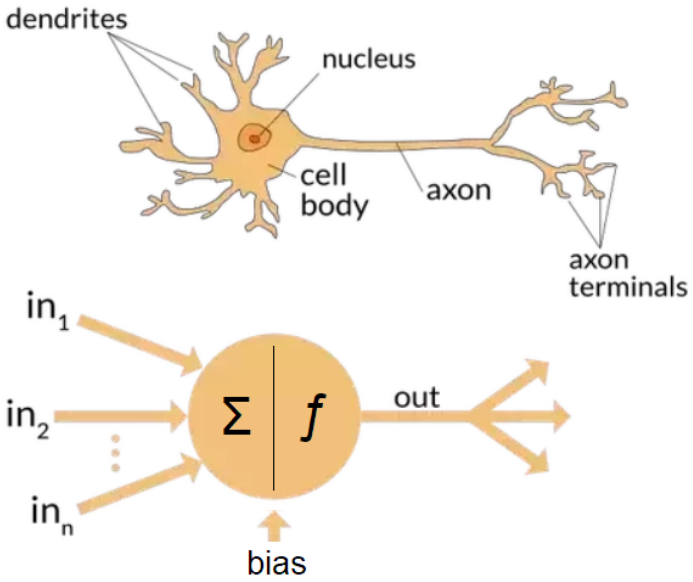


Figure 2.6: A biological neuron and a most basic neural network. (Figure taken from [45]).

According to Csáji [8], neural networks can be trained to learn an input-output mapping, where the weights of a neural network are adaptively changed depending on the surrounding environment. By using activation functions for the neurons, a nonlinearity is introduced into the model and allows to model more complex, physical processes.

The universal approximation theorem, as shown by Csáji [8], states that a multilayer feedforward network with at least one hidden layer containing finite number of hidden units can approximate any continuous function on a compact (i.e. closed and bounded) subset of $\mathbb{R}^n$. This makes them universal approximators and also implies the variety of functions that can be approximated. In a mathematically rigorous way this was formulated by Csáji [8] as follows

Theorem 1 *Let $\phi(.)$ be an arbitrary activation function. Let $X \subseteq \mathbb{R}^m$ and X is compact. The space of continuous functions on X is denoted by $C(X)$. Then*

$\forall f \in C(X), \forall \epsilon > 0 : \exists n \in \mathbb{N}, a_{ij}, b_i, w_i \in \mathbb{R}, i \in \{1, ..., n\}, j \in \{1, ..., m\}$:

$$(A_n f)(x_1, ..., x_m) = \sum_{i=1}^{n} w_i \phi(\sum_{j=1}^{m} a_{ij} x_j + b_i)$$

as an approximation of the function $f(.)$*; that is*

$$||f - A_n f|| < \epsilon$$

2.5.2 Network Architectures

Multilayer Perceptron

A Multilayer Perceptron (MLP) defines any feedforward neural network with an input layer, at least one hidden layer and an output layer, each layer followed by a potentially nonlinear activation function. In Figure 2.7 a very basic example of an MLP with one hidden layer can be seen.

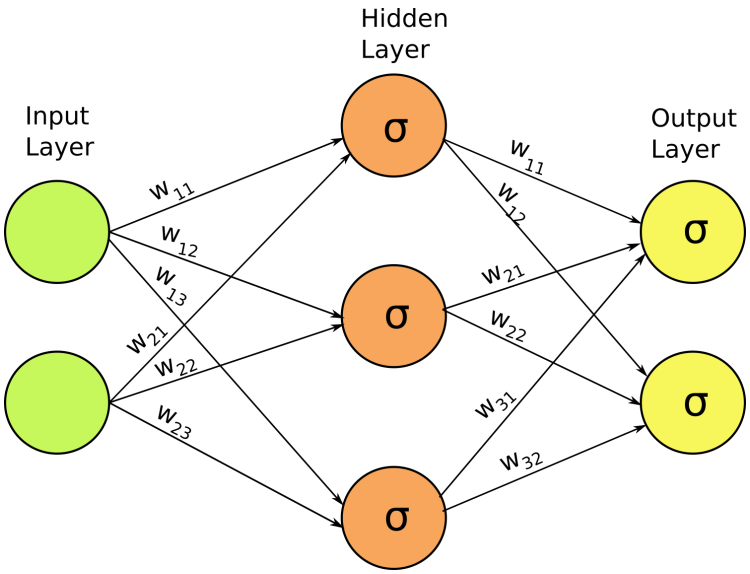


Figure 2.7: Simple MLP with one hidden layer.

For simplicity, the input layer will further be denoted as x, the output layer as y, z will be the weighted output from the previous layer, and a the activated output. The input layer contains two units, it is therefore composed of $x = (x_1^{(1)}, x_2^{(1)})$ and due to the fact that there is no activation function it holds that $x = a^{(1)} = z^{(1)}$. The hidden layer has three hidden units with an activation function. In this case it is the sigmoid function which is defined as

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.15}$$

The weights are labelled in such a way that w_{ij} represents the weight from the previous layer i to the successive layer j. The output $z^{(2)}$ can be written as the following matrix-vector product.

$$\begin{pmatrix} z_1^{(2)} \\ z_2^{(2)} \\ z_3^{(2)} \end{pmatrix} = \begin{pmatrix} w_{11}^{(1)} & w_{21}^{(1)} \\ w_{12}^{(1)} & w_{22}^{(1)} \\ w_{13}^{(1)} & w_{23}^{(1)} \end{pmatrix} \begin{pmatrix} a_1^{(1)} \\ a_2^{(1)} \end{pmatrix} + \begin{pmatrix} b_1^{(1)} \\ b_2^{(1)} \\ b_3^{(1)} \end{pmatrix} \tag{2.16}$$

The result in the input layer is then element-wise passed on to the activation function, in this case the sigmoid function, which maps all input values into the range of $[0, 1]$.

$$a^{(2)} = \sigma(z^{(2)}) \tag{2.17}$$

Finally, the weighted result from the units in the hidden layer get summed up in the output layer, which can again be written as a matrix-vector product.

$$\begin{pmatrix} z_1^{(3)} \\ z_2^{(3)} \end{pmatrix} = \begin{pmatrix} w_{11}^{(2)} & w_{21}^{(2)} & w_{31}^{(2)} \\ w_{12}^{(2)} & w_{22}^{(2)} & w_{32}^{(2)} \end{pmatrix} \begin{pmatrix} a_1^{(2)} \\ a_2^{(2)} \\ a_3^{(2)} \end{pmatrix} + \begin{pmatrix} b_1^{(2)} \\ b_2^{(2)} \end{pmatrix} \tag{2.18}$$

Since the activation function in this layer is again chosen as the element-wise sigmoid function (this does not have to be the same activation function as in the previous layer), this leads to

$$a^{(3)} = \sigma(z^{(3)}) \tag{2.19}$$

The final output y is then

$$y = a^{(3)} = \sigma(W^{(2)} \underbrace{\sigma(\overbrace{W^{(1)}a^{(1)} + b^{(1)}}^{z^{(2)}})}_{a^{(2)}} + b^{(2)}) \tag{2.20}$$

From this it can be seen that nonlinear activation functions are crucial, otherwise the layers in a neural network could always be combined to one single layer leading to only one input and one output layer without a hidden layer. The universal approximation theorem requires at least one hidden layer to approximate any continuous function.

Convolutional Neural Network

Convolutional Neural Networks (CNNs) are neural networks that are based on the mathematical operation of convolutions. According to Goodfellow et al. [13], for an image, which is a discretized two-dimensional array $I(i, j)$, a two-dimensional kernel K is typically used to perform weighted average operations over a range of $\pm m, n$ at each location, which can be formulated as

$$S(i, j) = (K * I)(i, j) = \sum_m \sum_j I(i - m, j - n)K(m, n) \tag{2.21}$$

CNNs first came up in a paper in 1999 by LeCun et al. [23], the proposed architecture is shown in Figure 2.8. The basic idea is to extract features from images rather than injecting all the pixels of an image into a feedforward neural network at once, which would require a huge amount of weights for several hundred input pixels.

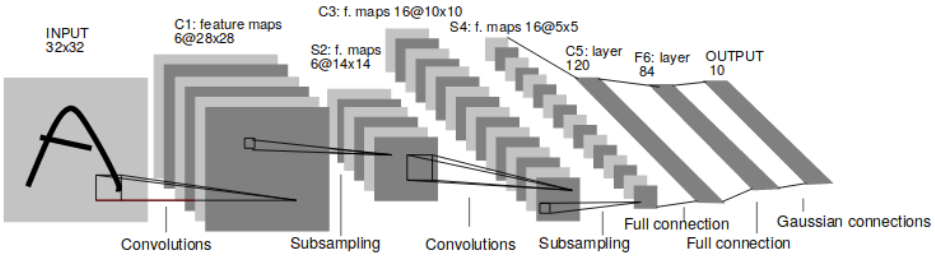


Figure 2.8: Architecture of first CNN as proposed by LeCun et al. [23].

CNNs brought about the ideas of a local receptive field where each layer receives inputs from a small neighbourhood in the previous layer. This enables the detection of basic shapes, edges, and corners and higher-order information can be extracted as the information gets fed along the different layers. Another huge advantage is the fact that much less parameters for the filter kernels are required to be learned due to shared weights as opposed to fully connected neural networks, where each unit in the previous layer is connected to each unit in the successive layer. According to Goodfellow et al. [13], a typical layer of a CNN consists of three stages: convolutions, an activation function, and a pooling function. The task of the activation function is usually to introduce a nonlinearity to the linear activations produced by the convolutions. The final pooling operation then helps to include geometric invariance to local translations by employing subsampling such as taking the maximum number or the average number in a small local neighbourhood.

Extensive theoretical background on how and why CNNs work that well and to what extend they can be used as approximators for functions has not not been present earlier. Recently, it was shown by Zhou [46] that they are indeed universal and can thus approximate any continuous function to an arbitrary accuracy provided the neural network is deep and large enough.

2.5.3 Backpropagation Algorithm

The backpropagation algorithm will be shown for the simple multilayer perceptron above. Each neural network has a loss function, wherefore there exist a lot of different possibilities. In this case, the loss function is composed by the sum of squared differences, i.e.

$$\mathcal{L}(y, y^*) = \frac{1}{2}||y - y^*||^2 \tag{2.22}$$

In order to train a neural network with gradient-based optimization schemes, the parameter updates are calculated in each iteration in order to adapt the network's parameters. The aim is therefore to find the derivatives of the loss function with respect to the network parameters $\theta = \{W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}\}$ by means of the chain rule.

$$\frac{\partial \mathcal{L}}{\partial W^{(2)}} = \underbrace{\underbrace{\frac{\partial \mathcal{L}}{\partial a^{(3)}}}_{y-y^*} \underbrace{\frac{\partial a^{(3)}}{\partial z^{(3)}}}_{\sigma'(z^{(3)})}}_{\delta^{(3)}} \underbrace{\frac{\partial z^{(3)}}{\partial W^{(2)}}}_{a^{(2)^T}} \tag{2.23}$$

The first part is named $\delta^{(3)}$ since it keeps reappearing in all of the other derivations as well. The derivative of the sigmoid activation function is $\sigma'(x) = \frac{e^{-x}}{(1+e^{-x})^2}$ and will simply be written as σ' in the derivation.

$$\delta^{(3)} = diag(\sigma'(z^{(3)}))(y - y^*) \tag{2.24}$$

This leads to the final result for the first derivation.

$$\frac{\partial \mathcal{L}}{\partial W^{(2)}} = \delta^{(3)} a^{(2)^T} \tag{2.25}$$

The second derivation is quite similar to the previous one and reads as follows.

$$\frac{\partial \mathcal{L}}{\partial b^{(2)}} = \underbrace{\frac{\partial \mathcal{L}}{\partial a^{(3)}} \frac{\partial a^{(3)}}{\partial z^{(3)}}}_{\delta(3)} \underbrace{\frac{\partial z^{(3)}}{\partial b^{(2)}}}_{1} = \delta^{(3)} \tag{2.26}$$

Next, computing the derivative with respect to $W^{(1)}$ yields

$$\frac{\partial \mathcal{L}}{\partial W^{(1)}} = \underbrace{\underbrace{\frac{\partial \mathcal{L}}{\partial a^{(3)}} \frac{\partial a^{(3)}}{\partial z^{(3)}}}_{\delta^{(3)}} \frac{\partial z^{(3)}}{\partial a^{(2)}} \frac{\partial a^{(2)}}{\partial z^{(2)}}}_{\delta^{(2)}} \underbrace{\frac{\partial z^{(2)}}{\partial W^{(1)}}}_{a^{(1)^T}} \tag{2.27}$$

$\delta^{(2)}$ is therefore defined as follows.

$$\delta^{(2)} = diag(\sigma'(z^{(2)}))W^{(2)^T}\delta^{(3)} \tag{2.28}$$

The complete derivation is then given by

$$\frac{\partial \mathcal{L}}{\partial W^{(1)}} = \delta^{(2)} a^{(1)^T} \tag{2.29}$$

Due to the fact that $\frac{\partial z^{(2)}}{\partial b^{(1)}} = 1$ the derivation for the last network parameter is calculated as follows.

$$\frac{\partial \mathcal{L}}{\partial b^{(1)}} = \delta^2 \tag{2.30}$$

After the gradients have been derived they can be used in a gradient-based optimization procedure, which aims to iteratively find the minimum of our objective loss function. In the case of gradient descent the update rule for the different network parameters θ_i with a stepsize η is generally denoted as follows.

$$\theta_i^{t+1} = \theta_i^t - \eta \frac{\partial \mathcal{L}}{\partial \theta_i} \tag{2.31}$$

2.6 Previous Approaches of Deep Learning in Quantitative MRI

There have not been many approaches to incorporate deep learning methods into qMRI. However, the work of Liu et al. [26] will be briefly summarized since it is related very closely to the work done in this master's thesis. Furthermore,

a short overview of the work of Sabidussi et al. [34] will be given, which was presented at the European Society for Magnetic Resonance in Medicine and Biology in 2019. Both of the following approaches entirely work with convolutions in image domain.

MANTIS: Model-Augmented Neural neTwork with Incoherent k-space Sampling for efficient MR T_2 mapping

Liu et al. [26] proposed a model-augmented neural network to map T_2. The proposed framework is depicted in Figure 2.9. They implemented a U-Net, which is a special type of CNN, to predict the proton density PD - referred to as I_0 - and T_2 from a multi-echo spin echo sequence with retrospectively undersampled input images. The relation between the signal equation and the parameters at the j^{th} echo time is as follows

$$S_j(I_0, T_2) = I_0 \cdot e^{-TE_j/T_2} \tag{2.32}$$

A signal model fidelity term was added to the loss function to obtain a cyclic loss that ensures that predicted parameter maps actually produce the acquired input data. This additional term serves as a regularizer in optimizing the neural network parameters as shown below. The matrix E consists of the Fourier transform and a successive undersampling pattern, while C denotes the output of the U-Net and d_j is the k-space measurement for the j^{th} echo time. The constants λ_{data} and λ_{cnn} are empirically chosen as 0.1 and 1.0.

$$\theta^* = \arg\min_{\theta} \left(\lambda_{data} \Big[\sum_{j=1}^{t} ||ES_j(C(i_u|\theta)) - d_j||_2^2 \Big] + \lambda_{cnn} \Big[||C(i_u|\theta) - (I_0, T_2)||_2 \Big] \right) \tag{2.33}$$

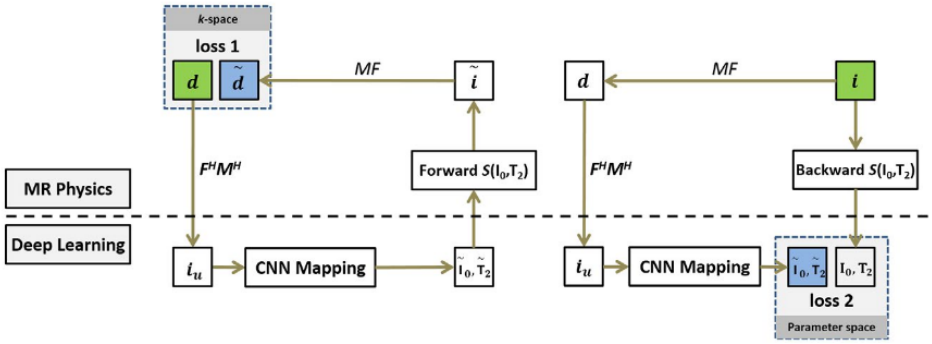


Figure 2.9: MANTIS framework with two loss components by Liu et al. [26].

The authors used a randomized undersamling pattern scheme to render the network more robust with respect to reconstructing images acquired with different undersampling patterns. They compared their approach with conventional iterative reconstruction approaches and were able to demonstrate improved reconstruction performance for acceleration factors $R = 5$ and $R = 8$.

A deep learning approach to T_1 mapping in quantitative MRI

In [34], Sabidussi et al. presented their work at the European Society for Magentic Resonance in Medicine and Biology in 2019, where they use a ResNet (as proposed by He et al. [16]) to predict T_1 maps from Inversion Recovery (IR) input data. A residual building block with its basic functionality is illustrated in Figure 2.10. As explained in [16], it represents the following relation

$$y = \mathcal{F}(x, \{W_i\}) + x \tag{2.34}$$

where y and x are the output and input of the layer and the function $\mathcal{F}$ is the residual mapping that is learned by the network. This seems to offer a huge advantage for deep convolutional networks with an increasing number of

layers, where it is easier to learn a residual mapping than unreferenced functions. Especially with deep networks, initialization of weights becomes crucial and increasingly more difficult if no previous knowledge can be incorporated. Problems with vanishing or exploding gradients might be amplified through the increasing number of layers. Thus, using residual blocks might provide remedy in these situations.

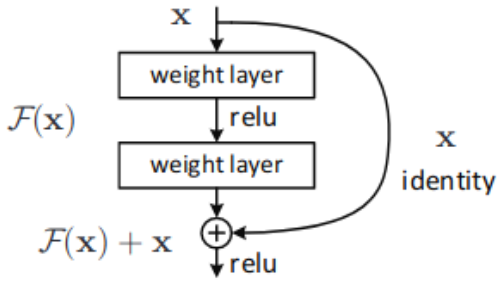


Figure 2.10: Residual building block [16].

Figure 2.11 depicts the implemented ResNet architecture by Sabidussi et al. [34]. As an input 10 T_1-weighted images from an IR experiment were used, whereas groundtruth and parameter maps were created by assigning T_1 and M_0 values to brain slices. The relation between signal intensity and parameter maps is given by

$$f_n(A, B, T_1) = A + Be^{-TI_n/T_1} \tag{2.35}$$

with $n = 1, ..., N$, TI_n inversion times, whereas A and B relate to proton density and inversion efficiency [34]. The ResNet was then able to predict T_1, A, and B maps from the series of fully-sampled IR input images.

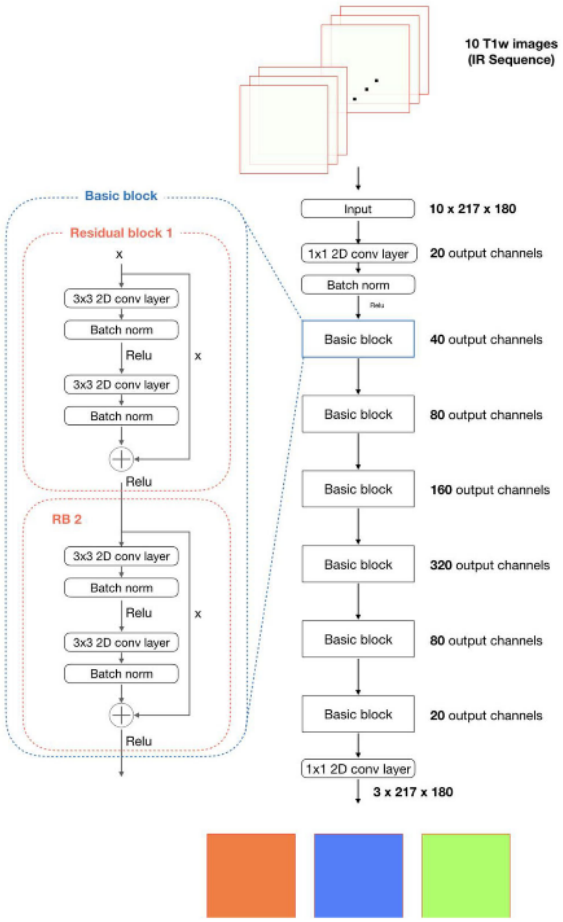


Figure 2.11: Neural network architecture for T_1 mapping by Sabidussi et al. [34].

2.7 Motivation

First of all, the enormous increase of deep learning in so many different areas of science and in particular in medical imaging suggests that it bears a huge potential to be used in future qMRI. Whereas this hype can at times be tempting to simply use machine learning whereever and whenever possible, it

certainly has its valid point in qMRI, where the time factor plays a crucial role and is still among the largest obstacles. Moreover, the theoretical work on multilayer perceptrons (Section 2.5.1) but also more recently on convolutional neural networks (Section 2.5.2) prepares the ground for combining CNNs and qMRI since any continuous function can be approximated by a large and deep enough CNN. This guarantees the theoretical possibility of approximating M_0 and T_1 maps from a VFA sequence, however, to an arbitrary accuracy. Further inspired by the novel developments of mapping MR parameters with neural networks (Section 2.6) the next step seems to be obvious to expand it to mapping parameter maps from a VFA sequence, where the relation between signal intensity and parameters is more complex than the underlying exponential function in the previous works. The implemented neural network and training procedure will therefore be described in the following section.

3 Methods

3.1 U-Net

3.1.1 Idea & Training Procedure

The original U-Net was proposed by Ronneberger et al. [36] to tackle a segmentation problem for biomedical images. The aim was to develop a neural network architecture that allows for localization in the output as it is often required in image processing rather than single class labels in classification tasks.

The proposed architecture is shown in Figure 3.1. The U-Net has the typical downsamling path consisting of convolutions, activation functions and pooling layers but without a fully connected layer at the last stage. Instead, the bottleneck marks the start of a successive upsampling path by alternately using transposed convolutions to enlarge the feature maps and convolutions followed by activation functions. This procedure is accompanied by skip connections, which ensure better localization by concatenating layers from the downsampling path with the respective layers from the upsampling path. The exemplary network in Figure 3.1 outputs 2 segmentation maps.

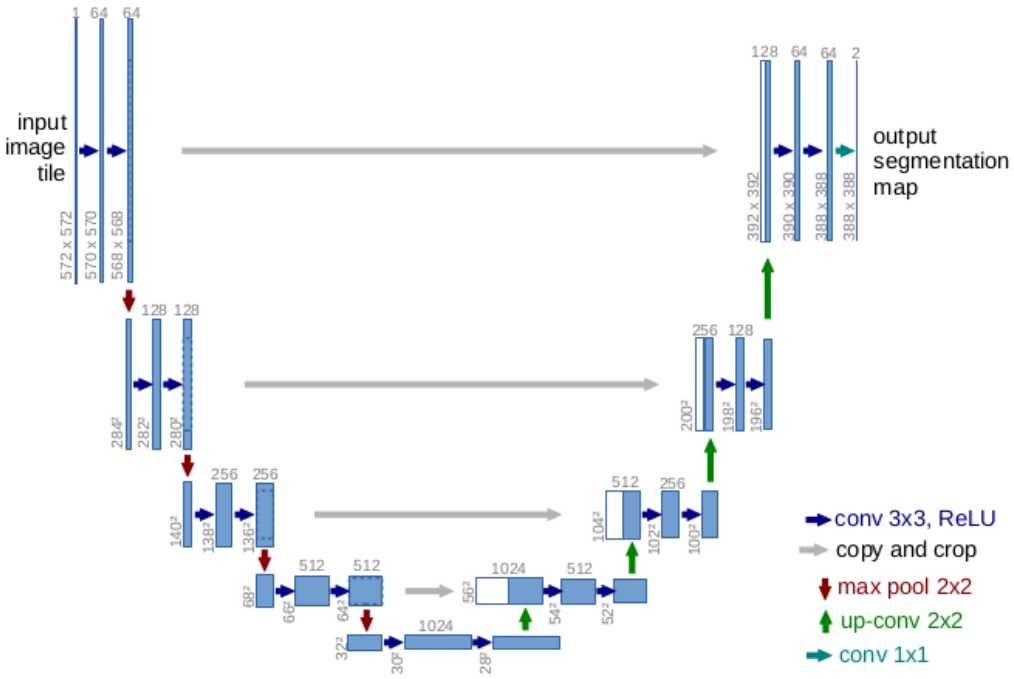


Figure 3.1: Original U-Net architecture [36].

The main advantage of U-Nets is routed in the fact that location information from the contractive path is combined with structural and textural information from the expansive path. Since they are trained in an end-to-end manner, this enables a pixel-by-pixel output and a per-pixel localization. Furthermore, learning only convolution kernels makes the architecture computationally more efficient since these kernels are independent of the in-plane size of the input image and no dense layers have to be used. Moreover, max pooling aims to increase the receptive field in each downsampling layer, however, at the cost of losing spatial information to some extend. This is the reason why skip connections are useful to restore this lost information in the subsequent upsampling layers.

The fact that the advantages of the U-Net architecture were widely recognized and that they have successfully been used in many problems so far have encouraged the decision to also use it for learning T_1 and M_0 maps in this work. This is supported by the universal approximation theorem, which was discussed in section 2.5.1, according to which it must theoretically be possible to estimate parameter maps from a set of VFA input images.

Figure 3.2 illustrates the training process for the U-Net during which the filter kernels θ are updated iteratively. The network receives input images which are then propagated through the U-Net to produce a prediction, which is compared with a set of reference parameters by means of a loss function $\mathcal{L}$. The parameters θ are then updated by calculating the gradients $\frac{\partial \mathcal{L}}{\partial \theta}$ by means of backpropagation for a gradient-based optimization scheme.

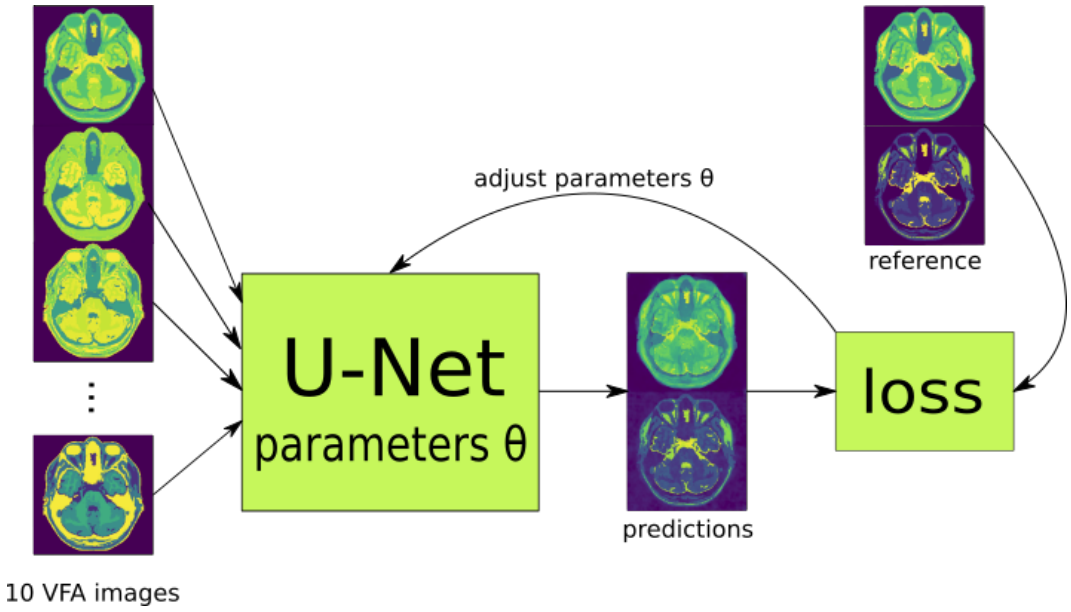


Figure 3.2: Training Procedure.

Hyperparameters are chosen prior to the learning process and include parame-

ters such as number of epochs, batch size, initial stepsize α, split ratio between training and test set, and decay rates β_1, β_2. The decay rates were set to the default values of the optimizer whereas hyperparameters such as initial stepsize and batch size were selected subsequently to a grid search procedure. After one epoch the network has seen the whole training dataset once, whereas its parameters get updated each time after encountering a training data batch.

3.1.2 Image Domain

In Figure 3.3 the implemented architecture derived from the original U-Net can be seen. Unlike the U-Net by Ronnerberger et al. [36] this network was not used to create segmentation maps but to estimate parameter maps of M_0 and T_1 by means of pixelwise regression. The input is a series of 10 variable flip angle (VFA) images with constant flip angles $\alpha_i = \{1°, 3°, 5°, 7°, 9°, 11°, 13°, 15°, 17°, 19°\}$, which were chosen to cover a wide range of possible T_1 values.

The feature maps were limited to a number of 20 in the first step as opposed to the original U-Net and the proposed network has one stage less in depth. These measures were simply taken to prevent overfitting as much as possible by reducing the number of learnable parameters by the U-Net. This ensures a balance between learnable and training parameters.

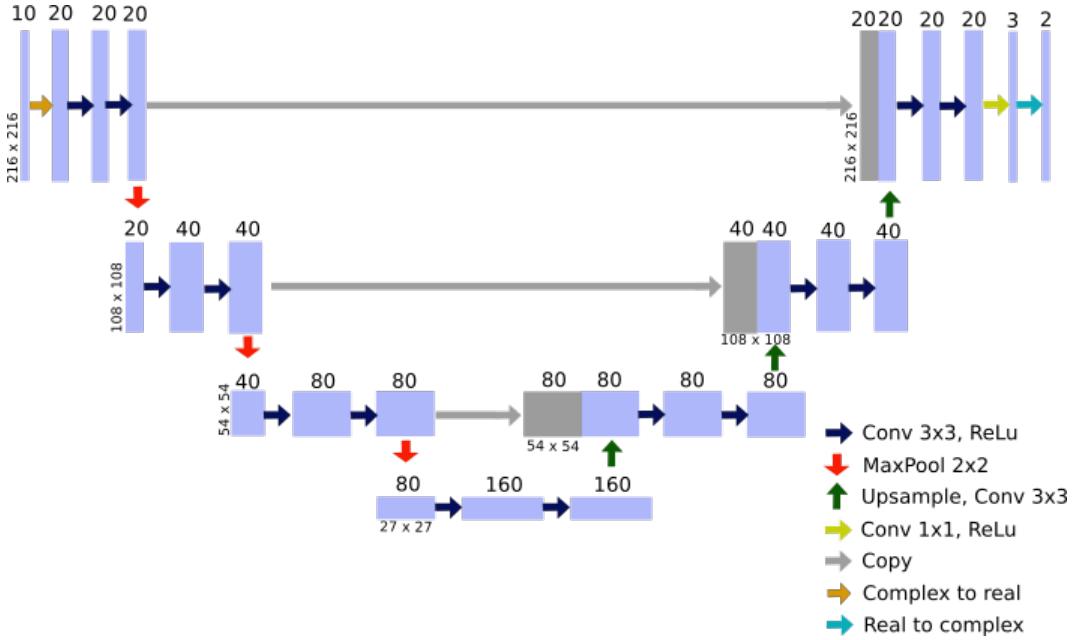


Figure 3.3: Implemented U-Net architecture.

This network architecture has a total of 838 860 learnable parameters. How much information is available for training will therefore depend on the number of training samples. Taking as an example a training set of 895 images of size 216×216, one obtains 41 757 120 training parameters, although it has to be taken into account that a lot of pixels (about $30 - 40\%$) are background pixels and thus do not contain information. In the following pages, the essential components of the U-Net are described in more detail.

Weight Initialization

A very basic strategy is to initialize the convolutional filter kernels of the network by randomly drawing from a Gaussian distribution with a fixed standard

deviation. However, if the initialization is not good - be it random or chosen in a specific way such as initializing all weights with zeros - the minimization of the network's loss function might not converge. The signal either explodes or vanishes and is rarely useful in the successive layers. The gradients in backpropagation will consequently also be extremely large or small, which significantly impedes convergence.

There are other, more sophisticated weight initialization schemes such as Xavier weight initialization [12] and He weight initialization as proposed by He et al. [17]. While Xavier weight initialization was developed for linear activation functions, convolutional neural networks often use the Rectified Linear Unit (ReLu) function as an activation function, which will be described below. This is where He weight initialization [17] comes into play. The method is based on the variances within each layer, where the authors derive the following relation.

$$\mathrm{Var}[y_L] = \mathrm{Var}[y_1]\left(\prod_{l=2}^{L}\frac{1}{2}n_l\mathrm{Var}[w_l]\right) \tag{3.1}$$

Here, n_l represents the number input units in the weight tensor of the current layer l. The factor $\frac{1}{2}$ stems from the ReLu activation function which sets half of the signal to 0 at the end of each layer, which is the only difference to Xavier initialization. Since the input signals should not change in magnitude when being passed through the network, the variance for the layer L should be the same as in layer 1. This is achieved by setting the whole product of the variances from all the layers to a certain scalar, e.g. $\frac{1}{2}n_l\mathrm{Var}[w_l] = 1, \forall l$. Therefore, all weights should have a Gaussian distribution with $\mu = 0$ and $\sigma = \sqrt{2/n_l}$.

Convolutions

The convolutions used in all layers in this work have a 3×3 kernel in-plane, and it is exactly the entries of these filters which will be learned in the training process.

In Figure 3.4, an example of a convolution process between two successive layers is shown. In this case, there are 3 channels in the input layer and one channel in the output layer. Therefore a filter kernel that is of dimension 3 in its depth is needed, leading to a $3 \times 3 \times 3$ filter in case of setting the spatial kernel size to 3×3. This results in a total of 27 learnable parameters.

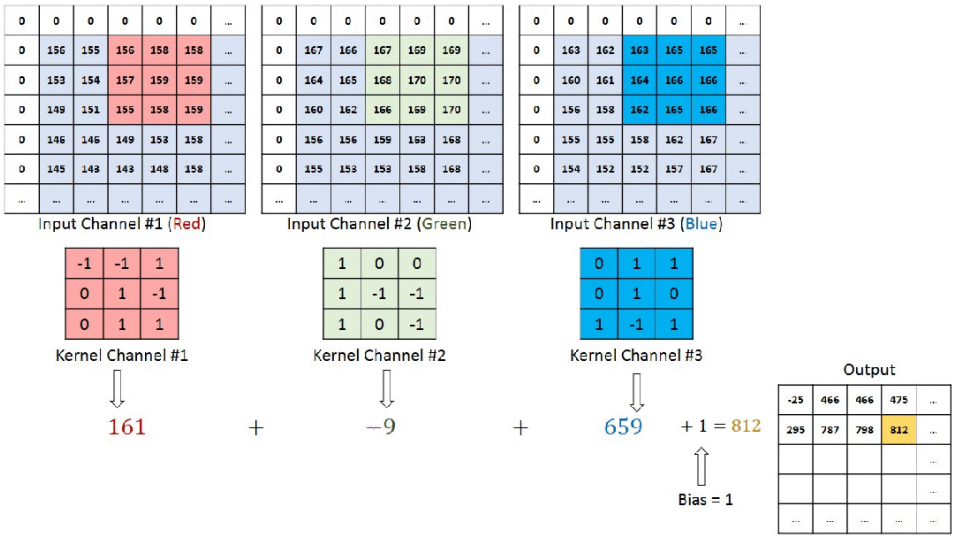


Figure 3.4: Example of a convolution taken from Saha [1].

In the first layer of the proposed U-Net in Figure 3.3, convolution kernels of size $3 \times 3 \times 20$ are needed, since the input layer has 20 channels which correspond to the real and imaginary channels of the 10 VFA input images. To obtain a number of 20 feature maps in the next layer, a total of 20 convolutions filters, each of size $3 \times 3 \times 20$, is required.

Activation Function

Figure 3.5 shows the activation function used in the U-Net, which is a ReLu function. According to Nwankpa et al. [31] it is the most commonly used

activation function and ensures faster computation since no exponentials and divisions are required for computation.

$$ReLu(x) = \max(0, x) = \begin{cases} x & \text{for } x > 0 \\ 0 & \text{else} \end{cases} \tag{3.2}$$

Its main advantage is claimed to be the fact that it eliminates the vanishing gradient problem, which can be observed more frequently with other activation functions. This becomes obvious by considering the derivative. For nonnegative inputs the gradient will always be 1, whereas for other activation functions such as the sigmoid function, the gradient might take on values much smaller than 1 which are successively multiplied with each other and therefore lead to vanishing gradients.

$$\frac{\partial ReLu(x)}{\partial x} = \begin{cases} 1 & \text{for } x > 0 \\ 0 & \text{else} \end{cases} \tag{3.3}$$

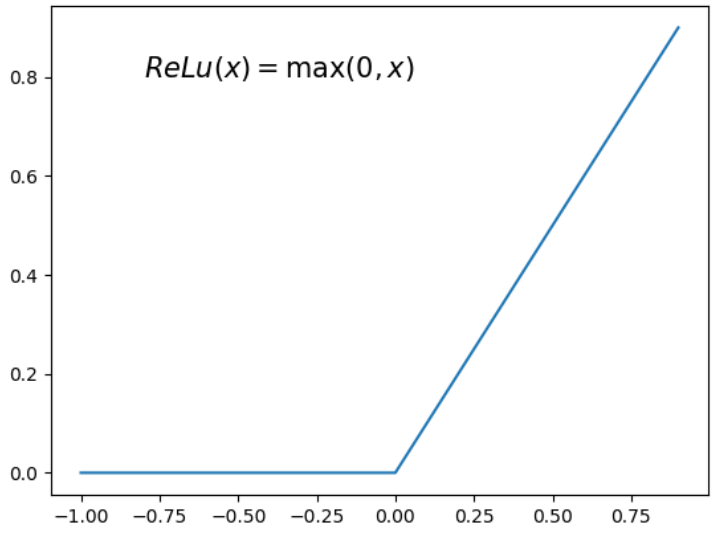


Figure 3.5: ReLu activation function.

The problem of vanishing gradients still persists for cases where a neuron's output is negative, which will be set to 0 with a ReLu activation function. These neurons will then be inactive and do not pass on information in the backpropagation process. A strategy to combat this issue would be to use activation functions such as the Leaky ReLu ($f(x) = \max(\alpha x, x)$) where α is a very small negative slope. This gives the affected neurons the chance to reactivate. However, from a different perspective, one could also argue that it is this feature of ReLus that ensures their biological plausibility by adding nonlinear behaviour to the activation function.

Batch Normalization

Batch normalization is an essential component in each convolutional layer. The concept was first introduced by Sergey Ioeffe and Christian Szegedy in 2015 [19]. They recognized that during training, the input distribution of each layer is altered since the weights of previous layers have changed. The authors refer to this as *internal covariate shift* and therefore suggest to normalize the inputs to each layer. This allows for higher learning rates and makes the network less sensitive to unsuitable initialization. They propose to normalize a layer x with its expectation and variance over the current batch by

$$\hat{x}^{(k)} = \frac{x^{(k)} - E[x^{(k)}]}{\sqrt{Var[x^{(k)}]}} \tag{3.4}$$

Simply normalizing each input layer might change what the respective layer can represent, therefore, it is essential to add a transformation with the learned parameters $\gamma^{(k)}$ and $\beta^{(k)}$, the new standard deviation and the new mean, to scale and shift the normalized value as follows.

$$y^{(k)} = \gamma^{(k)}\hat{x}^{(k)} + \beta^{(k)} \tag{3.5}$$

This results in the following algorithm for a mini-batch $\mathcal{B} = \{x_{1...m}\}$ by Ioeffe and Szegedy [19].

Algorithm 2: Batch Normalization

Input: mini-batch $\mathcal{B} = \{x_{1...m}\}$,
parameters to be learned: γ, β
Output: $\{y_i = BN_{\gamma,\beta}(x_i)\}$
$\mu \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$ `// mini-batch mean`
$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2$ `// mini-batch variance`
$\hat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$ `// normalize`
$y_i \leftarrow \gamma\hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i)$ `// scale and shift`

Max Pooling

As described by Goodfellow [13], max pooling takes the maximum value within a specified rectangular neighbourhood. It is used to render detected features invariant to small translations of the input, this indicates that even though the input is changed by a small amount, the output values would remain more or less the same. This concept is illustrated in Figure 3.6 taken from Goodfellow [13], where max pooling with a specified width of 3 and a stride of 2 is used. Using a stride greater than 1 leads to a reduced representation size which is how the max pooling layer is used to downsample feature maps.

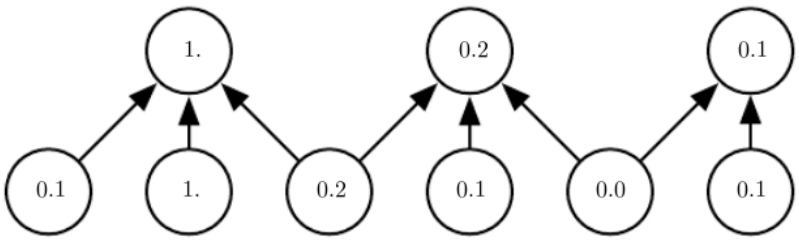


Figure 3.6: Max pooling with downsampling taken from Goodfellow [13].

Upsampling

A very common method to upsample the feature maps in the expanding path is using transposed convolutions, which enlarge the spatial dimensions. An example to illustrate this process with a 3×3 convolution kernel and a stride of 2 is given in Figure 3.7 by Pröve [3].

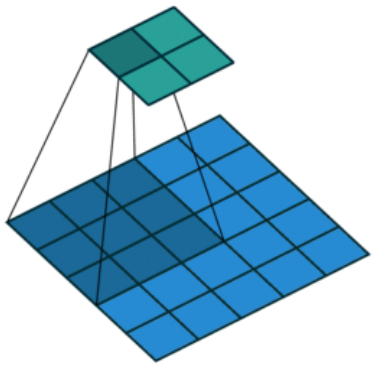


Figure 3.7: Example of a transposed convolution taken from Pröve [3].

As described by Odena et al. [32], transposed convolutions might lead to checkerboard artifacts. This happens if the transposed convolution has an uneven overlap, because the kernel size is not divisible by the stride, which is the spacing between successive filter movements along the input image. This overlap might persist in both dimensions of the image and creates a characteristic checkerboard artifact in the resulting feature map. Neural networks, according to Odena et al. [32], are often not capable of learning to compensate for the consequently arising overlapping patterns, especially if multiple channels are present. Exemplary feature maps when using deconvolution as a means of upsampling in the implemented U-Net are shown in Figure 3.8. These are feature maps from the upsampling path in the first layer above the bottleneck with spatial feature map dimensions 54×54.

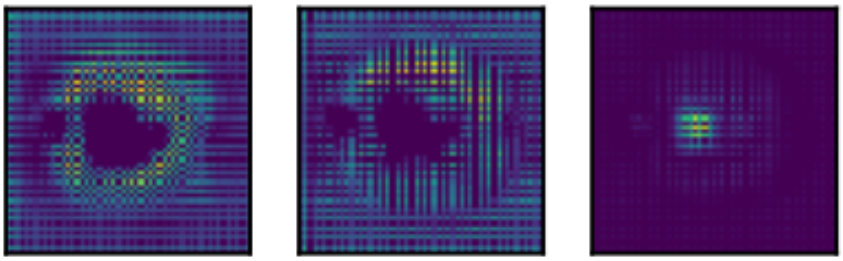


Figure 3.8: Checkerboard artifacts with deconvolution in an exemplary deeper layer with spatial feature map dimensions 54×54.

Therefore, the implemented U-Net uses nearest neighbour interpolation to upsample the feature maps in the upsampling path. The interpolution results in a resized image and is then followed by a standard convolution with a 3×3 kernel as a solid alternative approach, as proposed by Odena et al. [32].

Loss Function

The cost function $\mathcal{L}(f(x|\theta), y)$ is a function that measures the deviation of the predicted output by the neural network $f(x|\theta)$ given parameters θ from the target output y. In this work it was chosen to be the mean squared error.

$$\mathcal{L}(\theta) = \sum_{i=1}^{n} ||f(x_i|\theta) - y_i||_2^2 \tag{3.6}$$

To update the parameters θ throughout the learning process, the gradient of the loss function with respect to all the parameters is used for backpropagation.

Regularization

Regularization is a commonly employed technique to prevent neural networks from overfitting and to introduce any given prior knowledge into the model. In

this master's thesis, the regularization for the loss term in the U-Net consists of a data consistency term to ensure physical validity following the work of Liu et al. [26]. Due to the fact that the relation between signal intensity and parameter maps is known as stated in Equation 2.10 in Section 2.3.2, this can be included in the model. The loss function is shown below, where $\lambda_{UNet} = 1.0$ and $\lambda_{data} = 0.1$ were taken from Liu et al. [26].

$$\mathcal{L}(\theta) = \lambda_{UNet}||UNet(S_{\alpha,u}|\theta) - (M_0, T_1)||_2^2 + \lambda_{data} \sum_{\alpha} ||\mathcal{F}^{-1} P \mathcal{F} S_\alpha (UNet(S_{\alpha,u}|\theta)) - S_{\alpha,u}||_2^2 \tag{3.7}$$

$UNet(S_{\alpha,u}|\theta)$ denotes the ouput of the U-Net given undersampled VFA input data and network parameters θ, $\mathcal{F}^{-1}$ and $\mathcal{F}$ are the (inverse) fourier transform operators, and P is the undersampling pattern. In every iteration the network makes sure that the parameter maps predicted by the currently learned filter configuration yield the images that the network received as an input. This is how the physical model is incorporated and it prevents the neural network from predicting parameter maps that do not make sense physically but would still lead to a low loss value if only the simple ℓ_2 loss term from Equation 3.6 was used.

Batch Gradient Descent

In his book on Deep Learning [13], Ian Goodfellow describes the different possibilities of updating the weights in machine learning optimization.

- **batch/deterministic gradient methods:** all training samples are processed at once in one large batch
- **stochastic gradient methods:** a single sample is used at a time
- **minibatch gradient methods:** more than one and less than all training samples are taken to adapt the weights, these methods are commonly referred to as stochastic methods

Taking only one sample at a time could lead to oscillating convergence behaviour and only yields an approximation for the parameters, whereas processing all training samples at one time is rarely possible due to hardware limitations. Therefore, for practical purposes, parameter updates are usually based on batch gradient methods, where a smaller number of samples is randomly taken from the training data each time.

Adaptive Moment Estimation

Adaptive Moment Estimation (Adam) is a method for stochastic optimization proposed by Kingma and Ba [22]. Being a first-order gradient method, Adam uses the first- and second-order moment estimates of the gradients to obtain individual adaptive learning rates. The algorithm described in their paper reads as follows.

Algorithm 3: Adam

Require: stepsize α, exponential decay rates for moment estimates $\beta_1, \beta_2 \in [0, 1)$, stochastic objective function with parameters θ

Initialize: parameters θ_0, 1^{st} moment vector $m_0 \leftarrow 0$, 2^{nd} moment vector $v_0 \leftarrow 0$, timestep $t \leftarrow 0$

while θ_t *not converged* **do**

- $t \leftarrow t + 1$
- $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ `// gradients w.r.t. stochastic objective at time` t
- $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ `// update biased first moment estimate`
- $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ `// update biased second raw moment estimate`
- $\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^t}$ `// bias-corrected first order moment estimate`
- $\hat{v} \leftarrow \frac{v_t}{1-\beta_2^t}$ `// bias-corrected second raw moment estimate`
- $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}$ `// update parameters`
- **return** θ_t

This algorithm minimizes the objective function $f(\theta)$ with respect to its parameters θ in stochastic minibatches. First it updates the moving averages m_t of the gradient and v_t of the squared gradient with the help of the hyperparameters $\beta_1, \beta_2 \in [0, 1)$. The moving averages are estimates of the 1^{st} moment (mean) and the 2^{nd} raw moment (uncentered variance) of the gradient, which then also get bias-corrected to prevent them from being biased towards 0 due to their initialization. In each iteration, these corrected moment estimates are

then used to update the parameters with stepsize α. The parameter ϵ is held very small ($\epsilon = 1e - 8$) and prevents a possible division by 0.

3.1.3 k-space Domain

Figure 3.9 illustrates the U-Net architecture that was used for learning the parameters M_0 and T_1 in k-space, wherefore the architecture was only slightly modified. For the input channels, the 10 complex k-space channels from the VFA image series were split into real and imaginary channels and were then concatenated, this accounts for the 20 input channels that can be seen in the diagram. The U-Net then learns the convolution filter parameters for the feature maps in k-space. In the final layer, the 4 feature maps for the parameters $\{\mathcal{R}(M_0), \mathcal{R}(T_1), \mathcal{I}(M_0), \mathcal{I}(T_1)\}$ are combined to produce 2 complex-valued parameter maps M_0 and T_1 in k-space, onto which the inverse fourier transform is applied to obtain the sought parameter maps in image space. The loss function operates on the predicted and the target parameter maps in image domain. For this architecture there are 840 680 learnable parameters in the filter kernels.

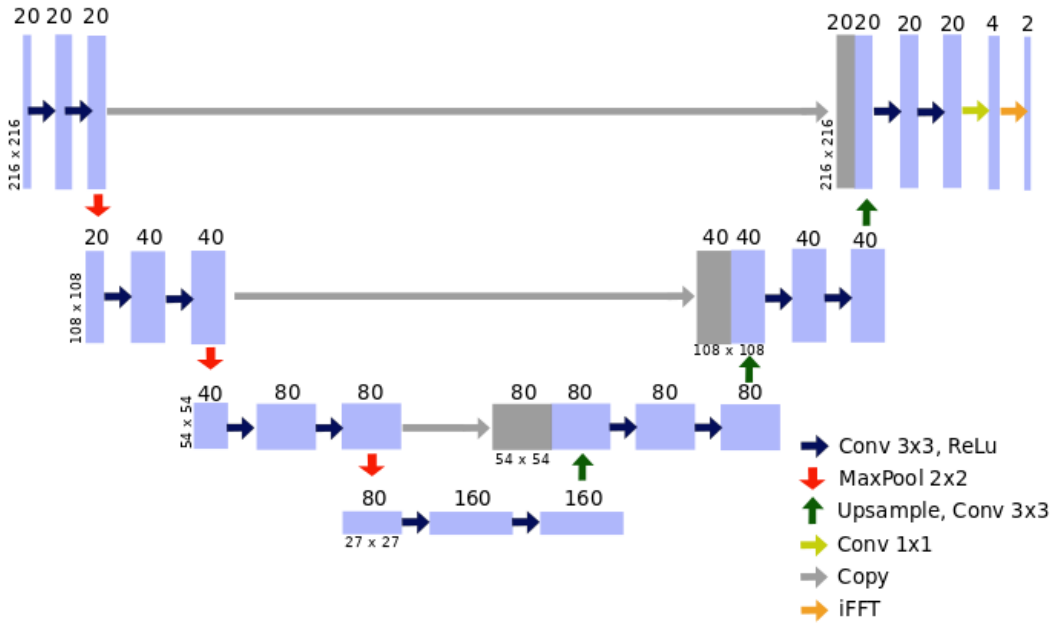


Figure 3.9: Implemented U-Net architecture in k-space.

3.2 Transfer-learning

Transfer-learning [9] is the process of using a pre-trained neural network to retrain either the whole network or a set of specified, usually later layers with a new dataset. It is a common tool if the training dataset is rather small, which is why a larger, different dataset is used to pre-train the network which then serves as a good initialization for the final network. The underlying idea is that the network learns to distinguish basic features and shapes of an image in the first layers whereas the specific details are learned in later layers, therefore the first layers could potentially be reused for the new dataset.

Dar and Çukur [9] propose a transfer-learning approach for training a neural

network with accelerated MRI data. The basic training was performed with a larger dataset in the source domain, which was followed by domain transfer including fine-tuning of the learned weights. They demonstrate the feasibility of transfer-learning between T_1- and T_2-weighted images and between natural and MR images.

There are different approaches to transfer-learning. One can either retrain the last few layers of the neural network, replace the final output layer by a new one or take arbitrarily many layers for fine-tuning up to the point where the whole network is retrained by the target dataset and the pre-trained network is thus used as an initialization. In this work, the output layer of the U-Net was replaced by a new layer with randomly initialized weights due to the fact that the brain phantom and in vivo data had some differences in their numerical target values ranges. The complete network was therefore retrained, but the pre-trained network was used to initialize the weights.

3.3 B_1 Inhomogeneity Maps

Since the B_1 field is spatially not homogeneous, it needs to be compensated for. This is especially important in qMRI, where an imperfect B_1 field leads to spatial variations in flip angles which then has a direct impact on the signal equation. The reference parameter maps were corrected for B_1 inhomogeneities prior to training, thus the U-Net implicitely aims to learn a correction scheme for the parameter predictions.

Being able to learn the B_1 inhomogeneity maps in a separate channel would be highly beneficial for future in vivo applications since acquiring B_1 inhomogeneity maps for flip angle correction usually requires a whole new measurement. Therefore, an additional output channel was specified for the existing U-Net

which aims to estimate the B_1 inhomogeneity map for each slice. Reference B_1 inhomogeneity maps were generated with Bloch-Siegert B_1 mapping [25] to use as groundtruth maps for training on in vivo data. For training on the brain phantom, reference B_1 inhomogeneity maps were numerically produced by means of a smooth $2D$ curved function with values ranging in $[0.8, 1.15]$, to model the deviation from the nominal B_1 field.

3.4 Systematic Errors

To simulate more realistic scenarios, a few systematic error cases were tested for with the trained U-Net. Therefore, the U-Net was trained in image domain on the numerical brain phantom data without any acceleration. Different error scenarios were then created and sent through the trained U-Net in an inference stage. Systematic errors included scaling M_0 by a constant factor, B_1 inhomogeneity resulting in imperfect flip angles, motion during the measurement, a fake tumor located in the white matter, the three undersampling cases and added noise levels of 2%, 5% and 10%.

3.5 Data Acquisition

3.5.1 Generic Phantom

A numerical brain phantom [27] with 120 slices each of size 216×216 was created, where some random examples of phantom parameter maps can be seen in Figure 3.10.

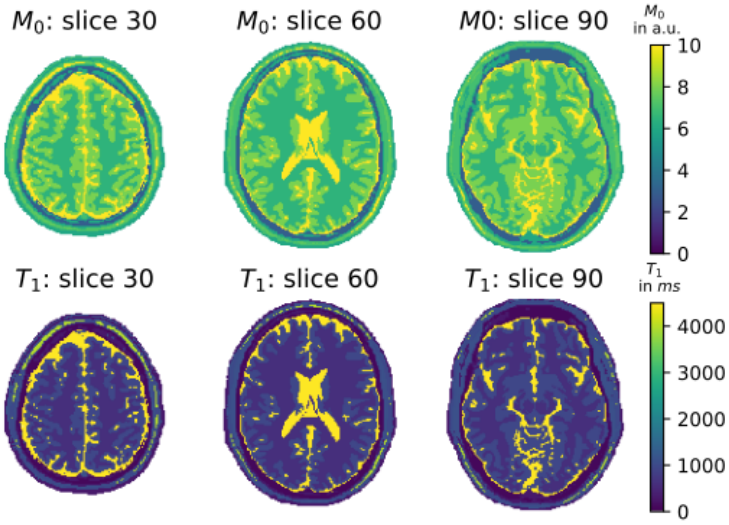


Figure 3.10: Exemplary parameter slices of phantom.

Further, a set of 25 slices from 35 measured brain volumes was available. Therefore, T_1 and M_0 maps per slice for each patient were generated by assigning each pixel the tissue with the highest probability according to underlying segmentation masks for white matter, grey matter and cerebrospinal fluid. Some random examples are shown in Figure 3.11.

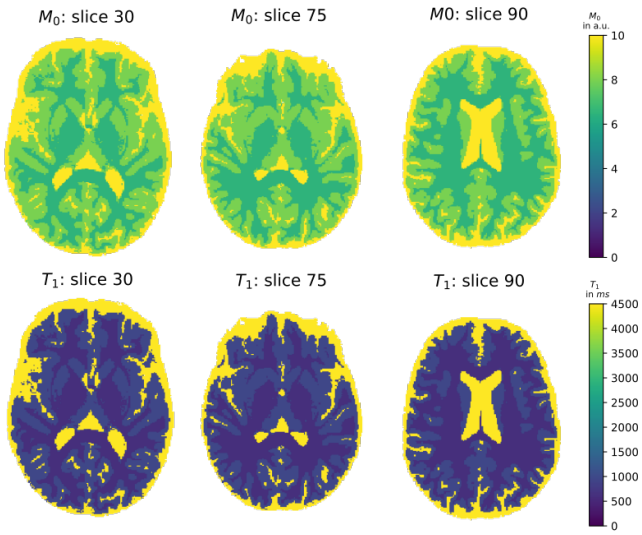


Figure 3.11: Exemplary parameter slices.

3.5.2 In Vivo Data

The single-coil in vivo data was acquired for four healthy volunteers with scan parameters shown in Table 3.1 by using a VFA sequence with the same flip angles of $\{1°, 3°, 5°, 7°, 9°, 11°, 13°, 15°, 17°, 19°\}$ as for the phantom data. This resulted in a total of 420 slices, which were split accordingly into 90% training and 10% test data. Dealing with in vivo data immediately implies that due to B_1^+ inhomogeneities the flip angles will not be perfect as implied by the spatial dependency in Equation 3.8 below and are therefore compensated for by flip angle correction within the signal model.

$$S_i(x) = M_0 \sin(\alpha_i(x)) \frac{1 - e^{-TR/T_1(x)}}{1 - \cos(\alpha_i(x)) e^{-TR/T_1(x)}} \tag{3.8}$$

The reference parameter maps were generated using iterative model-based reconstruction without spatial regularization [29].

Table 3.1: Sequence parameters of the 3D-SPGR sequence for four volunteers.

	TR in ms	TE in ms	FOV in $mm \times mm \times mm$	Scan Matrix in $a.u.$	Bandwidth in Hz
Volunteer 1	4.86	1.97	$224 \times 224 \times 64$	$224 \times 224 \times 64$	108 640
Volunteer 2	4.8	1.95	$224 \times 224 \times 64$	$224 \times 224 \times 64$	110 880
Volunteer 3	4.8	1.95	$224 \times 224 \times 64$	$224 \times 224 \times 64$	110 880
Volunteer 4	4.8	1.95	$224 \times 224 \times 64$	$224 \times 224 \times 64$	110 880

3.6 Data Preprocessing

3.6.1 Normalization

In general, there exist different approaches to attain consistent input data, such as either normalizing the data to a certain interval or standardizing to a mean μ and standard deviation σ. Standardization, also referred to as z-score normalization, centers the training data about $\mu = 0$. This is usually sufficient and additional normalization to a certain range is not necessary if ReLu activation functions are used, since their derivative is always 1 for a positive input. Normalization to an interval is more crucial if an activation function such as the sigmoid function is used, since its derivative for larger input values is close to 0, at the cost of removing information by scaling all input to the same interval.

Therefore, for both training in image space and in k-space, the input data was standardized to $\mu = 0$ and $\sigma = 1$. To achieve this target, the global mean and standard deviation of the whole dataset was used. Standardization in k-space was exerted for real and imaginary channels together to preserve differences

amongst the respective channels. The target parameter maps M_0 and T_1 were both normalized to be in the range $[0, 1]$ to compensate for the different value ranges they each live in.

Gaussian noise of 2% from the channel with the highest signal intensity was added to all the numerical phantom data in order to simulate noise similar to in vivo noise levels. This approximation is valid given the fact that noise in the raw k-space data is assumed to be Gaussian distributed.

3.6.2 Undersampling Method

A regular 1D undersampling pattern was chosen to retrospectively subsample the data in k-space. Therefore, five percent of the central k-space lines were kept to enhance the available low frequency information and additionally every 2^{nd}, 4^{th} or 8^{th} line of the remaining k-space was taken whereas the other lines were set to 0. This yields effective acceleration rates of $R = 1.89, R = 3.43$ and $R = 5.84$. Likewise, one can say that 52.78%, 29.167% and 17.13% of k-space are filled for each of the respective subsampling scenerios. These subsampling masks were equally applied to the VFA images within each slice. The three different sampling patterns can be seen in Figure 3.12. The sampling patterns were then shifted to match the location of the k-space center before the subsequent multiplication with k-space data.

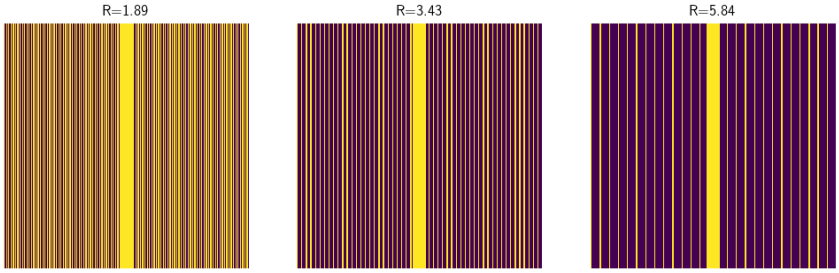


Figure 3.12: Sampling masks for acceleration factors $R = 1.89, R = 3.43$ and $R = 5.84$.

Applying these sampling masks to the images, i.e. multiplying them with the k-space data, produces spatial aliasing or wrap-around artifacts in the results in image domain. This is due to the fact that taking every n^{th} k-space line is a violation of the Nyquist criterion. Skipping k-space lines leads to a worse resolution in k-space Δk and this in turn decreases field of view (FOV) in image domain due to the relation:

$$\Delta k = \frac{1}{FOV} \tag{3.9}$$

What follows are the typical backfolded images, an example of which can be seen in Figure 3.13.

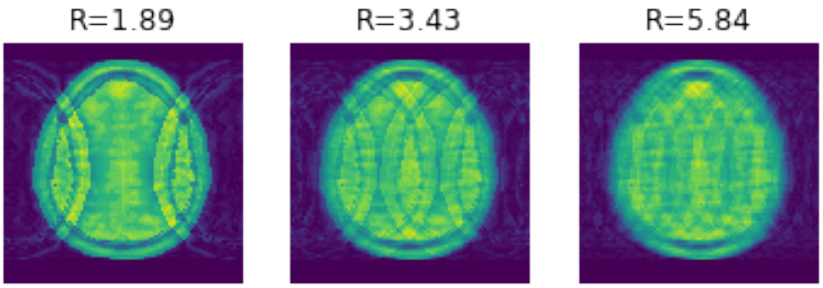


Figure 3.13: Aliased images for acceleration factors $R = 1.89, R = 3.43$ and $R = 5.84$.

3.7 Quantitative Evaluation

In addition to a qualitative evaluation of the images, quantitative measures on all anatomical structures were taken to enable a more objective comparison. Among them is the Structural Similarity Index (SSIM) proposed by Wang et al. in 2004 [44]. For two images x and y, i.e. a predicted image and its reference image, it is defined as follows

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{3.10}$$

$C1$ and C_2 are constants stabilizing the division, which are defined as $C_1 = (K_1L)^2$ and $C_2 = (K_2L)^2$, where L is the dynamic range of the pixel values and $K_1 \ll 1$ and $K_2 \ll 1$ are small constants. μ_x and σ_x^2 are the mean and the variance of an image x, whereas σ_{xy} denotes the covariance of x and y, i.e. how much these two images vary together. As explained by Wang et al. [44], the SSIM is based on the assumption that human visual perception usually extracts structural information from an image and it thus incorporates structure, luminance and contrast terms in its error measure. The SSIM was calculated by using the module `measure` from the open-source image processing library `scikit-image` [42]. Therefore, the constants K_1 and K_2 were taken from the original paper [44] and thus set rather arbitrarily to $K_1 = 0.01$ and $K_2 = 0.03$, which yielded the best performance according to Wang et al. [44].

The second quantitative measure that was used to evaluate images is the normalized root-mean-square error (nRMSE) for a predicted image $\hat{y}$ and its reference image y.

$$nRMSE = \frac{1}{y_{max} - y_{min}} \sqrt{\frac{\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}{N}} \tag{3.11}$$

The nRMSE is the normalized version of the root-mean-square error, which in its most basic form consists of the squared, pixelwise difference between prediction and reference image.

4 Results

4.1 Numerical Simulation

4.1.1 Image Domain

Table 4.1 contains the hyperparameters for training the U-Net in image domain.

Table 4.1: Experimental setup in image domain.

Hyperparameters	
#training samples	895
#test samples	100
Epochs	1800
Batch size	8
Stepsize α	0.001
Decay rate β_1	0.9
Decay rate β_2	0.999

Figure 4.1 shows reference and predicted parameter maps for undersampled VFA images of the brain phantom. The quantitative results are given in Table 4.2.

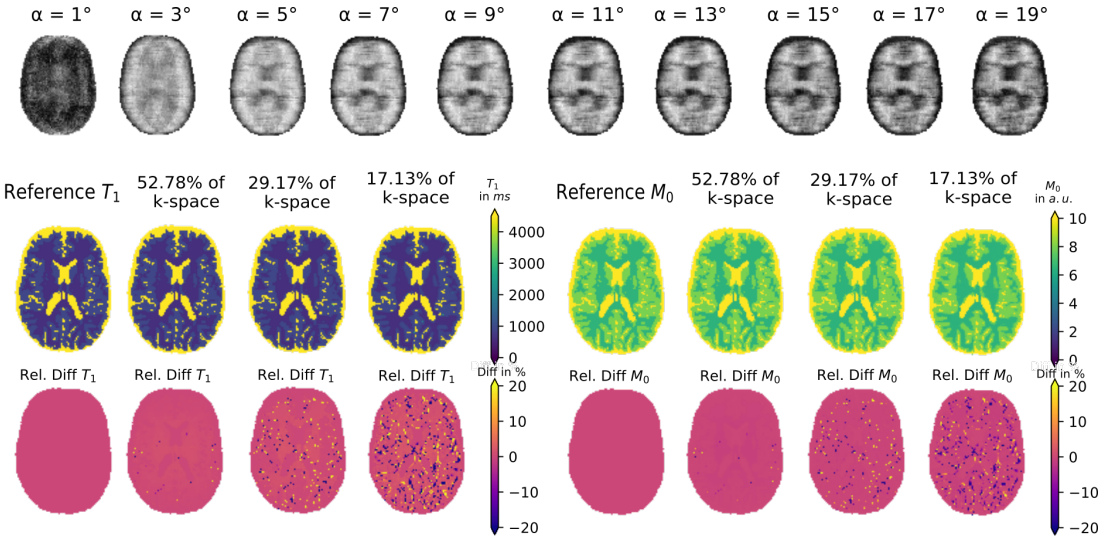


Figure 4.1: Reference and predicted T_1 and M_0 for different acceleration factors. The top row shows the undersampled input image series for $R = 5.84$. Bottom left and right figures show reference and predicted T_1 and M_0, respectively, with pixelwise relative difference maps below.

Table 4.2: nRMSE and SSIM for estimated M_0 and T_1 maps for acceleration factors $R = 1.89$, $R = 3.43$, $R = 5.84$.

Acceleration factor	nRMSE (%)		SSIM (%)	
	M_0	T_1	M_0	T_1
R = 1.89	0.47	1.01	99.7	97.18
R = 3.43	1.62	2.8	96.74	94.52
R = 5.84	2.96	5.72	93.1	90.76

Figure 4.2 shows exemplary learned filter kernels of the first convolution in image domain. Since the input contains 20 channels and the successive layer is required to have 20 feature maps, this leads to a total of 20 $3 \times 3 \times 20$ convolution kernels.

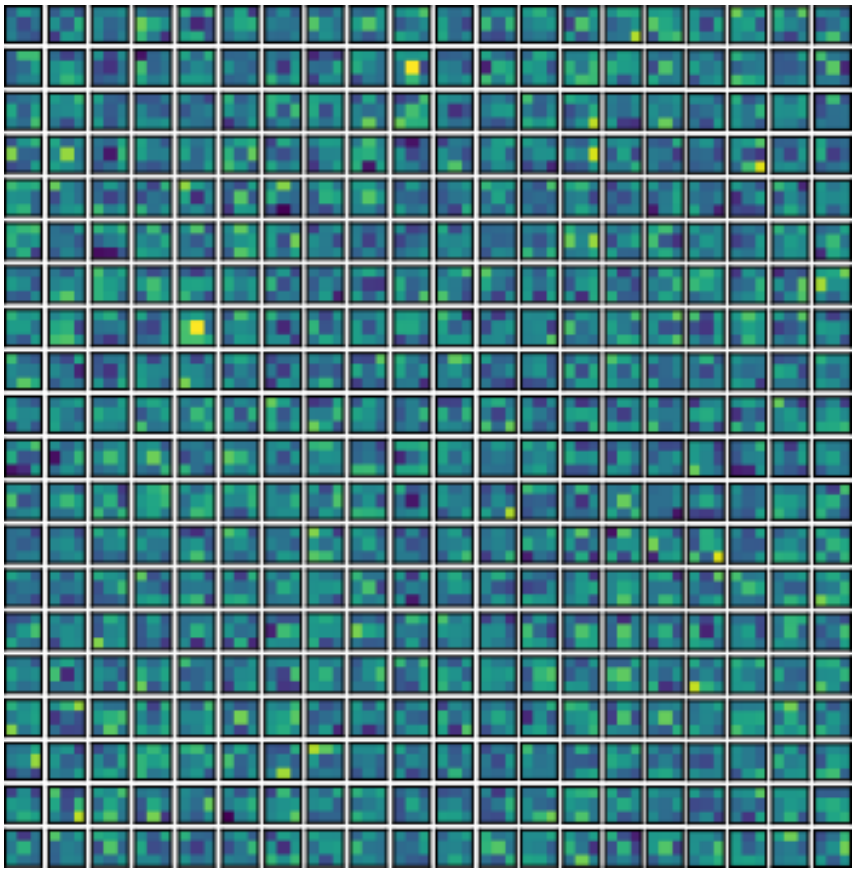


Figure 4.2: 20 learned filter kernels of size $3 \times 3 \times 20$ of the first convolution in image domain.

4.1.2 k-Space Domain

Table 4.3 contains the hyperparameters for training the U-Net in k-space domain.

Table 4.3: Experimental setup in k-space.

Hyperparameters	
#training samples	895
#test samples	100
Epochs	1000
Batch size	12
Stepsize α	0.001
Decay rate β_1	0.9
Decay rate β_2	0.999

Figures 4.3 and 4.4 show the predicted parameter maps without acceleration and with acceleration of $R = 1.89$, whereby the U-Net was trained on k-space data. Table 4.4 provides the corresponding quantitative results. Quantitative measures were only taken with respect to all anatomical structures and the background was excluded, since it does not contain information of interest.

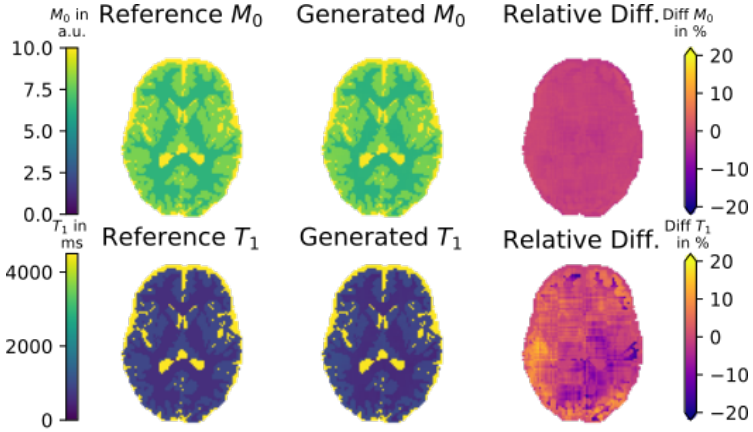


Figure 4.3: Reference and predicted M_0 and T_1 for training on phantom data in k-space without acceleration. For both parameter maps the pixelwise relative difference map is shown on the right.

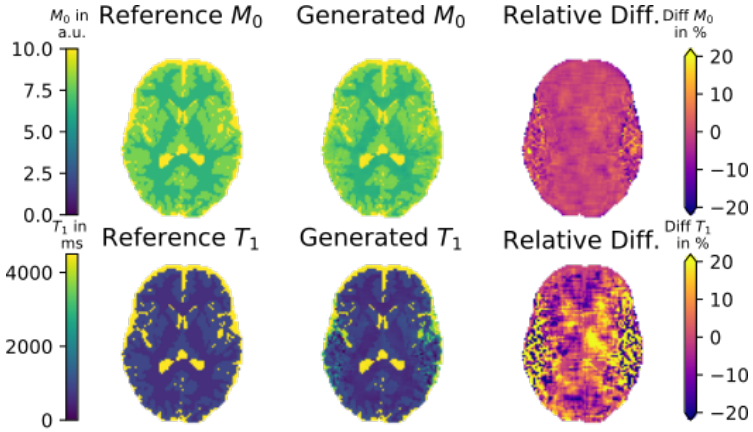


Figure 4.4: Reference and predicted M_0 and T_1 for training on phantom data in k-space data with acceleration $R = 1.89$. For both parameter maps the pixelwise relative difference map is shown on the right.

Table 4.4: nRMSE and SSIM for estimated M_0 and T_1 maps for learning on k-space data without acceleration ($R = 1.0$) and with acceleration factor $R = 1.89$.

Acceleration factor	nRMSE (%)		SSIM (%)	
	M_0	T_1	M_0	T_1
R = 1.0	0.5	0.56	99.15	94.38
R = 1.89	2.39	4.23	93.28	90.02

4.2 Transfer-learning with in vivo data

Table 4.5 contains the hyperparameters for training the pre-trained U-Net on in vivo data in image domain.

Table 4.5: Experimental setup for transfer-learning on in vivo data.

Hyperparameters	
#training samples	378
#test samples	42
Epochs	1200
Batch size	8
Stepsize α	0.001
Decay rate β_1	0.9
Decay rate β_2	0.999

Figure 4.5 shows reference and predicted parameter maps for undersampled VFA in vivo images of a healthy volunteer. The network was pre-trained with the brain phantom training dataset and used as an initialization for a transfer-learning scheme with the acquired in vivo data. The respective quantitative results are given in Table 4.6.

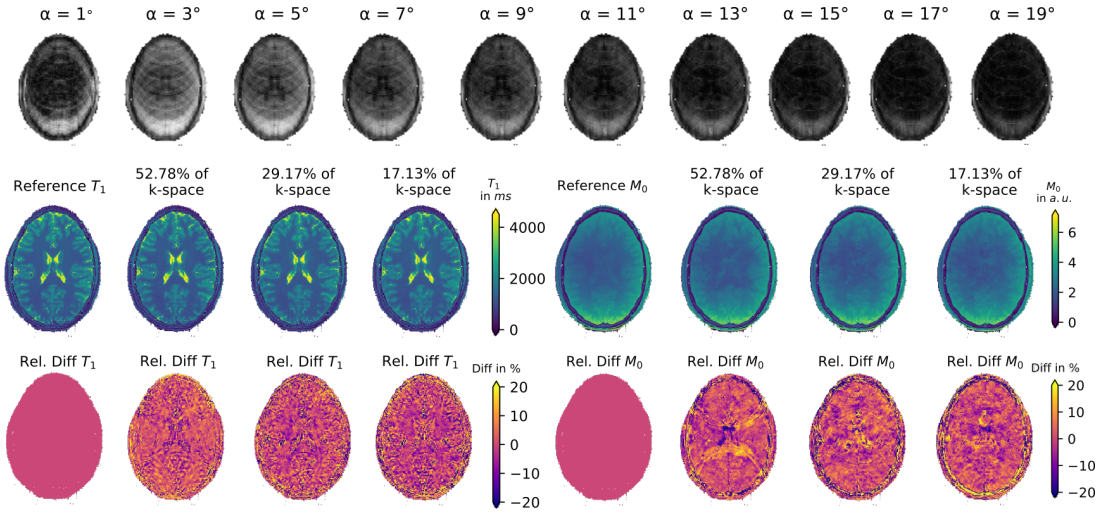


Figure 4.5: Reference and predicted T_1 and M_0 for different acceleration factors. The top row shows the undersampled input image series for $R = 5.84$. Bottom left and right figures show reference and predicted T_1 and M_0, respectively, with pixelwise relative difference maps below.

Table 4.6: nRMSE and SSIM for estimated M_0 and T_1 maps for fine-tuning in vivo data for acceleration factors $R = 1.89$, $R = 3.43$, $R = 5.84$.

Acceleration factor	nRMSE (%)		SSIM (%)	
	M_0	T_1	M_0	T_1
R = 1.89	2.28	2.45	86.42	92.26
R = 3.43	2.76	3.07	81.21	88.82
R = 5.84	3.12	3.56	81.09	86.52

Figure 4.6 shows the line plots for the selected line of the reference and generated T_1 map shown at the top.

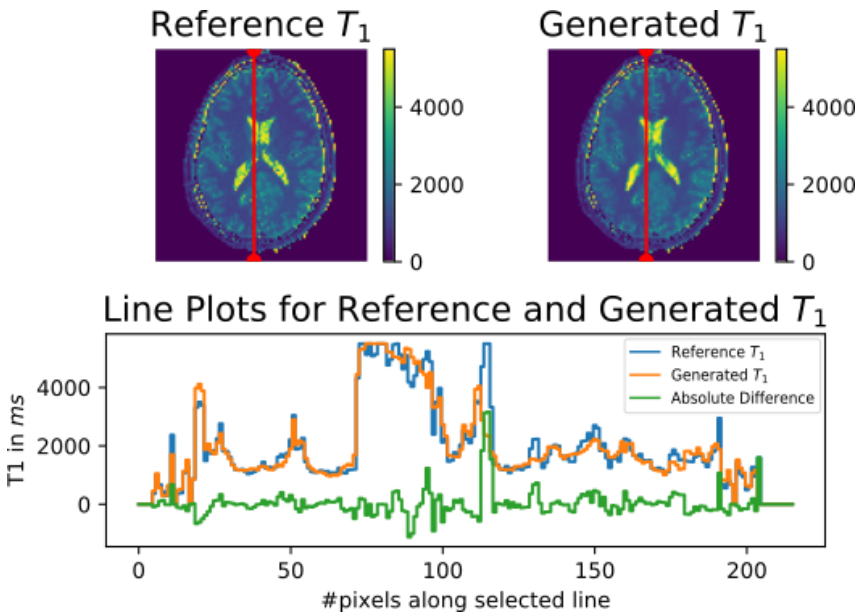


Figure 4.6: Line plots along selected line through reference and generated T_1 for acceleration factor $R = 1.89$. Top left and right figures show the reference map for T_1 and the generated map for T_1, respectively, including the chosen line for the line plots.

Figure 4.7 shows the reference and generated T_1 from the same slice, comparing the two parameter maps in more detail and highlighting some of the remaining artifacts in the prediction map.

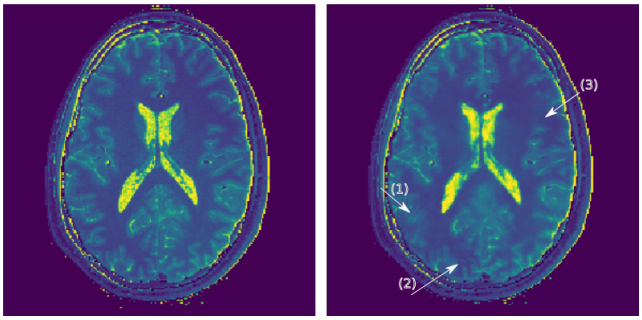


Figure 4.7: Reference (left) and generated (right) T_1 for acceleration factor $R = 1.89$ showing remaining artifacts.

4.3 B_1 Inhomogeneity Maps

Table 4.7 contains the hyperparameters for training the U-Net in image domain with an additional output channel for the B_1 inhomogeneity map.

Table 4.7: Experimental setup in image domain for parameter and inhomogeneity maps.

Hyperparameters	
#training samples	895
#test samples	100
Epochs	1600
Batch size	8
Stepsize α	0.001
Decay rate β_1	0.9
Decay rate β_2	0.999

Figure 4.8 shows predicted M_0 and T_1 maps and the additionally predicted B_1 inhomogeneity map with the respective groundtruth maps for accelerated ($R = 1.89$) VFA input images. The corresponding quantitative results can be found in Table 4.8. Figure 4.9 shows some exemplary feature maps of the trained U-Net of an earlier and a later layer.

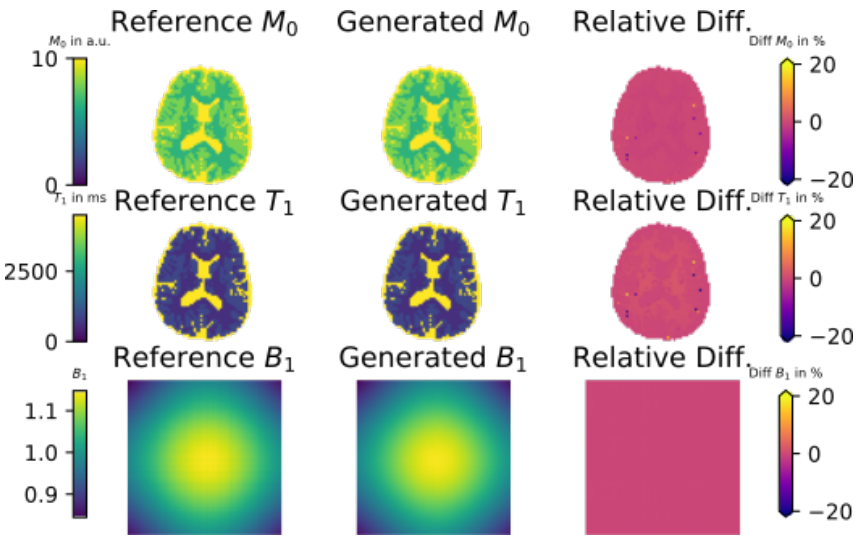
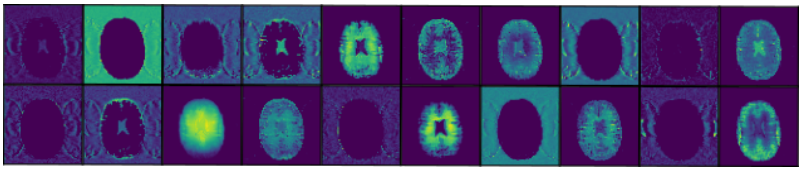


Figure 4.8: Reference and predicted M_0 (top row), T_1 (middle row), and B_1 (bottom row) with acceleration factor $R = 1.89$ for brain phantom data.

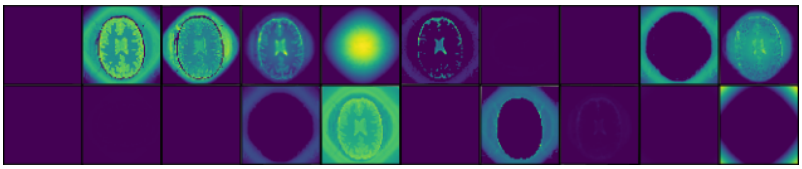
Table 4.8: nRMSE and SSIM for estimated M_0, T_1 and B_1 inhomogeneity maps for for acceleration factor $R = 1.89$ for brain phantom data.

Acceleration factor	nRMSE (%)			SSIM (%)		
	M_0	T_1	B_1	M_0	T_1	B_1
R = 1.89	0.57	1.24	0.52	99.53	96.2	99.95

Figures 4.10 and 4.11 depict predictions for M_0, T_1, and B_1 for two exemplary in vivo brain slices. The parameter and inhomogeneity maps were generated via transfer-learning from the pre-trained network. Corresponding quantitative values can be found in Table 4.9. The experimental setup is summarized in Table 4.5.



(a) Feature maps of an early layer.



(b) Feature maps of the penultimate layer.

Figure 4.9: (a) Feature maps of an early layer after the first convolution with the input. Each feature map is the result of a $3 \times 3 \times 20$ convolution with all 20 input channels. (b) Feature maps of the penultimate layer.

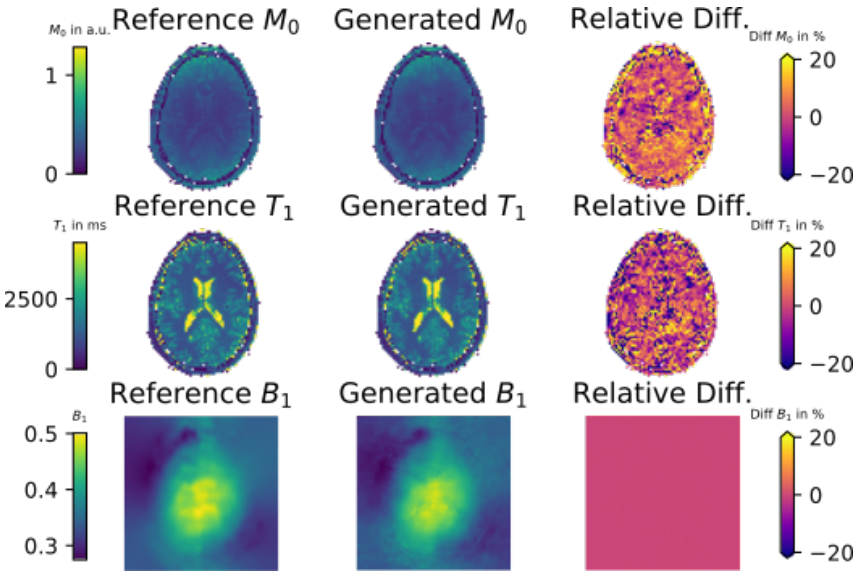


Figure 4.10: Reference and predicted M_0 (top row),T_1 (middle row) and B_1 (bottom row) with acceleration factor $R = 1.89$ for in vivo data sample 1.

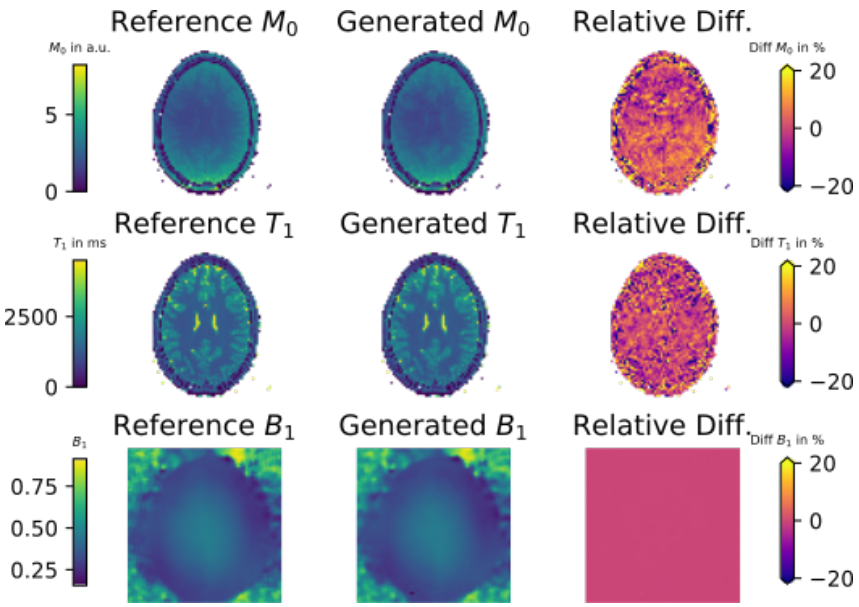


Figure 4.11: Reference and predicted M_0 (top row),T_1 (middle row) and B_1 (bottom row) with acceleration factor $R = 1.89$ for in vivo data sample 2.

Table 4.9: nRMSE and SSIM for estimated M_0, T_1 and B_1 inhomogeneity maps with acceleration factor $R = 1.89$ for brain phantom data.

Sample	Acceleration factor	nRMSE (%)			SSIM (%)		
		M_0	T_1	B_1	M_0	T_1	B_1
1	R = 1.89	2.53	5.06	2.84	94.78	86.96	99.7
2	R = 1.89	2.36	2.58	2.07	85.48	89.84	97.92

4.4 Systematic Errors

In addition to the previous experiments, a few systematic erros were simulated in the input images and were fed into the trained U-Net in inference stage. The

U-Net was trained in image domain on the numerical brain phantom data with hyperparameter settings as listed in Table 4.1. The corresponding reference and predicted parameter maps of the exemplary slice that was used amongst all cases is shown in Figure 4.12, where no systematic error was added. Table 4.10 summarizes quantitative results for all simulated scenarios.

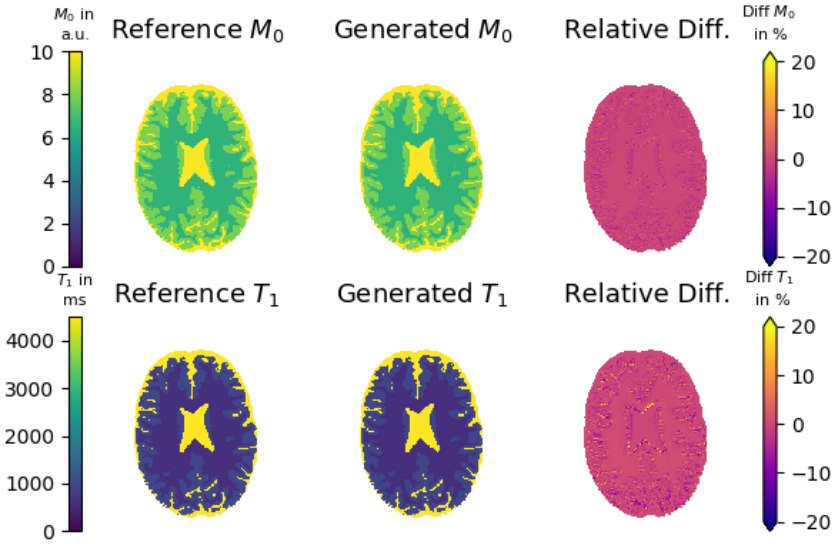


Figure 4.12: Reference and predicted parameter maps with pixelwise relative difference error maps of exemplary slice, where no systematic errors have been added.

Figure 4.13 shows qualitative results for multiplying M_0 and thus directly the input image series by a constant factor of 10, which corresponds to a different scaling factor in an experimental setting.

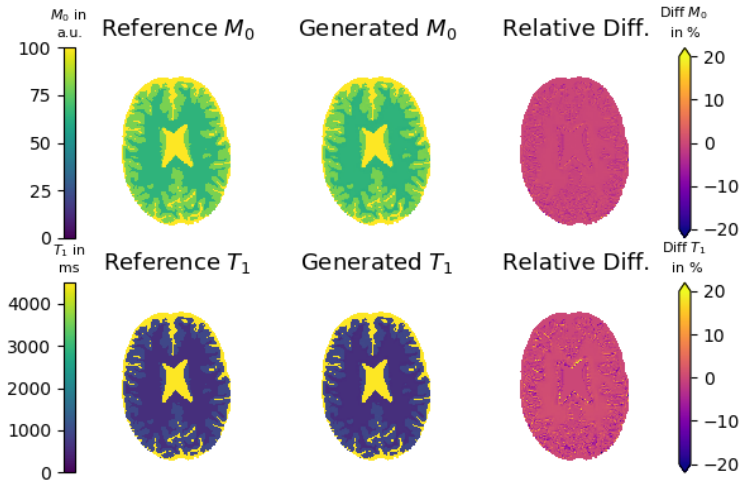


Figure 4.13: Reference and predicted parameter maps with pixelwise relative difference error maps for multiplying M_0 and S by a constant factor.

Figure 4.14 shows qualitative results for a simulated B_1 inhomogeneity, even though the U-Net never encountered input data comprised of imperfect flip angles during training. The nominal B_1 deviation is in the range of $[0.85, 1.15]$.

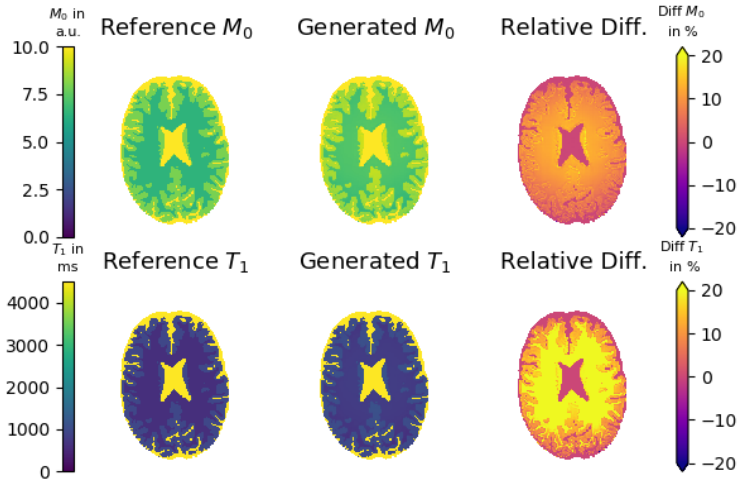


Figure 4.14: Reference and predicted parameter maps with pixelwise relative difference error maps for simulated B_1 inhomogeneity.

Figure 4.15 shows qualitative results for simulated motion during the measurement. This implies that the input images were rotated in-plane, whereas movement in the third dimension was not taken into account. The first case (left) considers a constant rotation of 8° for the flip angles $\{\alpha = 11° : 2° : 19°\}$ and the second case (right) deals with random motion of $\alpha \in [-10°, 10°],\ \forall\alpha$.

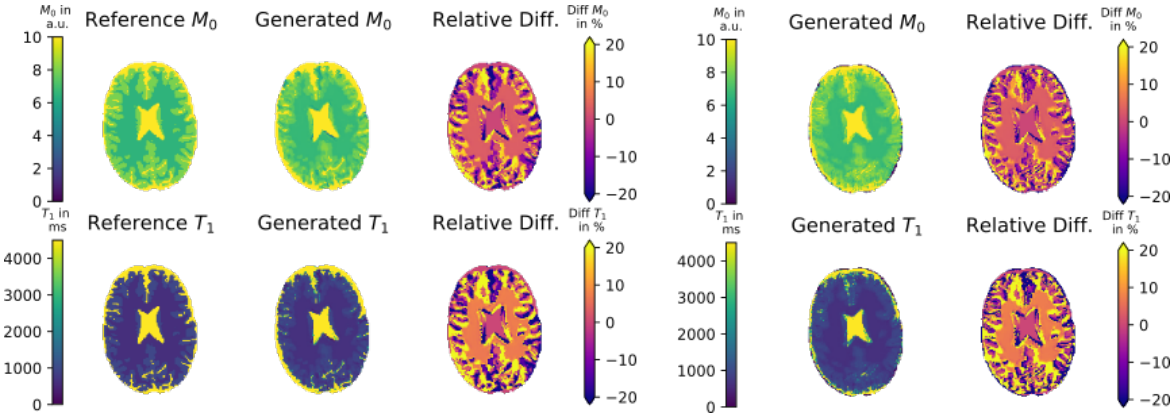


Figure 4.15: Reference and predicted parameter maps with pixelwise relative difference error map for two cases of simulated motion. On the left, a constant rotation for the last five flip angle measurements is induced and on the right, random motion during the entire measurement is simulated.

Figure 4.16 shows qualitative results for a fake tumor embedded in white matter. Figure 4.17 depicts corresponding line plots through the tumor of the reference and the generated T_1 map.

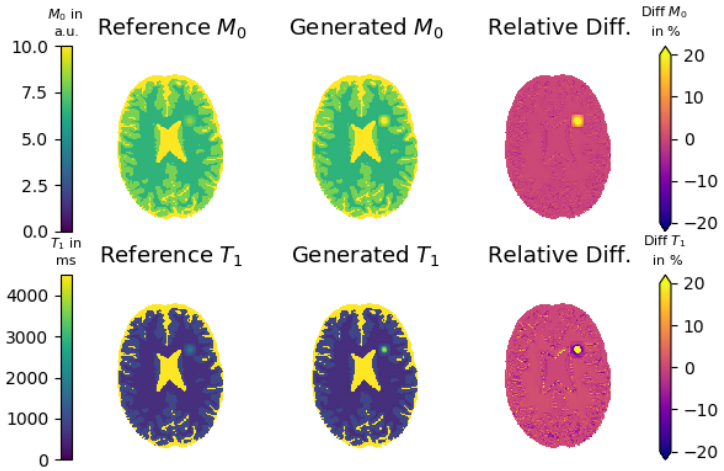


Figure 4.16: Reference and predicted parameter maps with pixelwise relative difference error map for a simulated tumor.

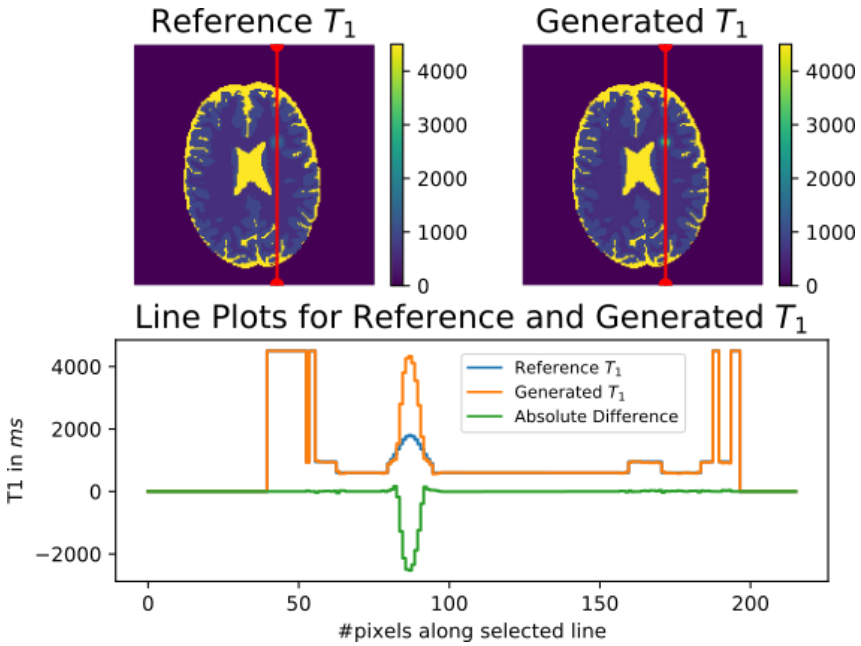


Figure 4.17: Line plots through reference and predicted T_1 parameter map for a fake tumor. Top left and right figures show the reference and the predicted T_1 map including the chosen line for the line plots.

Figure 4.18 shows qualitative results for accelerated VFA data for $R = 1.89$, $R = 3.43$, $R = 5.84$. In this case, undersampled data was fed into the U-Net, which was only trained on fully sampled data, thus it does not know how to correct for the undersampling artifacts that it has never encountered in training.

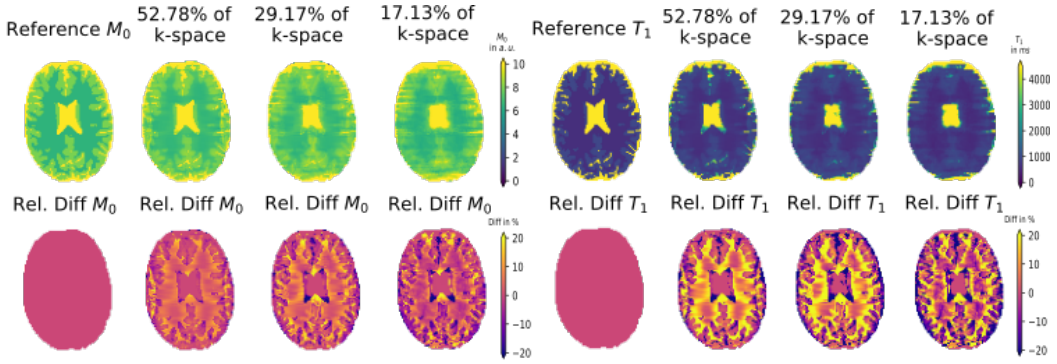


Figure 4.18: Reference and predicted parameter maps with pixelwise relative difference error maps for accelerated VFA data for $R = 1.89$, $R = 3.43$, $R = 5.84$ from a U-Net that was trained on fully sampled data.

Figure 4.19 shows qualitative results for noisy VFA data for 2%, 5%, 10% noise with respect to the flip angle with maximum signal intensity.

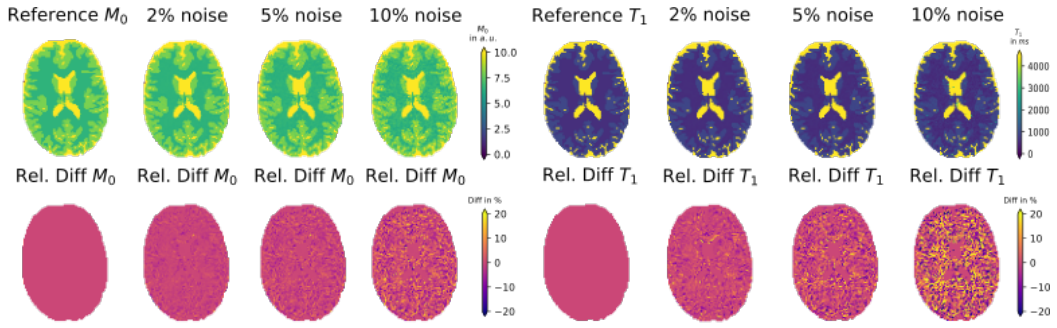


Figure 4.19: Reference and predicted parameter maps with pixelwise relative difference error maps for VFA input data with 2%, 5%, 10% noise.

Table 4.10: nRMSE and SSIM for estimated M_0 and T_1 parameter maps for different systematic error cases.

Scenario	nRMSE (%)		SSIM (%)	
	M_0	T_1	M_0	T_1
normal	0.49	0.22	99.18	93.21
constant factor	0.49	0.22	95.44	93.21
B_1 inhomogeneity	3.57	1.81	97.8	92.68
constant motion	6.12	14.32	82.77	76.58
random motion	7.42	13.14	82.31	75.63
tumor	0.94	1.18	98.98	93.66
$R = 1.89$	3.63	6.44	90.82	86.4
$R = 3.43$	5.41	9.77	85.05	80.45
$R = 5.84$	9.37	11.93	80.67	77.23
2% noise	0.63	0.29	96.97	93.99
5% noise	1.11	0.67	91.46	91.3
10% noise	2.96	3.1	83.71	83.62

5 Discussion

5.1 Numerical Simulation

5.1.1 Image Domain

Figure 4.1 and Table 4.2 demonstrate that quantifying M_0 and T_1 by means of a U-Net in image domain is possible. This was generally expected since there was recent work already done in this field, as briefly discussed in Section 2.6. Both of these two works, the first one learning T_2 maps and the second work learning T_1 maps from an IR experiment, use the underlying rather simple exponential relationship between signal intensity and parameters. In this case, when fitting T_1 from VFA images, the relation that the neural network has to approximate is more complex and not as straightforward as in the previous works. Nevertheless, the outcomes are promising with respect to future use of deep learning in clinical qMRI. Incorporating the signal model into the objective function ensures the physical validity of the model and thus includes valuable prior knowledge that should not be neglected.

Introducing undersampling into the input images as depicted in Figure 4.1 yields typical backfolding artifacts which are manifested in image domain. The U-Net was shown to be capable of removing these artifacts in addition to performing parameter mapping. Obviously, the smaller the acceleration factor R, which corresponds to less k-space lines being discarded, the better the final

reconstruction results. Even for an acceleration factor of $R = 5.84$, where only every 8^{th} line of k-space was kept in addition to a fully sampled k-space center, the quantitative results shown in Table 4.2 are still reasonably good considering the accurate structures visible in Figure 4.1. However, at this stage, one can qualitatively see that the amount of pixelwise outliers has increased in Figure 4.1.

Considering the quantitative results given by the normalized root-mean-square error nRMSE and the Structural Similarity Index SSIM, one can observe that for M_0 the error is slightly lower and the SSIM is a little better than for T_1. This might have two different causes. On the one hand, the neural network receives more information during training from the complex-valued M_0 target maps in the form of real and imaginary channels as opposed to T_1 references, which are simply real-valued. On the other hand, which is probably the decisive aspect, the relation between signal intensity and T_1 is of a much more complex nature and is nonlinear, unlike the linear scaling between M_0 and the signal intensity.

An example of learned filter kernels from the first convolution is shown in Figure 4.2, which show a lot of different characteristics. In general, they can not really be classified into certain expected filter types, since there are many more convolutions to follow which in combination achieve to learn the desired mapping. The deeper the layer and the more convolutions the input data has gone through, the larger the receptive field due to intermediate max pooling layers. The receptive field is the region of pixels that is relevant in a convolution, in this case the 3×3 neighbourhood in each pixel is defined as its receptive field. Deeper convolution kernels will therefore contain more abstract kernels which can hardly be interpretated by the human visual perception. They might therefore seem to be very random, but this level of complexity can simply not be captured by human understanding.

5.1.2 k-Space Domain

So far, no approaches of combining deep learning and qMRI by shifting the learning process into k-space exist. This is comprehensible given the fact that CNNs are mainly based on convolutions, where the most basic idea is that features such as shapes and edges from images get detected and extracted. In k-space these typical image-like structures are not present as such. Nevertheless, training the U-Net on fully sampled k-space input data works surprisingly well and the reconstructed parameter maps are very accurate, which is shown in Figure 4.3 and in Table 4.4. It seems to be capable to learn the mapping between input VFA series and parameter maps in k-space.

However, as soon as any undersampling is included in the input data, the method breaks down and the final results always contain backfolding artifacts, as depicted in Figure 4.4. It is understandable, considering that regular undersampling patterns, as they were used in this work, remove whole lines of k-space. The U-Net is then targeted to fill up the missing k-space lines by means of many convolutions, however, these convolution operations only have 3×3 filter kernels. It is a difficult task to extract information from only few k-space lines which are each surrounded by zero-filled lines, if only such small filter kernels are used.

Another important aspect is, as mentioned above, the fact that convolutions are aimed to detect actual geometric shapes, which are not present as such in k-space. It is therefore reasonable that such local operations, each working on a small neighbourhood, are not able to recover missing k-space lines in addition to mapping the input k-spaces to the output parameter maps. Therefore, the results for learning the parameter maps in k-space for fully sampled input data is remarkably good given the fact that almost the same architecture as in image domain is used. One might now think about simply using larger filter kernels, however, this rapidly introduces overfitting issues as this leads to an enormous increase in learnable parameters. This can then only be compensated for by

introducing massive amounts of training data, which is known to be difficult to accomplish when working with medical image data.

Hypothetically, letting the network learn the parameter mapping in k-space, does seem to offer the advantage that k-space data also contains inherent symmetries which could be exploited in the learning process. There have in general been approaches with non-local layers as explained by Wang et al. [43], which could potentially improve the current results significantly. In this work, a non-local, globally operating layer is placed immediately at one of the first layers of the architecture and is supposed to learn to fill up the missing k-space lines, followed by the regular U-Net which can then map the parameters from a more or less fully sampled k-space. Although this is a global layer, it is different to a fully-connected layer, since non-local neural networks consider relationships between different sites in an image [43] and fully-connected layers use learned weights. Using a fully-connected layer at the beginning, which receives input images of size 216×216, would exceed the available memory, this is a reminder that especially in deep learning there are definitely hardware limits that pose restrictions at times.

There are also other ideas to improve the network without using a global layer. It might be possible to obtain better results by including dilated convolutions, which are regular convolutions with an increased receptive field. If many dilated convolutions with varying receptive fields in parallel were combined, this could produce better parameter maps with less undersampling artifacts, however, the problem still remains that these are still all local operations in k-space.

There exists another approach including deep learning in k-space, coming from the field of accelerating parallel MRI [39] and from standard image reconstruction problems [38]. Both of these works have in common that they employ a regular U-Net in k-space followed by another U-Net in image domain. They are therefore not operating solely in k-space and have the opportunity

to correct remaining artifacts in image domain with the second U-Net. This raises the question whether it is actually possible to shift the complete learning process into k-space domain whilst keeping the proposed U-Net architecture. Another approach by Han et al. [15] works exclusively in k-space by rearranging k-space data into a low-rank Hankel matrix. The network's task is then to learn to fill up the missing points in k-space using low-rank Hankel matrix completion. This approach works entirely in k-space and produces good results. However, it is computationally more expensive in the preprocessing step.

In general, one must not forget that k-space does not contain locatable intensity values as they are present in an ordinary image but rather spatial frequencies. Therefore, each individual point in k-space holds information on all pixels in the image. If a point in k-space is not predicted accurately by means of a neural network, this can have drastic effects in the reconstructed final image as opposed to a few mismatched pixels in image domain, which can often be neglected. Mapping MRI parameters in k-space is in theory a good idea and seems feasible, however, extreme care with regards to the used methods has to be taken to ensure satisfactory results.

5.2 Transfer-learning with in vivo data

Transfer-learning is a powerful method if not enough training data is available. This was also the case in the course of this work, as it is usually very challenging, especially in the medical domain, to obtain sufficient amounts of in vivo training data. As shown in the results in Figure 4.5 and in Table 4.6, the predicted values for T_1 and M_0 are in good agreement with the reference. Structures are accurately reconstructed even though the network had only limited amount of data for training. The pixelwise relative difference maps show that there is more noise present which is only natural considering the fact that in vivo

images were used. Despite the decent quantitative results, there are still some backfolding artifacts remaining in the predictions. They are not very prominent due to noise for increasing acceleration. Unlike for the brain phantom, the quantitative values for T_1 are better than for M_0. This is most likely traced back to the fact that measurements from different volunteers were used, leading to different numerical ranges for the pseudo proton density M_0. Care had to be taken with regard to the scaling process to remove the substantial differences in M_0 ranges.

Figure 4.6 shows the intensity profile along a selected line of a generated T_1 map in comparison to its reference T_1 map. The course of intensity values along the selected line through the whole brain closely follows the reference line plot. This seems to be the case amongst the whole range of different T_1 values. Figure 4.7, which shows the prediction and reference T_1 maps from the same slice as in Figure 4.6, highlights some of the remaining artifacts. The labelled artifact (1) is clearly from the undersampling procedure, but artifacts (2) and (3) reveal minor structural changes in the prediction in comparison to the reference parameter map. It is not possible to locate the reason for their occurrence, as this might be due to backfolding, but it could also be due to the fact that the network was not trained well enough or was not able to generalize optimally on unseen data.

Further research on the amount of retrained layers would be necessary to optimize transfer-learning in clinical applications, since it could possibly also suffice to re-train only the upsampling path of the network or the last few output layers. However, if an approach like this is used, extreme care needs to be taken with respect to the original training data, since the two datasets must not be too different regarding numerical ranges and underlying structure.

5.3 B_1 Inhomogeneity Maps

As can be seen in Figure 4.8 and in Table 4.8, the results for learning additional B_1 inhomogeneity maps together with the parameter maps for the brain phantom data are surprisingly good and show accurate inhomogeneity maps. The network is able to distinguish outputs with anatomical structure like the parameter maps and those with smooth structures such as the inhomogeneity map, even though accurate flip angle knowledge is required for conventional fitting approaches.

Figure 4.9 shows exemplary feature maps of two different layers of the trained U-Net. One can cleary see that the feature maps of the earlier layer still contain backfolding artifacts which have not yet been corrected for. Moreover, the feature map holding information on the inhomogeneity map is embedded into the anatomical structures of the parameter maps. In contrast, the feature maps from the deeper layer, which is in fact the penultimate layer, contain isolated information on the parameter maps and on the B_1 inhomogeneity map, which indicates that the network was successful in separating the respective structures from each other.

Figures 4.10 and 4.11 and Table 4.9 show qualitative and quantitative results for predicting parameter and inhomogeneity maps by means of transfer-learning. The procedure for transfer-learning is the same as before with the small addition of adding a separate output channel for learning the B_1 inhomogeneity maps. The B_1 inhomogeneity map for the second sample is predicted very accurately and no remaining error can be detected in the pixel-by-pixel difference map. This is also the case for the first sample. However, since the groundtruth maps for the first sample were reconstructed from measurements which went through substantial undersampling, the reference map contains blocky areas in the frequency encoding direction. This effect is improved in the prediction by the

trained U-Net, where the overall structure is much smoother. Furthermore, some remaining structures of the parameter maps including undersampling artifacts can be detected in the predicted B_1 map, indicating that for the approach of transfer-learning, the predictions for B_1 are not entirely separable from the parameter maps.

In general, the results for predicting B_1 maps from measured in vivo data were expected to be of such a good quality, since the in vivo training data is compiled from four volunteers with a groundtruth B_1 map for each training slice. Since all reference maps are very similar to either one of the displayed inhomogeneity maps in Figures 4.10 and 4.11, it comes to no surprise that the results are indeed very good. It would therefore be interesting to include more measurements from other volunteers into the training process and to use one entire measurement originating from only one person as test data.

5.4 Systematic Errors

Multiplying M_0 and thus also the signal from the SPGR equation by a constant factor should not have any influence on the prediction, since the input data is standardized and the output labels are required to be in the range $[0, 1]$. The predicted parameter maps in Figure 4.13 seem to qualitatively confirm this idea, however, the quantitative values for M_0 have slightly deteriorated as listed in Table 4.10. The reason why the SSIM has seemingly decreased is due to the fact that this measure is taken with respect to the denormalized images, after they have been scaled back to their original values. Regarding the original prediction as displayed by the trained U-Net, there is no difference in the SSIM with respect to the reference in Figure 4.12. However, when M_0 is scaled by a constant factor, this directly influences its mean μ and standard deviation σ and has in consequence an effect on the nonlinear relation of the

SSIM and μ and σ. Therefore, this can only be observed for M_0, whereas the quantitative measures for T_1 remain unchanged.

Feeding the network with a VFA input series with a B_1 inhomogeneity also has a huge influence on the predictions as depicted in Figure 4.14, if the network was not specifically trained to recognize the inhomogeneities. The trained U-Net does not know how to deal with imperfect flip angles since it has never encountered them in training and therefore is not able to capture this feature in the resulting predictions. The influence of the B_1 inhomogeneity can be clearly seen in the pixelwise relative difference map.

The possibility that a patient moves during an exam was simulated in two different scenarios. In the first case, it can be assumed that after five flip angle measurements the patient moved in such a way that a constant rotation of 8° was imposed on the following five flip angle measurements. The second case considers motion where the patient moves arbitrarily, inducing a random rotation of $[-10°, 10°]$ for each measurement. As expected and as shown in Figure 4.15, this introduces blurring into the predictions and the anatomical structures are not resolved distinctly anymore. The random motion leads to predictions which are even more blurred and also to worse quantitative results, as listed in Table 4.10. This is likely due to the fact that for constant motion the network receives at least five input images which are not shifted whereas for random motion, none of the input images is in its original position. In general, these motion artifacts could potentially be learned by the U-Net provided sufficient data is included in the training set. This could either be with a constant motion behaviour, which should be fairly straightforward for the network to learn, or an entirely randomized motion pattern, which corresponds to a more realistic scenario.

In Figure 4.16 the predictions for a fake tumor in white matter are shown. The tumor introduces new intensity values to the parameter maps that the trained

U-Net has not seen before, which is especially critical for phantom data due to discrete value pools. Therefore, it is evident that this can not be predicted correctly by the network as this scenario has never occurred during training. Figure 4.17 depicts a line plot of the intensity values for T_1 through the tumor showing that it is only the tumor values that can not be estimated accurately. The reference map contains a smooth transition of the values from the border of white matter to the center of the tumor, which can not be captured by the network. One might argue that the U-Net should be able to predict pathologies such as a tumor accurately, even though it has not encountered it in training, by interpretating the input image series and its intensity course. This is contradicted by the fact that the network learns solely based on convolutions which aim to detect structure in the input rather than only focusing on the change of the intensity course in the input VFA images.

Figure 4.18 shows results for feeding subsampled VFA input data into the U-Net which was not trained on accelerated data. This is done for the three cases of $R = 1.89$, $R = 3.43$ and $R = 5.84$ and obviously, the network can not achieve to correct for the consequently arising backfolding artifacts. The higher the acceleration factor, the worse the predicted parameter map. This behaviour is also emphasized by quantitative results in Table 4.10.

The final scenario which was evaluated deals with different noise levels that were added to the VFA input series. The predictions can be seen in Figure 4.19. For adding noise such as 2% the results are quantitatively and qualitatively very good since this is in the range of added noise during the training. However, if this is increased up to a noise level of 15%, the network fails to recover denoised parameter maps, since it was not trained on this task. Nevertheless, displayed anatomical structures are still accurate, which is supported by quantitative values in Table 4.10.

6 Conclusion & Outlook

While deep learning has already seen a huge boost in standard medical image processing applications such as image segmentation and classification and further in accelerated multi-coil reconstruction problems, there have been only very few approaches of incorporating it into quantitative MRI. This master's thesis has shown the potential of learning M_0 and T_1 maps with a model-augmented U-Net.

For learning parameter maps on training data in image domain, the good qualitative and quantitative results show the enormous potential of using deep learning in qMRI. A comparison to learning on k-space data led to the conclusion that with the present architecture it is more meaningful to learn on data in image domain. However, there might be possible future options to enhance the architecture and render it more suitable for k-space learning, as discussed in Section 5.1.2.

It was shown that it is possible to additionally learn B_1 inhomogeneity maps in conjunction with the parameter maps, which can be included in the signal model of the objective function. Being able to learn B_1 inhomogeneity maps offers a great perspective for future applications since this would indeed spare the need for an additional measurement.

The network, after being trained on numerical brain phantom data, was transferred to work on single-coil in vivo brain data of four volunteers by means of

transfer-learning. This method has in the past shown to yield good results in case of limited training data. The pre-trained model serves as a good initialization point for further training with in vivo data. The acceleration potential of three scenarios with different acceleration factors of up to approximately 6 was shown for phantom as well as for in vivo data. Future research on the amount of re-trained layers would be interesting to further enhance the results and completely remove remaining backfolding artifacts to render it a clinically feasible method.

There are other existing architectures, such as ResNet, which was briefly mentioned as a previous approach of learning T_1 maps [34]. This could potentially also provide very good results as this type of architecture is usually employed for deeper networks with many layers. However, since the U-Net produced satifactory results and the architecture seems justifiable for qMRI, no other architectures were implemented and tested.

Other possible improvements include the incorporation of receive coils into the fitting process, an issue which is rarely dealt with in deep learning based reconstruction problems. Furthermore, the subsampling strategy could be adapted to non-cartesian sampling patterns as they are used with compressed sensing based algorithms which require incoherent artifacts in some transform domain.

Bibliography

[1] *A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way*. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53. Accessed: 2019-10-03 (cit. on p. 35).

[2] Zeynettin Akkus et al. "Deep Learning for Brain MRI Segmentation: State of the Art and Future Directions." In: *Journal of Digital Imaging* 30.4 (Aug. 2017), pp. 449–459. ISSN: 1618-727X. DOI: 10.1007/s10278-017-9983-4. URL: https://doi.org/10.1007/s10278-017-9983-4 (cit. on p. 3).

[3] *An Introduction to different Types of Convolutions in Deep Learning*. https://towardsdatascience.com/types-of-convolutions-in-deep-learning-717013397f4d. Accessed: 2019-11-08 (cit. on p. 39).

[4] Stefan Blüml et al. "Spin-lattice relaxation time measurement by means of a TurboFLASH technique." In: *Magnetic resonance in medicine* 30 3 (1993), pp. 289–95 (cit. on p. 14).

[5] Hai-Ling Margaret Cheng and Graham A Wright. "Rapid high-resolution T1 mapping by variable flip angles: Accurate and precise measurements in the presence of radiofrequency field inhomogeneity." In: *Magnetic Resonance in Medicine* 55.3 (2006), pp. 566–574. DOI: 10.1002/mrm.20791. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mrm.20791. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.20791 (cit. on pp. 11, 14–16).

[6] François Chollet et al. *Keras*. https://keras.io. 2015 (cit. on p. 3).

[7] Kelvin Chow et al. "Saturation Recovery Single-Shot Acquisition (SASHA) for myocardial T-1 mapping." In: *Magnetic resonance in medicine : official journal of the Society of Magnetic Resonance in Medicine / Society of Magnetic Resonance in Medicine* 71 (June 2014). DOI: 10.1002/mrm.24878 (cit. on p. 3).

[8] Balázs Csanád Csáji. "Approximation with Artificial Neural Networks." MA thesis. Faculty of Sciences, Etvs Lornd University, Hungary, 2001 (cit. on p. 17).

[9] Salman Ul Hassan Dar and Tolga Çukur. "A Transfer-Learning Approach for Accelerated MRI using Deep Neural Networks." In: *CoRR* abs/1710.02615 (2017). arXiv: 1710.02615. URL: http://arxiv.org/abs/1710.02615 (cit. on p. 45).

[10] Sean C. L. Deoni, Brian K. Rutt, and Terry M. Peters. "Rapid combined T1 and T2 mapping using gradient recalled acquisition in the steady state." In: *Magnetic resonance in medicine* 49 3 (2003), pp. 515–26 (cit. on pp. 3, 14).

[11] "Front Matter." In: *Quantitative MRI of the Brain*. John Wiley & Sons, Ltd, 2004, pp. i–xvi. ISBN: 9780470869529. DOI: 10.1002/0470869526.fmatter. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/0470869526.fmatter. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/0470869526.fmatter (cit. on p. 2).

[12] Xavier Glorot and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks." In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. Ed. by Yee Whye Teh and Mike Titterington. Vol. 9. Proceedings of Machine Learning Research. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 249–256. URL: http://proceedings.mlr.press/v9/glorot10a.html (cit. on p. 34).

[13] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016 (cit. on pp. 20, 21, 38, 41).

[14] Jacques Hadamard. "Sur les problèmes aux dérivées partielles et leur signification physique." In: *Princeton University Bulletin* 13 (1902), pp. 49–52 (cit. on p. 13).

[15] Yoseob Han and Jong Chul Ye. "k-Space Deep Learning for Accelerated MRI." In: *CoRR* abs/1805.03779 (2018). arXiv: 1805.03779. URL: http://arxiv.org/abs/1805.03779 (cit. on p. 76).

[16] Kaiming He et al. "Deep Residual Learning for Image Recognition." In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385 (cit. on pp. 25, 26).

[17] Kaiming He et al. "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification." In: *IEEE International Conference on Computer Vision (ICCV 2015)* 1502 (Feb. 2015). DOI: 10.1109/ICCV.2015.123 (cit. on p. 34).

[18] Chang Min Hyun et al. "Deep learning for undersampled MRI reconstruction." In: *Physics in Medicine & Biology* 63.13 (June 2018), p. 135007. DOI: 10.1088/1361-6560/aac71a. URL: https://doi.org/10.1088%2F1361-6560%2Faac71a (cit. on p. 3).

[19] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." In: *CoRR* abs/1502.03167 (2015). arXiv: 1502.03167. URL: http://arxiv.org/abs/1502.03167 (cit. on pp. 37, 38).

[20] Kyong Hwan Jin et al. "Deep Convolutional Neural Network for Inverse Problems in Imaging." In: *CoRR* abs/1611.03679 (2016). arXiv: 1611.03679. URL: http://arxiv.org/abs/1611.03679 (cit. on p. 3).

[21] Noorshaida Kamaruddin et al. "Relationship between Hounsfield unit in CT scan and gray scale in CBCT." In: *AIP Conference Proceedings* 1791.1 (2016), p. 020005. DOI: 10.1063/1.4968860 (cit. on p. 2).

[22] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2014. arXiv: 1412.6980 [cs.LG] (cit. on p. 42).

[23] Yann LeCun et al. "Object Recognition with Gradient-Based Learning." In: *Shape, Contour and Grouping in Computer Vision*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 319–345. ISBN: 978-3-540-46805-9. DOI: 10.1007/3-540-46805-6_19 (cit. on pp. 20, 21).

[24] Lisa Leroi. "Quantitative MRI : towards fast and reliable T1, T2 and proton density mapping at ultra- high field." PhD thesis. Université Paris-Saclay, 2018 (cit. on p. 9).

[25] Andreas Lesch et al. "Ultrafast 3D Bloch–Siegert B-mapping using variational modeling." In: *Magnetic Resonance in Medicine* 81.2 (), pp. 881–892. DOI: 10.1002/mrm.27434. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mrm.27434. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.27434 (cit. on p. 47).

[26] Fang Liu, Li Feng, and Richard Kijowski. "MANTIS: Model-Augmented Neural neTwork with Incoherent k-space Sampling for efficient MR parameter mapping." In: *Magnetic Resonance in Medicine* 82.1 (2019), pp. 174–188. DOI: 10.1002/mrm.27707. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mrm.27707 (cit. on pp. 23–25, 41).

[27] Fang Liu et al. "Fast Realistic MRI Simulations Based on Generalized Multi-Pool Exchange Tissue Model." In: *IEEE Transactions on Medical Imaging* 36.2 (Feb. 2017), pp. 527–537. ISSN: 1558-254X. DOI: 10.1109/TMI.2016.2620961 (cit. on p. 47).

[28] David Look and D. Locker. "Time saving in measurement of NMR and EPR relaxation times." In: *Review of Scientific Instruments* 41 (Feb. 1970), pp. 250–251. DOI: 10.1063/1.1684482 (cit. on p. 3).

[29] Oliver Maier et al. “Rapid T1 quantification from high resolution 3D data with model-based reconstruction.” In: *Magnetic Resonance in Medicine* 81 (Oct. 2018). DOI: 10.1002/mrm.27502 (cit. on p. 49).

[30] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL: http://tensorflow.org/ (cit. on p. 3).

[31] Chigozie Nwankpa et al. “Activation Functions: Comparison of trends in Practice and Research for Deep Learning.” In: *CoRR* abs/1811.03378 (2018) (cit. on p. 35).

[32] Augustus Odena, Vincent Dumoulin, and Chris Olah. “Deconvolution and Checkerboard Artifacts.” In: *Distill* (2016). DOI: 10.23915/distill.00003. URL: http://distill.pub/2016/deconv-checkerboard (cit. on pp. 39, 40).

[33] Adam Paszke et al. “Automatic differentiation in PyTorch.” In: *NIPS-W*. 2017 (cit. on p. 3).

[34] E. Ribeiro Sabidussi et al. “A deep learning approach to T1 mapping in quantitative MRI.” 2019 (cit. on pp. 24–27, 83).

[35] John P. Ridgway. “Cardiovascular magnetic resonance physics for clinicians: part I.” In: *Journal of Cardiovascular Magnetic Resonance* 12 (2010) (cit. on pp. 7, 9).

[36] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation.” In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab et al. Cham: Springer International Publishing, 2015, pp. 234–241. ISBN: 978-3-319-24574-4 (cit. on pp. 29, 30, 32).

[37] Peter Schmitt et al. “Inversion recovery TrueFISP: quantification of T(1), T(2), and spin density.” In: *Magnetic resonance in medicine* 51 4 (2004), pp. 661–7 (cit. on pp. 3, 10).

[38] Roberto Souza and Richard Frayne. *A Hybrid Frequency-domain/Image-domain Deep Network for Magnetic Resonance Image Reconstruction.* 2018. arXiv: 1810.12473 [eess.IV] (cit. on p. 75).

[39] Anuroop Sriram et al. *GrappaNet: Combining Parallel Imaging with Deep Learning for Multi-Coil MRI Reconstruction.* 2019. arXiv: 1910.12325 [eess.IV] (cit. on p. 75).

[40] Nikola Stikov et al. "On the accuracy of T1 mapping: Searching for common ground." In: *Magnetic Resonance in Medicine* 73.2 (2015), pp. 514–522. DOI: 10.1002/mrm.25135. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mrm.25135. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.25135 (cit. on pp. 10–12).

[41] Andrew J. Taylor et al. "T1 Mapping." In: *JACC: Cardiovascular Imaging* 9.1 (2016), pp. 67–81. ISSN: 1936-878X. DOI: 10.1016/j.jcmg.2015.11.005. eprint: http://imaging.onlinejacc.org/content/9/1/67.full.pdf. URL: http://imaging.onlinejacc.org/content/9/1/67 (cit. on pp. 10, 11).

[42] Stéfan van der Walt et al. "scikit-image: Image processing in Python." In: *CoRR* abs/1407.6245 (2014). arXiv: 1407.6245. URL: http://arxiv.org/abs/1407.6245 (cit. on p. 53).

[43] Xiaolong Wang et al. "Non-local Neural Networks." In: *CoRR* abs/1711.07971 (2017). arXiv: 1711.07971. URL: http://arxiv.org/abs/1711.07971 (cit. on p. 75).

[44] Zhou Wang et al. "Image Quality Assessment: From Error Visibility to Structural Similarity." In: *Trans. Img. Proc.* 13.4 (Apr. 2004), pp. 600–612. ISSN: 1057-7149 (cit. on p. 53).

[45] *What is the differences between artificial neural network (computer science) and biological neural network?* https://www.quora.com/What-is-the-differences-between-artificial-neural-network-computer-

science-and-biological-neural-network. Accessed: 2019-10-03 (cit. on p. 17).

[46] Ding-Xuan Zhou. *Universality of Deep Convolutional Neural Networks*. 2018. arXiv: 1805.10769 [cs.LG] (cit. on p. 21).

[47] Bo Zhu et al. "Image reconstruction by domain transform manifold learning." In: *CoRR* abs/1704.08841 (2017). arXiv: 1704.08841. URL: http://arxiv.org/abs/1704.08841 (cit. on p. 3).

List of Figures

List of Tables