Volume-based Translation of Phase-Cycle Balanced Steady State Free Precession MRI Data to Diffusion Tensors

Thesis submitted in partial fulfilment of the requirements for the degree

Master of Science

Graduate Training Centre of Neuroscience
Faculty of Science
Faculty of Medicine
University of Tübingen

Presented by
Fabian Klopfer, M.Sc.
from Villingen-Schwenningen, Germany

Tübingen, 2024

Thesis Advisor	Dr. Rahel Heule Department for High-field Magnetic Resonance Max Planck Institute for Biological Cybernetics Center for MR Research Children's Hospital University of Zurich
Second Advisor	Prof. Dr. Klaus Scheffler Department for High-field Magnetic Resonance Max Planck Institute for Biological Cybernetics

Disclosures

- I affirm that I have written the dissertation myself and have not used any sources and aids other than those indicated.
- I affirm that I have not included data generated in one of my laboratory rotations and already presented in the respective laboratory report.
- I affirm that generative AI has been used only as a means of correcting spelling, and grammar errors as well as improving the overall language of the presented work.

Tübingen, March 2024,

Abstract:

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Contents

1. Introduction

Diffusion-weighted imaging (DWI) has revolutionized the field of medical imaging by providing a unique window into the microstructural organization of tissues [39]. At its core, DWI leverages the diffusion of water molecules to reveal the underlying architecture of cellular structures. This information is invaluable in clinical settings, as it allows for the diagnosis, characterization, and monitoring of a wide array of conditions, including ischemic stroke, brain tumors, and various neurodegenerative diseases [77].

Traditional DWI techniques, while powerful, typically rely on specialized pulse sequences that can be time-consuming in acquisition [3]. These sequences involve the application of strong diffusion gradients, which can lead to prolonged scan times and potentially compromise patient comfort. Furthermore, the quality of DWI data can be affected by factors like low signal-to-noise ratios, motion artifacts and signal dropouts, particularly at high b-values (which are necessary for capturing subtle diffusion characteristics) [85]. These limitations underscore the need for innovative approaches that can overcome the challenges associated with conventional DWI.

Balanced steady-state free precession (bSSFP) imaging has emerged as a promising avenue for addressing these challenges. bSSFP sequences offer several key advantages over traditional DWI, including faster acquisition times, superior signal-to-noise ratio, and reduced susceptibility to specific artifacts [75]. The bSSFP signal is sensitive to a variety of tissue properties, including T1 and T2 relaxation times, proton density, and off-resonance effects [36, 4]. In [54, 55] the authors explore the connection between bSSFP data and white matter tracts based on the signal frequency response profile. However, extracting the specific diffusion-related information from this complex signal requires sophisticated analysis techniques.

Deep learning, a subfield of machine learning that focuses on artificial neural networks, has demonstrated remarkable success in a wide range of image analysis tasks [12]. In particular, convolutional neural networks (CNNs), which are designed to exploit the spatial structure of images, have achieved state-of-the-art performance in areas like image classification, segmentation, and enhancement [11, 72, 33]. This thesis investigates the potential of deep learning, specifically CNNs, to accurately estimate diffusion parameters from bSSFP data. We hypothesize that a carefully designed and trained CNN can effectively learn the complex mapping between bSSFP images and the underlying diffusion tensor, thus providing a rapid and reliable alternative to traditional DWI. Previous work has shown promising results in generating diffusion tensor scalar maps from phase-cycle bSSFP data, highlighting the potential of this approach [5] to retrieve additional clinically relevant quantitative parameters beside relaxometry [37] from phase-cycled bSSFP data.

The following chapters will lay the groundwork for this investigation by providing a comprehensive overview of the relevant background knowledge. We will first delve into the fundamental principles of magnetic resonance imaging (MRI), including the physics of nuclear magnetic resonance and the technical aspects of MRI scanners. We will then explore the unique characteristics of bSSFP imaging and the various factors that influence the bSSFP signal. Next, we will examine the concept of diffusion-weighted imaging, the mathematical models used to describe diffusion, and the different diffusion parameters that can be derived from DWI data. Then, we will present a comprehensive overview of deep learning, focusing on the fundamentals of neural

networks (with a particular emphasis on CNNs), a common architecture, and training strategies. With this foundation in place, we will present our proposed deep learning methodology, including the architecture of our CNN, the training process, and the evaluation metrics used to assess its performance. Finally, we will present the results of our experiments, discuss the implications of our findings for the future of DWI, and identify potential avenues for further research.

2. Background & Related Work

In this chapter, relevant background knowledge and related previously published work is discussed.

2.1. Magnetic Resonance Imaging

2.1.1. Fundamentals of MRI

Magnetic resonance imaging (MRI) uses the ability of nuclei to absorb and emit energy in the radio frequency (RF) spectrum to generate images of organisms and objects. This ability is called Nuclear Magnetic Resonance and depends on charge, spin, and mass, which are the intrinsic properties of particles.
Nuclear Magnetic Resonance-active Nuclei are those atoms with an odd number of protons and/or neurtons[58, 29]. They have a spin angular momentum defined as

$$S = \hbar I,$$

where I is the spin operator and $\hbar$ is Planck's constant normalized by 2π. The odd number of nucleons also induces a magnetic dipole moment

$$\mu = \gamma S.$$

γ [MHz/T] is called the gyromagnetic ratio and is unique to each nuclear magnetic resonance-active nuclei. It is the ratio of the magnetic dipole moment $\mu \left[\frac{N \cdot m}{T}\right]$ to the spin of a particle $S[N \cdot m \cdot s]$. The gyromagnetic ratio governs the frequency of precession and is measured empirically.
Specifically, ^{1}H has a spin of 1/2, a high natural abundance of 0.998 and is therefore used primarily for clinical MRI. Its gyromagnetic ratio is $\gamma = 42.57$ MHz/T.
When placing such nuclei into a magnetic field, the spins have a tendency to align with its direction, leading to a net or bulk magnetization per unit volume (or voxel) v,

$$M = \sum_{v \in V} \mu_v.$$

Additionally, due to spin and mass, the nucleus exhibits an angular momentum, i.e. it precesses e.g. in a gravitational field. The dynamics of this angular momentum or pecession can be described by

$$\frac{d\mu}{dt} = \mu \times \gamma B,$$

when considering a single nucleus and for a summing over a unit volume

$$\frac{dM}{dt} = M \times \gamma B.$$

The combination of magnetic and angular momentum causes the nucleus to precess in a magnetic field as well with a specific frequency — called the Larmor frequency:

$$\omega = \gamma B,$$

where B $[T]$ is the strength of a magnetic field and γ is the gyromagnetic ratio.

A scanner consists of a primary superconductive electromagnetic coil, that is surrounded by a cryostat and thermal insulation, an excitation coil, spatial localization coils, and a receiver coil, as shown in 2.1. The excitation and the receiving coil are merged into one RF coil here.

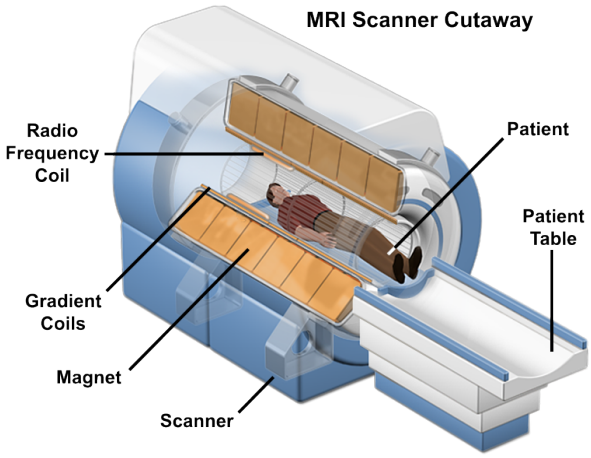

Figure 2.1. *MRI scanner with a part cut-away such that the main components are visible [48].*

The **primary electromagnet** provides the main magnetic field B_0 to polarize the NMR-active nuclei with field strengths up to $14.2T$, but usually about $1.5T$ to $3T$ in medical applications. This is several orders of magnitude larger than the earth magnetic field which is approximately in the range of $[22, 67]$ μT. To provide such high fields, super-conductivity needs to be reached, thus the coil is cooled using liquid helium. The B_0 field is spatially uniform and temporally stable in an ideal scanner, such that the same polarization is applied to all atoms. In practice this field is not perfectly homogeneous, such that magnetic field probes are used to measure the field and shim cards are used to correct these inhomogenieties. This is called passive shimming. Alternatively, active shimming coils can be used to smooth the inhomogenieties. From the perspective of a human subject the poles are oriented towards the head and the feet of the subject. We call this axis the z-axis.

Additionally, an **excitation system** is necessary to generate magnetic pulses $B_1(t)$ to cause a nutation as well as refocussing, spoiling, inversion and saturation in the region of interest. The $B_1(t)$ field is a radio frequency field with the excitation frequency defined by the B_0 field strength, Larmor frequency of the excitation target nuclei. For example with $B_0 = 3\text{T}$ we get

$$42.58\frac{\text{MHz}}{\text{T}} \cdot 3\text{T} = 127.74\text{MHz}.$$

Typical pulse durations are on the intervall $[0.1, 10]$ ms and amplitude about $< 25\mu\text{T}$. The shape of the excitation pulse is governed by an envelope function in terms of amplitude trajectory. The excitation pulses are perpendicular to the B_0 field, which is important to enable the excitation pulses to tip the spin system in the presence of the strong polarizing field. While the B_0 field causes polarization, thus alignment and precession, the $B_1(t)$ pulses cause a "tipping" or nutation of the precession — shown in 2.2, generating transverse magnetization such that it is detectable using Faraday's law of induction. An example of an excitation pulse is

$$B_1(t) = B_1^e(t)\left(\cos\left(\omega_{RF}t + \theta\right) i - \sin\left(\omega_{RF}t + \theta\right) j\right),$$

with $B_1^e(t)$ a pulse envelope function, e.g. `sinc` or `rect` functions, ω_{RF} the excitation carrier frequency which should match the Larmor frequency of the spin system, and i, j the polarization directions. Additionally a RF pulse has a flip angle α which is the angle between the

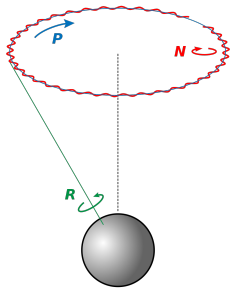

Figure 2.2. *A visualization of rotation r (in our case we call it spin) in green, precession p in blue — at the Larmor frequency — and nutation n in red [19].*

z-axis and the B_1 pulse (called inclination in spherical coordinates) and a phase θ which is the angle in the xy plane, starting at the x-axis (azimuth in spherical coordinates). The excitation system is commonly implemented using a birdcage coil as it is highly efficient in terms of energy, excitation and heating mitigation. The $B_1(t)$ field is supposed to be highly uniform, especially radially while decaying slightly axially.

One or mutiple **Receiver coils** read out the response of the excited tissue. This can be the same RF coil that is used for excitation or a set of dedicated coils. As the precessing magnetization causes induction in the receiving coil(s), the flux in the receiving coil changes causing an electromotive force

$$\epsilon = -\frac{\partial \Phi}{\partial t}.$$

The picked up signal is further split up, multiplied by $\cos \omega_0 t$ and $\sin \omega_0 t$ respectively and low-pass filtered to get the in-phase signal $I(t)$ and the quadrature signal $Q(t)$. These signals are then quantized, yielding $I(t)$ as the real and $Q(t)$ as the imaginary part of the signal.

The excitation forces the magnetization off from its equilibrium state and thus induces relaxation back to the equilibrium again after excitation. This relaxation can be separated into two components: longitudinal and transverse relaxation.

The longitudinal component can be described as a return to the equilibrium state of the magnetization on the z-axis

$$\frac{dM_z}{dt} = -\frac{M_z - M_e}{T_1},$$

where M_e is the equilibrium nuclear magnetization, which is proportional to the applied magnetic field B_0 [58]. The solution to this equation is

$$M_z = M_e + (M_z(0) - M_e)e^{-t/T_1}.$$

T_1 is the longitudinal relaxation time or spin-lattice time. It is proportional to the applied field strength and thus lengthens when B_0 is increased in strength. Practically, it is the time that it takes for M_z to return to 63% of its thermal eqilibrium state M_e [8, 50].

The transversal component can be formulated as a decay process

$$\frac{dM_{xy}}{dt} = -\frac{M_{xy}}{T_2}.$$

Accordingly, T_2 is the transverse relaxation time, also called spin-spin relaxation time. Solving the above equation gives

$$M_{xy} = M_e e^{-t/T_2}.$$

It is less dependent on the B_0 field strength and $T_2 \leq T_1$ holds always. T_2 is more rapid, the less mobile the spins are, i.e. T_2 is larger in water than in solids. Practically, in one T_2, M_{xy} loses

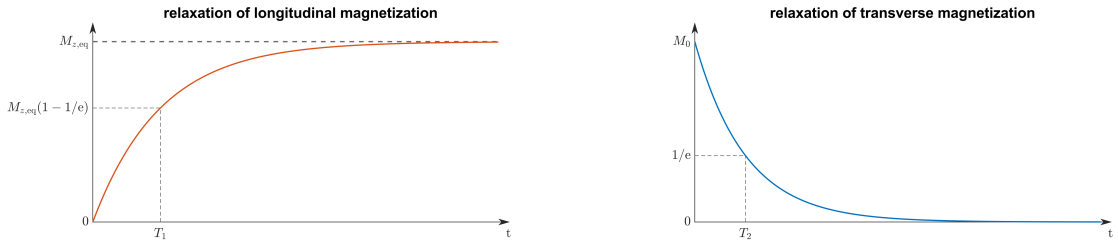

Figure 2.3. ***Left:*** *T_1 relaxation curve shows the return to the equilibrium state [69].* ***Right:*** *T_2 relaxation curve visualizes the transveral magnetization decay to zero [70].*

Tissue	T_1 [ms]	T_2 [ms]
Grey Matter (GM)	950	100
White Matter (WM)	600	60
Muscle	900	50
Cerebrospinal Fluid (CSP)	4500	2200
Fat	250	60
Blood	~ 1400	$\sim 180 - 250$

Table 2.1. *Longitudinal relaxation T_1 and transversal relaxation T_2 times for typical tissue types at 1.5 T main magnetic field strength [64].*

63% of its magnetization right after excitation. Typical T_1 and T_2 values for common tissue types at 1.5 T are shown in table 2.1.

In order to ease the description of the following concepts, the notions of **the lab and the rotating frame** will be introduced now. The laboratory frame is the frame of reference from the view in which the room or the scanner is anchored. The coordinate system is defined based on the B_0 field, where the z-axis is in the direction of the B_0 field, i.e. from the feet of the subject in the scanner to its head. The x-axis is parallel to the floor, while the y-axis is parallel to the walls. Put differently, y corresponds to the height, x to the width and z to the depth of the room. In this frame, we can observe rotation/spin, precession and rotation. In contrast, the rotating frame, the rotational/spinning and the precessional behavior is factored out, such that only the nutational information is preserved. This can be achieved with a transformation of the xy-plane, while the z-axis stays the same as in the lab frame. Intuitively, the xy plane in the rotational frame is rotating at frequency ω relative to the xy-axis of the lab frame. This is depicted in 2.4.

The equation of motion for a system of spins in the lab frame can be described by the Bloch equation, which consists of the precession & nutation, the transversal and the longitudinal relaxation terms:

$$\frac{dM}{dt} = M \times \gamma B - \frac{M_x i + M_y j}{T_2} - \frac{(M_z - M_e)k}{T_1}.$$

B aggregates the static B_0 field, the excitation $B_1(t)$ field and the gradient $G(t)$ fields.

Let $M_R = \left(M_{x'}, M_{y'}, M_z\right)^T$, $B_R = \left(B_{x'}, B_{y'}, B_z\right)$, R_z the rotation matrix around the z-axis, $M = R_z(\omega t)M_R$ and $B = R_z(\omega t)B_R$. When defining the tranverse magnetization as complex $M_R(t) = M_{x'}(t) + M_{y'}(t)$ for the rotating frame and $M(t) = M_x(t) + iM_y(t)$ we can relate the lab and the rotating frame via

$$M(t) = M_R(t) \exp -i\omega t$$

and with ω the Larmor frequency, $M_{x'}$ and $M_{y'}$ become constants without excitation pulses.

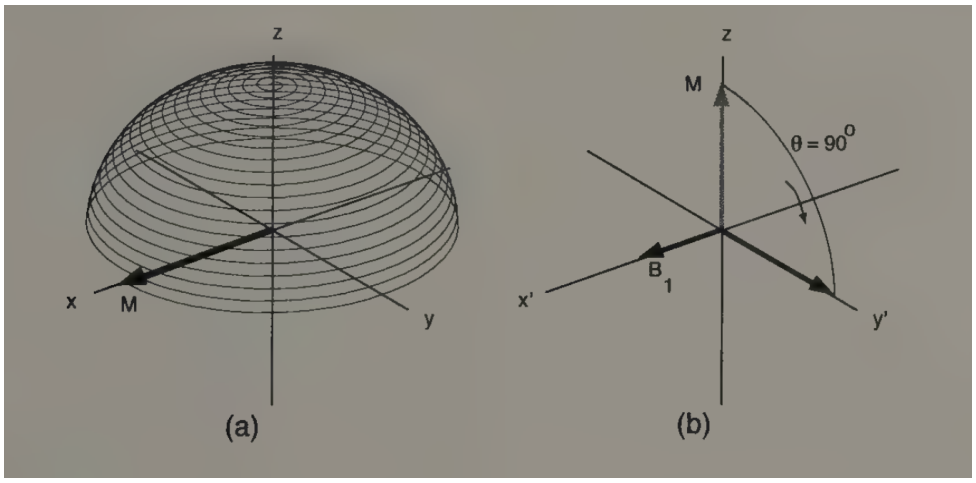

Figure 2.4. ***(a)*** *An excitation pulse causes the magnetization towards the transverse plane in the laboratory frame.* ***(b)*** *The same phenomenon in the rotating frame. Note how the spiral with the z-axis as center is transformed into a geodesic [58].*

With this we can reformulate the Bloch equation in the rotating frame

$$\frac{dM}{dt} = M_R \times \gamma B_{\text{eff}} - \frac{M_{x'}i + M_{y'}j}{T_2} - \frac{(M_z - M_e)k}{T_1},$$

where

$$B_{\text{eff}} = \frac{\omega_R}{\gamma} + B_R$$

and $\omega_R = (0, 0, -\omega)$.

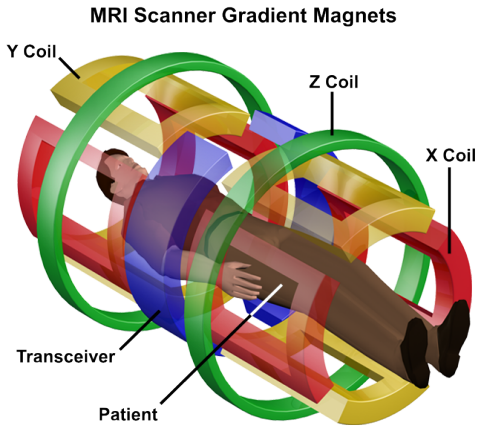

Figure 2.5. *A conceptual visualization of the gradient coils of an MRI scanner [48].*

A **Gradient system** is used to select regions of interest from the full organism and to iterate over different regions.
In figure 2.5 the coil for gradients in the z-direction is shown in green, which spans the head to the feed of the patient. The x gradient coil is shown in red and spans left to right of the patient. Lastly, the yellow part shows the coil to generate gradients in the y direction which spans back to chest of the patient. These coils are opperated at a very high current and voltage and thus requires internal water cooling.

Spatial information is encoded in three ways:

- slice selection

- phase encoding, and
- frequency encoding.

The gradient coils do also serve to cause spin de-, re- and pre-phasing to minimize artifacts, after slice selection and before readout respectively. Finally, the images can be sensitized or de-sensitized to motion, which will be used in DWI [3].

The gradient fields generated are usually measured in mT and depend on the size of the field of view. The literature states $50-100$ mT / m [3]. Further, the gradient fields are parallel to B_0 and thus only add or subtract to B_0 in the z-direction. They vary spatially in a linear manner, depending on the slew rate S_R [T/m/s].

The z-gradient G_z is typically generated by a Maxwell pair coil operated with equal-and-opposite currents, causing a B_0 field variation in the z-direction while being highly unifrom in the transverse plane. For both the x-gradient G_x and the y-gradient G_y Golay pair coils which in turn cause a linear variation in the B_0 field along the x and y direction respectively, while being highly uniform in the yz and xz plane respectively [9]. These gradients cause an increase or decrease in spin precession in the rotating frame, causing an isochromat to emerge, i.e. a plane where the spins precess at the same frequency, as well as faster and slower precessing spins above and below the isochromat.

Mathematically, the gradient fields can be described as inhomogeneous B fields whose z-component varies along the gradient direction. For the x gradient field for example, with G_x the gradient amplitude and x the position relative to isocenter, we get

$$B_{G,z}(x) = G_x \cdot x.$$

As we have three gradients along the respective axes, and with $\overrightarrow{r} = (x, y, z)$ the offset from the isocenter, for the gradient field we have

$$B_{G,z}(\overrightarrow{r}, t) = (G_x(t) \cdot x + G_y(t) \cdot y + G_z(t) \cdot z)k = (\overrightarrow{G}(t) \cdot \overrightarrow{r})k$$

Thus, overall we get the following expression for the magnetic field

$$\overrightarrow{B} = B_0 k + (\overrightarrow{G}(t) \cdot \overrightarrow{r})k + B_1^e(t) \left(\cos\left(\omega_{RF} t + \theta\right) i - \sin\left(\omega_{RF} t + \theta\right) j\right)$$

Now let us revisit the Larmor frequency in the rotating frame, i.e. where the frame of reference rotates at frequency $\omega = \gamma B_0$ around the z-axis

$$\omega = \gamma \overrightarrow{B} \Rightarrow \gamma \overrightarrow{G}(t) \cdot \overrightarrow{r} = \omega(\overrightarrow{r}, t).$$

First, this means that the Larmor frequency is now dependent on the position of the region of interest (ROI) and the time and we can control it locally using the gradient coils. Second, depending on how strong and how long the gradient pulse sequence is turned on, the more phase offset we get, i.e.

$$\phi_G = \int_0^t \overrightarrow{G}(\tau) \cdot \overrightarrow{r}(\tau) d\tau.$$

If we assume that the phase offset is caused by a rectangular waveform in one dimension, for example x we have

$$\phi_G(x, t) = \gamma G_x \cdot x \cdot t.$$

To select a slice, a spatial gradient of frequency of constant magnitude in one dimension, an RF pulse with frequencies matched to the slice and a re-phasing gradient are reuquired. The former two actually select the slice, while the latter re-phases the spins within a slice to increase the signal-to-noise ratio (SNR). The selected slice is in the plane that is perpendicular to the

direction of the applied gradient and its thickness depends on the strength of the gradient and the excitation bandwith, i.e. the step size with respect to ω_{RF}.

With that and using Farraday's law of induction we get the MRI signal equation

$$S(\overrightarrow{k}) = \int M_{xy}(\overrightarrow{r}, 0) e^{-i2\pi \overrightarrow{k}\overrightarrow{r}} d\overrightarrow{k}$$

where M_{xy} is the transverse magnetization of the object in the scanner and $e^{-i2\pi \overrightarrow{k}\overrightarrow{r}}$ is the encoding term rastering through k-space. The acquired data has units of inverse length, i.e. spatial frequency. Thus, the Fourier transform of the acquired data is an image of the object under inversigation.

Generally, k-space can be defined as

$$\overrightarrow{k}(\tau) = \frac{\gamma}{2\pi} \int_0^\tau \overrightarrow{G}(t) dt$$

So far we discussed slice selection. However, that is only one part of the encoding term and does not explain, how data is acquired in k-space. In order to sample data in k-space, we need to encode two more dimension which is done by phase and frequency encoding.

Phase encoding occurs after excitation and before readout. This is done by applying a gradient pulse with a different magnitude each repetition of the acqusition loop. Usually one phase encoding step is applied per echo. In terms of k-space it is an offset from the origin to higher frequency regions depending on the pulse length and strength of the pulse. The longer or the stronger the pulse, the further the offset from the origin in k-space. This is depicted in 2.6: G_y corresponds to the phase encoding gradient in (a). In (b) we can see how two different phase encoding amplitudes generate a different offset (the arrows pointing to the top left). It can be applied either in one direction for 2D imaging or in two directions for 3D imaging and can be combined with other pulses, like for slice-selection or rephasing. The resolution depends on the number of phase encoding steps per slice, as well as the duration of the gradient pulse and its amplitude.

Frequency encoding is mutually exlusive with other simultaneous pulses. It's also called the readout gradient and usually consists of two steps for gradient echo sequences: the pre-phasing pulse is used to dephase the spins such that the readout rephases the spins and the peak amplitude is reached at echo time. The actual readout consists of a constant magnitude gradient pulse and add linear spatial variation of the precession frequencies in the slice. Spin-echo sequences usually do not have a pre-phasing pulse. In 2.6 we can see that G_x is the frequency encoding gradient, consisting of the pre-phasing and the readout. The latter enables the acquisition across a line in k-space (horizontal arrows).

Finally, Computers control the different coils, store the measurements and reconstruct images from the measured magnetizations as shown schematically in 2.7.

Overall the components of MRI scanners are not as ideal as presented here. Both B_0 and B_1 may posses inhomogenieties, while gradient coils act rather non-linear than linear. Gradients induce currents — called eddy currents — in nearby conducitve structures. Concomitant fields are another example of noise that emerges in real world hardware. Most of these noise sources are addressed by either hardware or software, while mitigating some is still a field of active research.

Using the excitation and the gradient system now allows to construct different **excitation sequences**. The two most basic types are gradient echo and spin echo sequences. An example

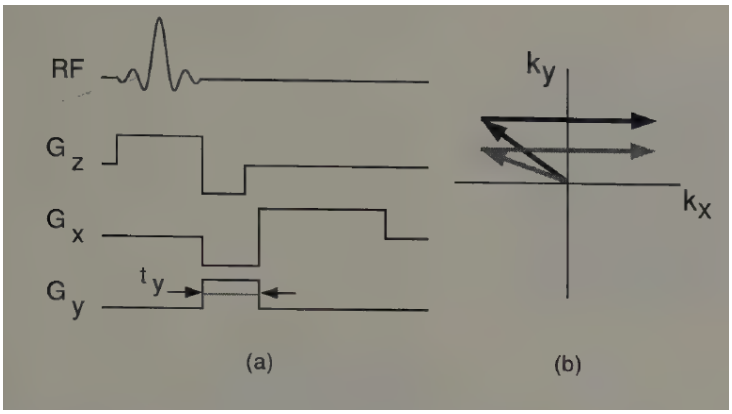

Figure 2.6. *Sampling in k-space is done using phase (G_y) and frequency (G_x) encoding. In (a) a timing diagram of a gradient echo sequence is depicted. The corresponding k-space sampling trajectory is shown in (b). The horizontal arrows correspond to the frequency encoding of the readout, while the more vertical arrows correspond to the offset by the phase encoding gradient [58].*

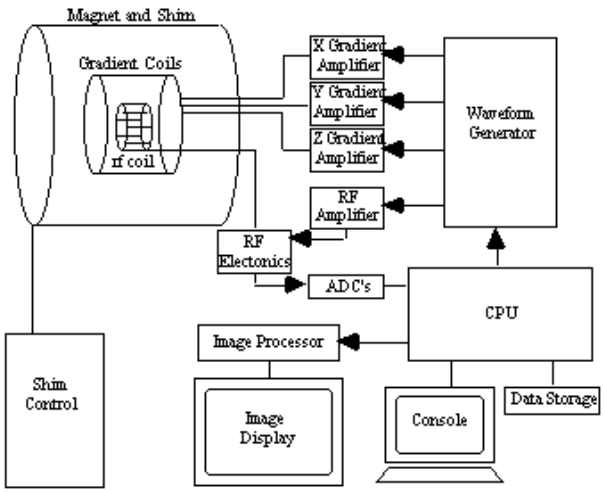

Figure 2.7. *Block diagram of the components in an MRI scanner system [14].*

of a gradient echo sequence is shown in 2.8. It uses the frequency-encoding gradient to refocus the phase of the spins in the region of interest. Gradient echo sequences have shorter repetition times T_R enabling faster imaging. After the echo, some transverse magnetization remains, leaving options to generate different image contrasts, as shown in 2.8.

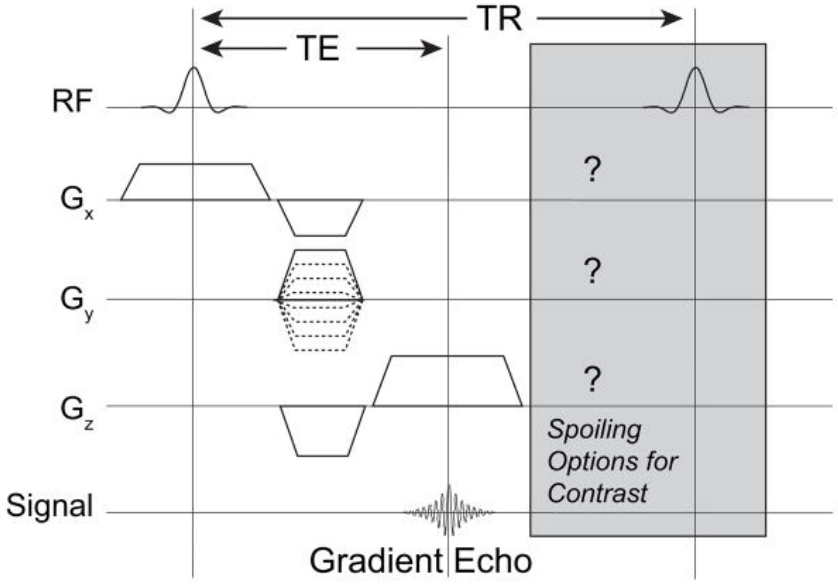

Figure 2.8. *Timing diagram of a gradient echo sequence [30]*

On the other hand there are spin echo sequences, which re-phase by initially applying a 90° pulse followed by 180° pulses per T_R rephasing pulse that flips the phase causing the dephasing factors (for example inhomogenieties) to rephase the spins. An example is shown in 2.9. Notice, that the readout gradient does not have a pre-phaser and that there is an initial 90° pulse followed by a 180° flipping pulse as just mentioned. Spin echo sequences can be used for

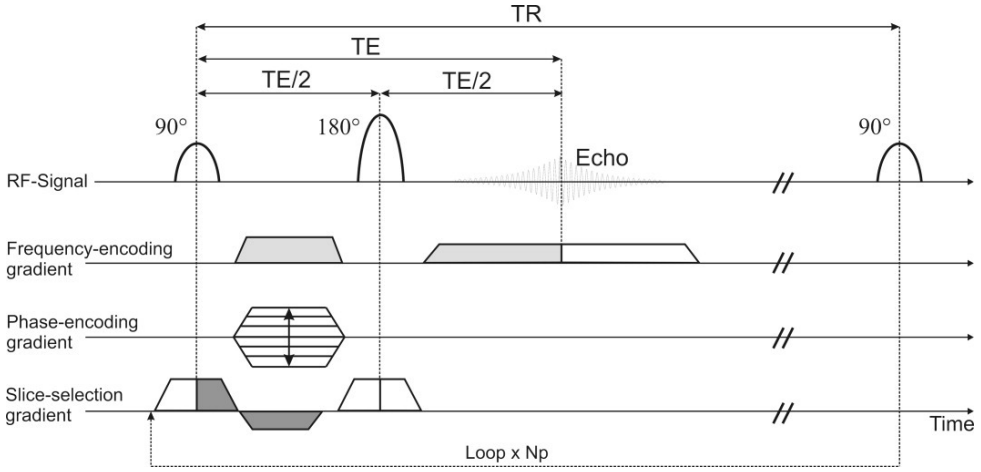

Figure 2.9. *Timing diagram of a spin echo sequence [43].*

generating a wide range of contrasts depending on the echo time T_E and the repetition time T_R. If both T_E and T_R are short, there is incomplete recovery of the transverse magnetization and the image will be T_1-weighted. When both are long, the signal can recover fully yielding a T_2-weighted image.

2.1.2. Balanced Steady-State Free Precession Imaging

Balanced steady-state free precession is a gradient echo sequence with short T_R, thus it enables rapid 2D or 3D imaging. It is fully balanced in the sense that in one T_R the gradient moments will integrate to zero on all three axes. A timing diagram for the sequence is shown in 2.10. A steady state is reached after a sequence of initial pulses, that are not used for imaging but

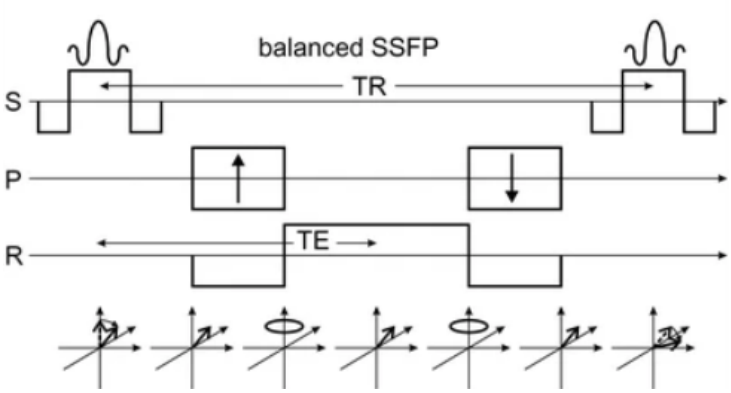

Figure 2.10. *Timing diagram of a bSSFP sequence. All gradients integrate to zero per axis [75].*

merely to reach the steady state of the spin system in the ROI. Neglecting off-resonance effects, the magnetization is repeatedly tipped back and forth around the z-axis. The magnitude of the magnetization stays relatively constant, once the steady state is reached. However, the magnetization moves on the surface of an elipsoid with the RF flipping between two planes that are an angle α appart. This ellipsoid has an eccentricity of $\sqrt{T_2/T_1}$, meaning that the contrast of the resulting image is T_2/T_1 weighted.
When acquiring an image using bSSFP, there are off-resonance effects. The flip angle α is

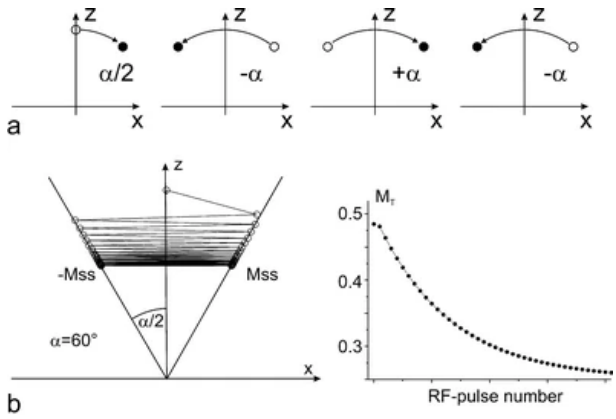

Figure 2.11. *(a) Tipping back and forth of the magnetization around the z-axis without dampening. (b) The same as a with relaxation included leading to an oscillating steady state [75].*

constant, when disregarding off-resonance effects. Taking the off-resonance effects into account, the effective flip angle β emerges, that is always larger or equal to α. The dephasing angle within one T_R θ, the flip angle α and the effective flip angle β are related by

$$\tan(\alpha/2) = \tan(\beta/2)\cos(\theta/2)$$

Intuitively dephasing due to field inhomogenieties — among other factors — leads to a deviation from the on-resonance condition where the rotation around the z-axis is spherical. This rotation is described as a rotation by θ around the z-axis, causes that the excitation pulses have a different effect on the magnetization, disturbing the spherical oscilation of the magnetization around

the z-axis. On the one handside, this causes the magetization to be aligned along the x-axis, providing a refocussing mechanism like in spin echo sequences. On the other handside, the disturbance of the spheriocal oscillation causes dephasing. The signal is attenuated proportionally to the amount of dephasing. When the effective flip angle reaches 180°, the signal vanishes completely, while when there is no disturbance of the spherical oscillation, the signal intensity is maximal. These off-resonance effects give rise to the characteristic signal-(off-resonance) frequency response profile of bSSFP images, as well as banding artifacts due to the signal drop at $\theta = \pm\pi$ [75].

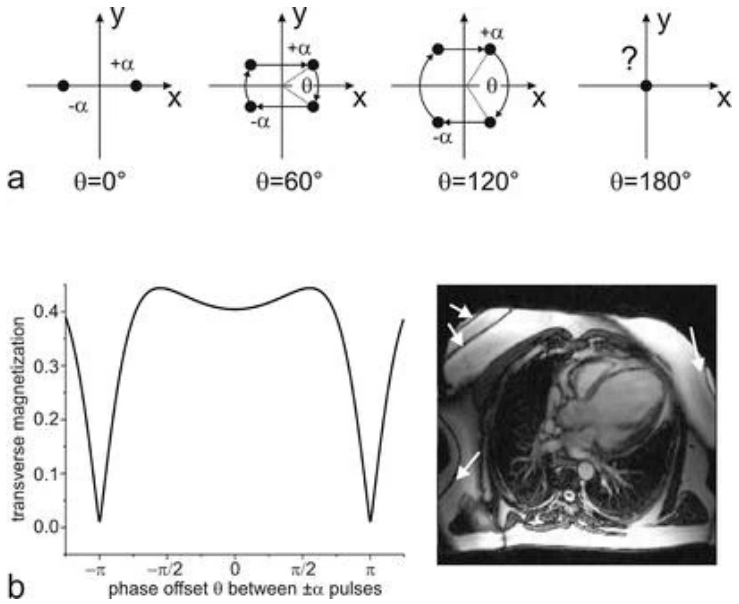

Figure 2.12. *(a) Position of the magnetization vector in the xy plane depending on the dephasing θ. (b) Signal magnitude as a function of the dephasing angle θ [75].*

Finally, the complex steady state magnetization directly after the RF pulse is applied is given by

$$M(t=0) = M_x(t=0) + iM_y(t=0) = -\frac{i}{D}(1-E_1)\sin(\alpha)(1-E_2e^{-i\theta})$$

$$D = (1-E_1\cos(\alpha))(1-E_2\cos(\theta)) - (E_1-\cos(\alpha))(E_2-\cos\theta)$$

with $E_i = e^{-T_R/T_i}$. The Magnetization at time t after an RF pulse is

$$M(t) = M(t)e^{-t/T_2}e^{i\phi(t)}$$

with $\phi(t) = t/T_R \cdot \Phi$ where Φ is the off-resonance related phase accumulated during each T_R [57, 21].

2.1.3. Diffusion-weighted Imaging

The Bloch equations can be extended to include diffusion effects using Fick's law of diffusion. This is called the Bloch-Torrey equation [84]:

$$\frac{dM}{dt} = \overrightarrow{M}_R \times \gamma \overrightarrow{B}_{\text{eff}} - \frac{M_x i + M_y j}{T_2} - \frac{(M_z - M_e)k}{T_1} + D\nabla^2\overrightarrow{M},$$

where

$$\overrightarrow{B}_{\text{eff}} = \frac{\omega_R}{\gamma} + \overrightarrow{B}_R,$$

$\omega_R = (0, 0, -\omega)$, and D is called the diffusion tensor.

Spins are thermodynamically driven to exchange positions. This process is irreversible and can lead to signal attenuation. Diffusion is modelled as a random walk by Fick's law. Adapting this to the diffusion of magnetization in one dimension, there is a probability of the spin moving left or right with each having the probability of 0.5 in the 1D case, 0.25 in the 2D case and $0.1\overline{6}$ in 3D. This gives rise to a Gaussian probability distribution for all spins for the ğD case of

$$P(\overline{r}, t) = \frac{1}{\sqrt{4\pi Dt}} \exp - \frac{\overrightarrow{r}^2}{4Dt}$$

In the case of free or isotropic diffusion the diffusion in all directions is similarly strong and will lead to a similar diffusion coefficient D. Contrary, if the diffusion is aniostropic or restricted, the probabilities for moving in a certain direction are dependent on the direction. This means the diffusion coefficients D are equal or similar for the isotropic case. For example in an axon, particles can move along the axon (in the axial direction) rather freely while being highly restricted in the radial directions. In this case the diffusion coefficients D are different from each other.

The phase induced by a gradient can be formulated as

$$\phi(t) = \gamma \int_0^t \overrightarrow{G}(\tau) \overrightarrow{r}(\tau) d\tau$$

where $\overrightarrow{r}(\tau)$ is the history of the spin based on the random walk formulated above. For stationary spins, phase is added by the positive part of the bipolar gradient pulse, while the negative part undoes that phase addition. For diffusing spins doing the random walk, phase is accumulated to a non-zero value. This transfers to systems of spins. So the larger the phase spread, the higher the diffusion. The longer and stronger the gradients are, the higher the sensitivity to diffusion.

The diffusion-weighted MRI (DWI) signal can be formulated as

$$S = S_0 e^{-b \overrightarrow{g}_j \cdot D \overrightarrow{g}_j},$$

with S the DW signal, S_0 the signal without diffusion effects, D the diffusion tensor (DT), g_j the diffusion encoding gradient unit vector, and b the b value.
The b value is a free parameter which is to be set by the exeprimenter. As e^{-bD} is a dampening factor, diffusion always attenuates the signal amplitude. Specifically, b has units of $[mm^2/s]$, is usually on the magnitude of $[100, 3000]$, and controls the signal attenuation induced by the diffusion. Thus, a high b value leads to high diffusion sensitivity but low SNR, while a low b value causes low diffusion sensitivity with a high SNR. For example a b value of 1000 causes 95% signal attenuation for freely diffusing water. As in biological tissue the diffusion coefficient is smaller than that of free moving water, the signal attenuation of water can be used as a lower bound. Diffusion coefficients for typical brain tissue types are shown in table 2.2.

Tissue	Diffusion Coefficient [10^{-6} mm^2 / s]
WM	$670 - 800$
Cortical GM	$800 - 1000$
Deep GM	$700 - 850$
CSF	$3000 - 3400$

Table 2.2. *Diffusion coefficients for typical brain tissues [77, 35].*

The gradient pulse that induces the diffusion weighting needs to be designed in such a way, that it does not add phase to stationary particles. This is the case for bipolar gradients for

example. Duration and amplitude of the pulse depend on the b value by

$$b = \frac{2}{3}\gamma^2 G^2 \delta^3,$$

for gradient echo diffusion weighted imaging and by

$$b = \gamma^2 G^2 \delta^2 (\Delta - \frac{\delta}{3})$$

for spin echo diffusion weighted imaging, where G is the gradient amplitude, δ is the pulse duration and Δ the duration between the gradient pulse beginning and the refocussing pulse.

When applying a bipolar pulse to stationary spins, the phase of the spins depends on their position on the positive lobe, while undoing the phase addition on the negative lobe. Doing the same with diffusing spins, the positive lobe causes a phase addition and slight attenuation while due to the movement of spins, the phase cannot be refocussed and the signal further attentuates strongly.

For spin echo sequences, the positive lobe is applied before the refocussing pulse, then there is a slight delay, followed by another positive pulse (as the refocussing pulse inverts the spins). As we want maximal diffusion weighting and short sequences, the gradient amplitude is fixed at its maximum value G_{max} such that δ and Δ are the left over free variables and chosen in trade-off with b.

The diffusion weighting gradients can be applied using each gradient coil either separately or a linear combination of them. In order to obtain diffusion information in all spatial directions, different linear combinations of gradient coils applying the DW gradient pulse are needed. The experimental setup to acquire diffusion tensor images (i.e. gathering diffusion coefficients in all directions) is the following:

1. Set $b = 0$, measure S_0.
2. Set $b \neq 0$. For $j \in \{0, \ldots, N_j \in \mathbb{N}_{\not\vdash}\}$, measure S_j.
3. Calculate D and it's Eigenspace to extract the diffusion coefficients.

A high singal loss induces a high diffusion coefficient and vice versa. From the Eigenspace of the diffusion tensor per voxel, different scalar measures can be calculated. Let $\lambda_0, \lambda_1, \lambda_2$ be the Eigenvalues of the diffusion tensor for a fixed but arbitrary voxel and v_0, v_1, v_2 the Eigenvectors. Let the Eigenvalues and Eigenvectors be sorted by Eigenvalue ascending. The axial diffusivity is simply the value of the largest eigenvalue

$$\mathrm{AD} = \lambda_2,$$

the radial diffusivity is the mean of the smallest two eigenvalues

$$\mathrm{RD} = \frac{\lambda_0 + \lambda_1}{2},$$

the mean diffusivity is the mean of the Eigenvalues

$$\mathrm{MD} = \frac{\sum_{i=0}^{2} \lambda_i}{3},$$

the fractional anisotropy is the fraction of the variance of the eigenvalues normalized by the root sum squared of the eigenvalues weighed by a factor

$$\mathrm{FA} = \sqrt{\frac{3}{2}} \frac{\sqrt{\sum_{i=0}^{2} (\lambda_i - \overline{\lambda})}}{\sqrt{\sum_{i=0}^{2} \lambda_i^2}},$$

with $\overline{\lambda}$ the mean of the Eigenvalues. Finally, the direction of the diffusion can be calculated by means of azimuth and inclination in polar coordinates where arctan is used in the sense of `atan2` correcting for quadrants

$$\text{Azimuth} = \arctan\left(\frac{v_2[1]}{v_2[0]}\right)$$

$$\text{Inclination} = \arccos\left(\frac{v_2[2]}{\sqrt{\sum_{i=0}^{2} v_2[i]^2}}\right)$$

and an RGB image of the major direction of the diffusion can be generated by

$$\text{RGB}_i = \text{FA} \cdot v_2[i].$$

2.2. Deep Learning

Machine learning (ML) gathered a lot of attention in the last decade. Specifically, neural networks (NN) were a main focus of ML research [90, 27, 46]. Deep learning (DL) is a sub-area of neural network research, where many layers are stacked on top of each other, such that the network is considered "deep". In this section, neural networks as well as their building blocks will be described shortly. Common neural network architectures used in practice will be introduced as well as the algorithm to train those networks. Finally, recent work in the fields of image-to-image generation and medical imaging will be presented.

2.2.1. Fundamentals of Neuronal Networks

A neural network is — in its most basic form — defined by several components:

- A set of topological constraints — also called the architecture — $\mathcal{T}$,
- a set of learnable parameters θ consisting of,
  - weight and bias parameters θ_W,
  - activation function parameters θ_A,
- additional intermediate functions like normalizations,
- hyper-parameters that are non-trainable and to be optimized by the user or a secondary optimization procesure like grid search like the learning rate ϵ, and
- a loss function $\mathcal{L}$.

The topological constraints define to which neurons the input is fed, which neurons will provide the output values, as well as how the information flows in the network or put differently, which operation is to be applied when.

In its simplest form a neural net consists of a single neuron with n inputs $x_0, \ldots, x_n$, weights and biases for each input $w_0, \ldots, w_n$ and $b_0, \ldots, b_n$, an activation function f_A, parameters to the activation function θ_A and outputs a single value with $n \in \mathbb{N}$ as shown in 2.13. This is called a **perceptron** or McCulloh-Pitts-neuron [73, 53]. The output is then computed by multiplying the inputs with the weights and adding the bias, summation of the resulting vector and the application of the activation function, as shown in 2.14.

$$f(\overrightarrow{x}; \theta) = f_A(\sum_{i=0}^{n-1}(x_i \cdot w_i) + b_i; \theta_A).$$

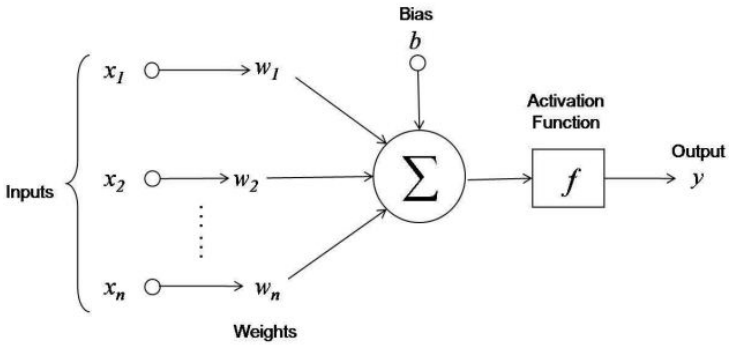

Figure 2.13. *Visualization of a perceptron*

As $\overrightarrow{w} \cdot \overrightarrow{x} + \overrightarrow{b} = (\overrightarrow{w}, \overrightarrow{b}) \cdot (\overrightarrow{x}, \overrightarrow{1})$, we can add the bias term as another weight variable and reformulate **Ask Flo**

$$f(\overrightarrow{x}; \theta) = f_A(\sum_{i=0}^{n}(x_i \cdot w_i); \theta_A).$$

In the following and in agreement with the basic definition of a NN, bias vectors will be treated as part of the weight matrix.
Perceptrons can be stacked on top of each other, but also next to each other. Stacking on

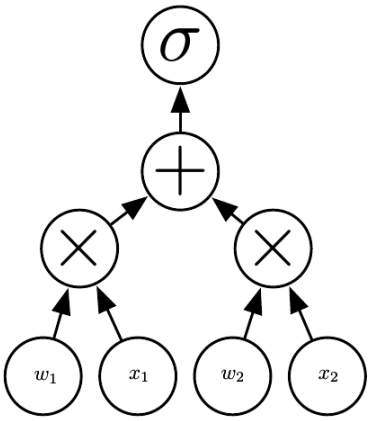

Figure 2.14. *The computational graph of a perceptron inference. σ is the activation function [26].*

top is straight forward: connect the output of the first peceptron to the input of the following one. Stacking horizontally is usually done using matrix operations and delaying the activation function application until the matrix multiplication of the inputs and the weights are finished. Note that this is only possible when using the same activation function across all neurons in such a horizontal stack. This horizontal stack is called a layer and can be extended to stack in 2D or 3D depending on the dimensionality of the input data.

In order to enable this construct to learn, two more components must be provided: the loss function and the optimization algorithm. The loss function is a function that quantifies the distance between a ground-truth and a prediction. Often metrics are used like the L_1 or L_2 norm, but generally, every function that maps an output of the NN to a real number is usable.

$$L_1(y, \hat{y}) = \frac{1}{N} \sum_{i=0}^{N-1} |\hat{y}_i - y_i|$$

The optimization algorithm minimizes the loss function in order to minimize the differences between the desired output and the actual output. The foundation for most optimization algorithms used in practice today is called gradient descent. It uses the gradient ∇ of a function to

find its minimum by iteratively adjusting the argument in the direction of the steepest descent of the derivative.

$$\theta^{i+1} = \theta^i - \epsilon \nabla \mathcal{L}(f(x; \theta^i), y),$$

where ϵ is a user specified parameter called the learning rate and y is the ground-truth. Notice that this introduces the constraint of differentiablility to all components in the neural network.

In a NN with multiple layers, the network can be viewed as a level-wise function composition:

$$f(\overrightarrow{x}; \theta) = f_l(W_l f_{l-1}(W_{l-1} \dots f_1(W_1 \overrightarrow{x})))$$

As we do not only have a single variable that is to be adapted but a set of variables, connected by a certain topology, back-propagation is needed. Back-propagation uses the chain rule to spread the changes within the network, computing the gradient of one functional unit at a time while avoiding to recompute intermediare terms in the chain rule [87]. Often multiple input samples are processed in one step and the loss function is averaged over the batch. Usually the weights (and biases) of a NN are initialized randomly or using some initialization procedure that has shown to be effective empirically, like the Xavier initialization [25].

Neural networks have been proven to be universal function approximators, i.e. can learn any function of arbitrary complexity, when given enough parameters, appropriate topology and a suitable loss function [81].
The two most common settings are classification and regression tasks. Classification takes an input and produces a discrete output — usually a class label — while regression outputs a continuous value.

Now commonly used layers, that are also used in this work will be introduced. We already saw the basic architecture of a **dense** layer: horizontally stacked perceptrons forms a dense layer. It is called dense as every input is potentially connected to all neurons (potentially as the weight can be zero after learning). This is then executed as a dense vector or matrix multiplication, again depending on the input dimensionality, followed by the application of the activation and other intermediate functions. Usually these layers are used as the final layer in a regression task, however for large inputs this is unfeasible as the layer would need

$$d_1 \cdot d_2 \cdot d_3 \cdot c$$

neurons, where d_i is the output dimensions and c is the number of channels with each neuron being connected to all neurons in the previous layer. For example with an output size of $(96, 128, 128)$ with 6 channels and $48 \cdot 64 \cdot 64 \cdot 24$ neurons in the previous layer, this would require ~ 22 trillion weights or $\sim 22 * 32B = 704$ TiB.

Convolutional layers are frequently used for image input data. Instead of connecting every input voxel (or pixel) to all neurons in the layer, convolution operates by learning a kernel of user defined size that is slided over the image. This keeps the amount of parameters low while still being able to process the full input. However this requires the multiplication of the input with the kernel multiple times, outputting one value per evaluation. The application of a convolution operation on a whole 2D input is shown in 2.15. The convolution operation can be defined mathematically for the 3D case for a single evaluation as

$$C(\overrightarrow{x}, i, j, k; \theta) = \sum_m \sum_n \sum_o x_{i+m, j+n, k+o} W^K_{m,n,o}$$

where W^K are the weights of the kernel. Deconvolution or transpose convolution works analogously, just that the input is slided over the zero-padded kernel. For details on the arithmetics of convolution and deconvolution operations including padding, strides and dilation, the reader is refered to an excellent review [17]

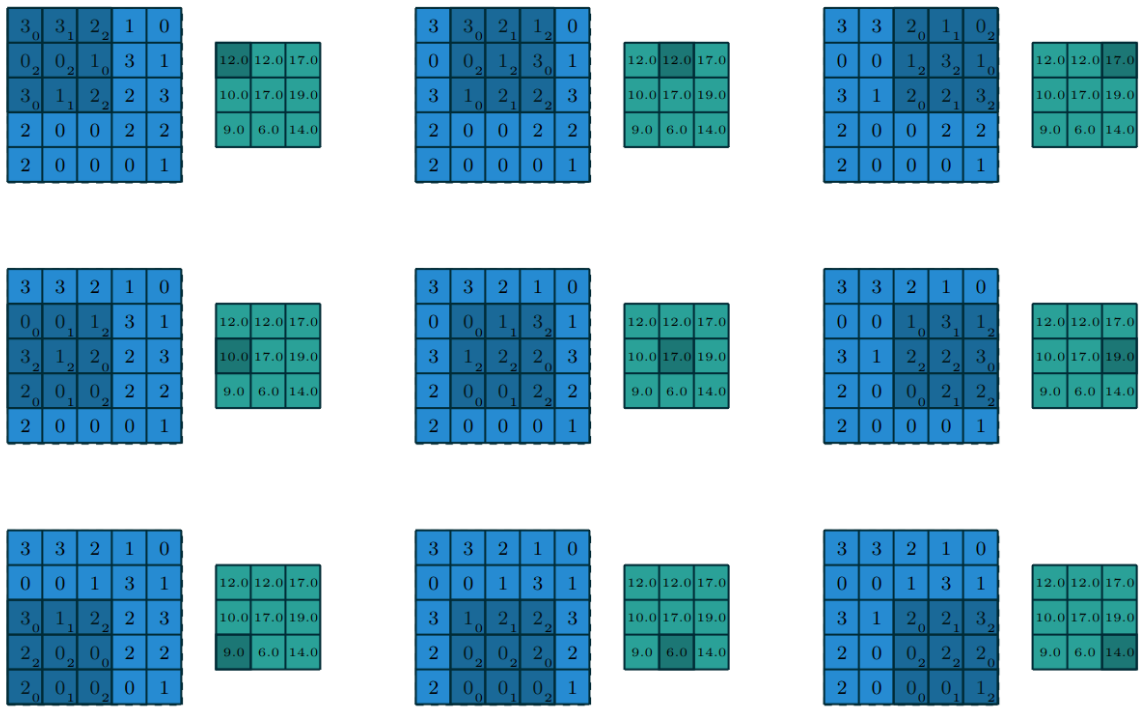

Figure 2.15. *Application of the convolution operation on a 2D input. For each element in the output, the convolution is applied once to the input image and then slided to the next position [17].*

Activation functions are the major souce of non-linearity in NNs. They are usually applied element-wise after the input-weight multiplication. Common examples of activation functions are the sigmoid function shown in 2.16 on the left handside

$$\sigma(x) = \frac{1}{1 + \exp(-x)},$$

the rectified linear unit (RELU) [56] shown in the middle in 2.16

$$g(x) = \max 0, x \ ,$$

or the parametrized rectified linear unit (PRELU) [34] shown on the right in 2.16

$$g(x, \theta) = \begin{cases} x & x > 0 \\ a^{\theta} x & x < 0 \end{cases}$$

where a is a trainable paramter and usually close to zero.

Normalization layers were introduced to make training more stable and to increase the speed of convergence [41]. The most commonly used normalization operations are shown in 2.17 and shortly described below.

- batch normalization [41]: computes the mean and variance of *a batch of inputs per channel* and normalizes the inputs to zero mean and a standard deviation of 1.
- Layer normalization [2]: computes the mean and variance of *a sample across all channels* and normalizes the inputs to zero mean and a standard deviation of 1
- Instance normalization [86]: computes the mean and variance of a *a sample across one channels* and normalizes the inputs to zero mean and a standard deviation of 1

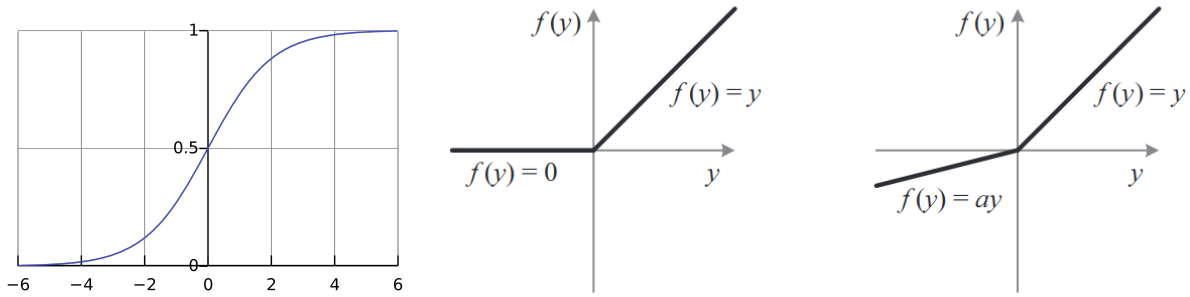

Figure 2.16. ***Left:*** *Sigmoid function.* ***Middle:*** *Rectified linear unit.* ***Right:*** *Rarametrized rectified linear unit [34]*

- Group normalization [95]: computes the mean and variance of a *a sample across a group of channels* and normalizes the inputs to zero mean and a standard deviation of 1

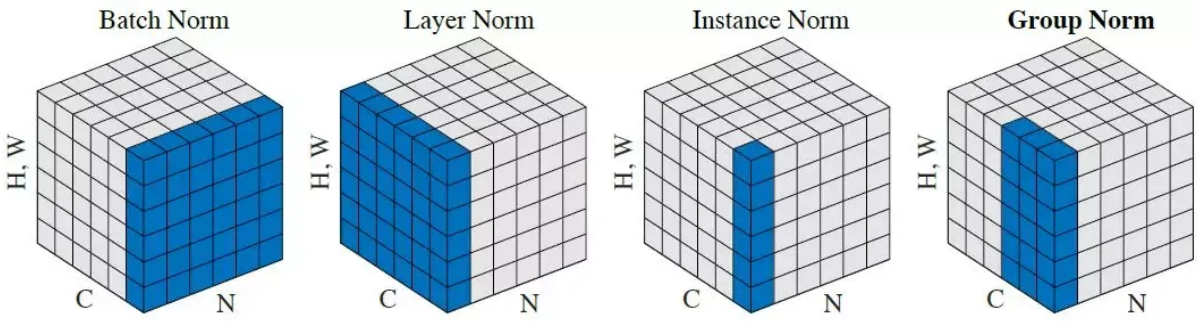

Figure 2.17. *Visualization of how the mean and variance is computed for each normalization operation that is used in practice.*

Drop-out layers are used to mask parts of the neurons in a layer in a probabilistic manner to decrease overfitting, i.e. to improve generalization [82]. Overfitting is the phenomenon that occurs when a machine learning model is trained for many epochs on the same training samples. When unseen samples are presented the network performs significantly worse than on the training data. A conceptual visualization of multiple dropout layers in a network is shown in 2.18. Drop-out also improve the robustness of the NN to artifacts & noise and serves as a passive augmentation technique, as when dropout occurs, it appears to the network as if the input has been masked randomly.

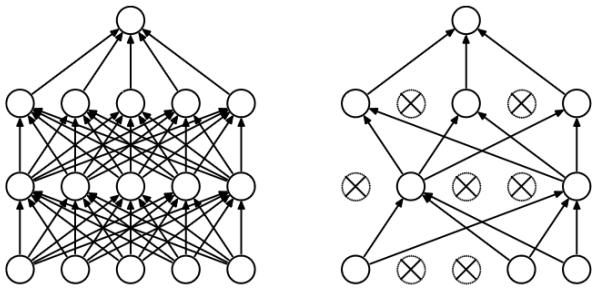

Figure 2.18. *Abstract visualization of dropout [82].*

Pooling layers are used to reduce the dimensionality between layers or blocks of layers. Reducing the size of the representation is essential in two ways: first, it reduces the computational costs in subsequent layers, while second it compresses the representation that was usually expanded to more channels. The second part is closely related to convolutions, in the sense of that convolutions usually slightly reduce the spatial size of the representation while expanding the number of channels. As the number of channels is usually increased by a factor per layer or block by a constant factor (e.g. 2), the spatial dimension only shrinks by a factor that is smaller. Pooling is also used to add spatial invariance, for example permutation invariance within the pixels that get pooled together [26]. To accomodate for that pooling is used. The two most common pooling operations used in practice are MaxPooling [99] and AveragePooling [24]. Max pooling groups a number of pixels together and returns the maximum per group. This is relatively sensitive to outliers but preserves strong signals in the network. Average pooling returns the average of a group of pixels. While it is not as sensitive to outliers, it may decrease the entropy of the signal. A visualization of both operations in shown in 2.19.

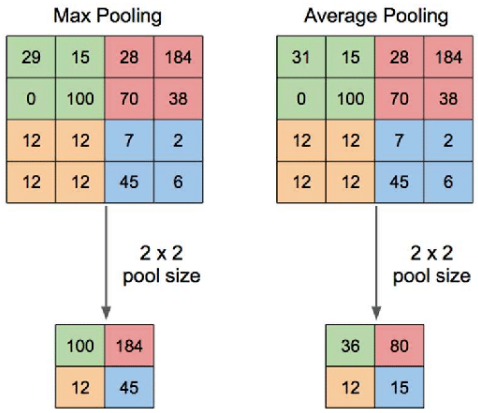

Figure 2.19. *Max pooling (left) and average pooling (right) visualized for single channel 2D input from the previous layer.*

Skip connections or residual connections were introduced by [33] in order to reduce convergence time and increase accuracy. Essentially, skip connections cause convolutional layers (in combination they are used very commonly) not to learn an unreferenced presentation but rather to extract the desired change in representation. This is done by "copying" the input to a layer or a block, then performing convolution, activation and normalization and finally adding the input to the layer to the output. Again, this encourages the convolution operation to only address the changes from the identity mapping that are beneficial to the task instead of learning to conserve the general representation of the input from the previous layer as well. Thus they can be seen as a kind of shortcut [33]. An illustration of a skip connection in the architecture of a block is shown in 2.20.

2.2.2. Architectures

Various deep learning architectures have been proposed in the past. The more successful ones are convolutional networks, generative adversarial networks, transformers and stable diffusion models [90, 27, 46, 80, 49]. Here we will focus on convolutional networks mostly.

A convolutional neural network uses convolutions, activation functions, pooling, and dropout as major layers in the network topology [26]. Modern architectures use the notion of a block which groups a number of these layers together into a unit and reuses these blocks with different input and output shapes, thus the number of total learnable parameters. An example is the

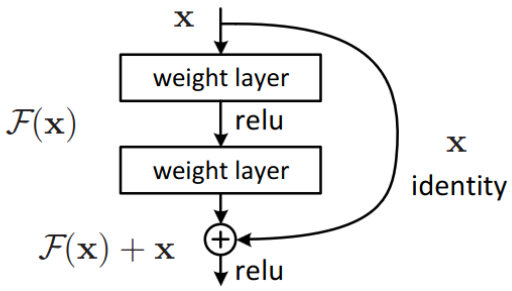

Figure 2.20. *Diagram of a block in a neural net that incorporates a skip connection [33]. Weight layers stand for convolution or dense layers for example.*

ResNet Block shown in 2.20 [33]. In this example, a block consists of a convolution, followed by an activation function, another convolution, the addition of the skip connection (i.e. the input to the block) and another activation function application. Usually dropout and normalization layers are also included.

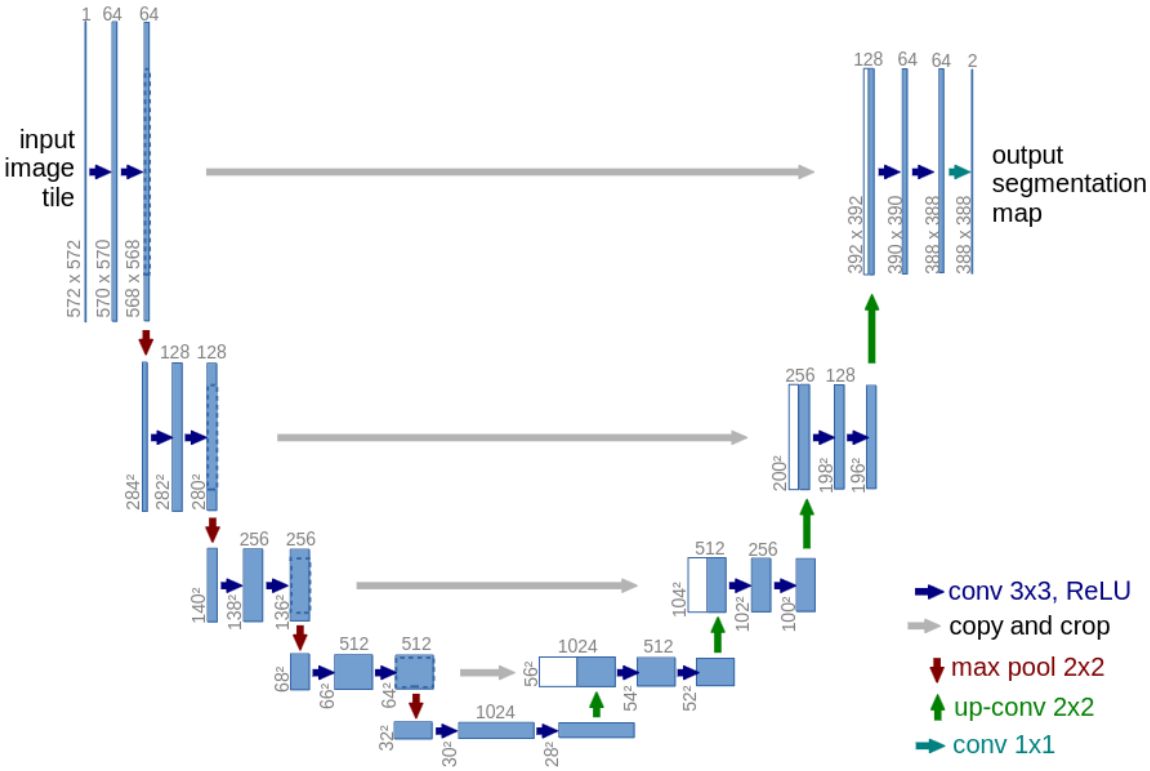

Figure 2.21. *The architecture of the UNet NN*

An extension of this method is the UNet architecture initially designed for the segmentation of cells in photomicrographs [72]. It is shown in 2.21. There is a contracting part that decreases the dimension of the input by a factor of approximately 2 per level, while doubling the number of features. As we can see, this network employs two convolutions and ReLU activations follwed by a max pooling operation per block in the contracting stage. The expanding path undoes the change in dimensionality at least partially, i.e. it increases the spatial size while decreasing the number of channels at each level. An expanding block first upsamples data by applying transpose

convolution (also called deconvolution) and then applies two convolution operations along with ReLU activations. Overall there are 4 of these blocks each on the contracting and on the expanding path. Blocks at corresponding positions between the paths have skip connections: the output of the first contracting block is concatenated to the expanding block after the upsampling. Finally, there is a final convolution with kernel size 1 to infer the output segmentation labels. This architecture has won the 2015 ISBI EM segmentation challenge [76] and is since then one of the goto models for many problems, as well as the foundation for other model types like diffusion models [71].

2.2.3. Training

Usually, the previously briefly introducted gradient descent method in combination with the back-propagation algorithm is employed to train the NN. As often not only a single data sample is processed but multiple together in a batch, the loss and the gradient update are performed on the mean of the loss of a batch. This renders the implementaion of the gradient descent method stochastic and is called stochastic gradient descent. Multiple extensions have been proposed, to increase the speed of convergence, to avoid getting stuck in local optima and to regularize the solution with additional constraints (e.g. on the absolute value of the weights) [74]. One such extension is called AdamW [51]. It is based on the the adaptive gradient-based optimization method Adam [44], which uses estimates of the first and second moment (mean and uncentered variance) of the gradient to include momentum to the parameter update. AdamW additionally introduces a decaying of the weights into the optimization procedure in order to increase sparsity into the model (reduce the number of calculations potentially in matrix multiplications) and to fight overfitting.

Given the architecture, the initialized weights and a dataset, the dataset is split into a training set and a test set that will not be used during training but rahter to compute prediction metrics on unseen data samples to evaluate generalization of the model. The algorithm is then executed and the weights are updated. Training on the training dataset once usually leads to underfitting, i.e. bad predictions as the weights have not been adapted enough. Thus, the training reiterates on the training set multiple times. A single iteration of the optimization algorithm on the weights and dataset is called an epoch.

Another way to target overfitting is early stopping. For that, a small part of the training set is split off to be used for validation at the end of each epoch, which computes the loss function on the validation set which is also unseen during training. If the validation loss does not further decrease (while the training loss might still), the training is stopped to achieve close-to-optimal generalizability [67, 23]. This procedure is visualized in 2.22

For tasks that prove to be difficult like question answering or image generation, pretraining has proven to be effective [65, 92]. An auxiliary task is used to train the model with an auxiliary dataset which can be orders of magnitude larger than the actual dataset for the task at hand. In a second step the model is then trained on the actual task dataset. Not only does this improve the prediction metrics of the final network, it also increases the speed of convergence. Early work on neural networks in the 1960s and 70s has additionally established the notion of transfer learning [6]. Here a network trained on another task and dataset that is related to the actual task is taken as initialization of the parameters for the actual task. During training, only the parameters of the output layer or block of the new network is adapted. Then, the learning rate is reduced by one or two orders of magnitude and either the layers are gradually unfrozen step by step or the full network is set to trainable. Many variations of this procedure have been developed in the past and are reviewed in [62] as well as extended in [15].

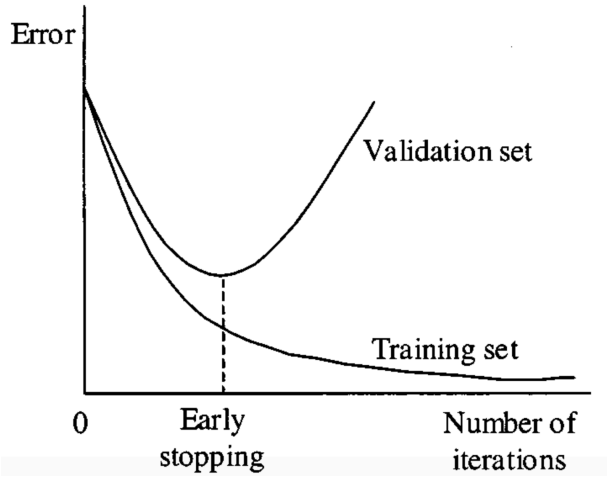

Figure 2.22. *The early stopping method visualized in terms of loss functions [23].*

2.2.4. Image-to-Image Translation

The task of image-to-image synthesis (or generation or translation) is to convert an input image from one domain to another. Examples include super-resolution, style transfer and image inpainting [59]. There is large variety of proposed NNs to solve this problem, as shown in 2.23. Most of these architectures use generative adverserial networks (GANs) and variational auto-

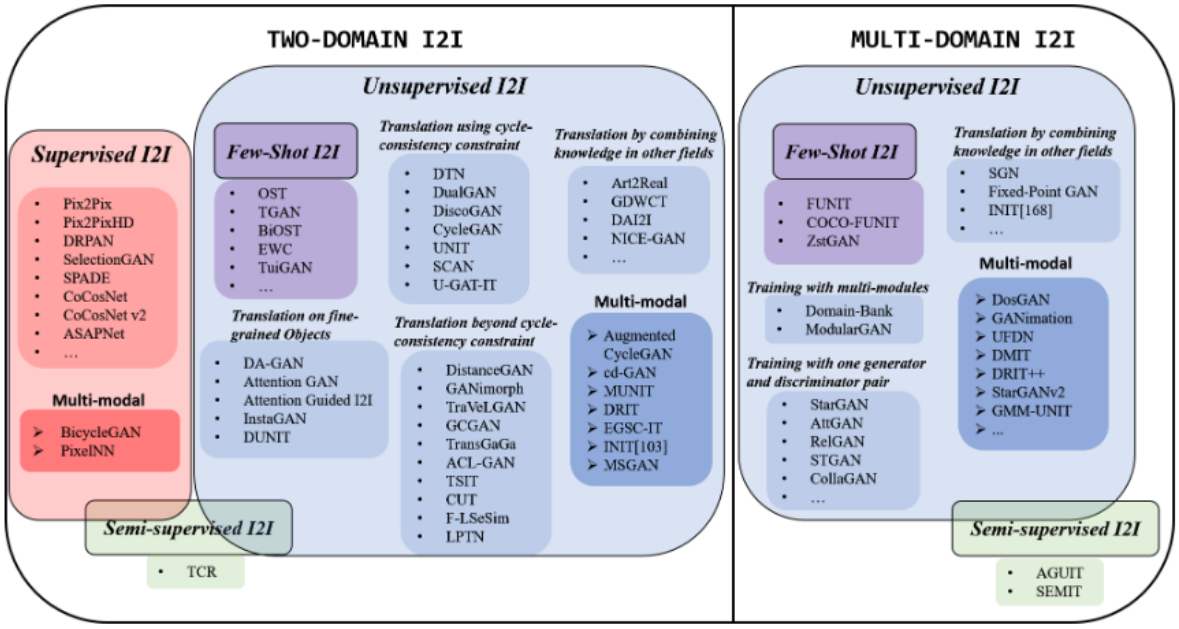

Figure 2.23. *Previously proposed methods for image to image translation [59].*

encoders (VAEs) [27, 45]. These will not be elaborated on in detail but certain aspects of these architectures are highlighted. Auto-encoders receive an input image and shall output the same image. Architecturally they consist of an encoder and a decoder. Between these two there is a representation of the input image in what is called the latent space. Auto-encoder variants differ not only between the layers and the number of parameters but also introduce additional mechanisms to ensure certain properties of the representation in the latent space [26]. The previously described UNet architecture can be seen as an example for such an encoder-decoder architecture where the contracting part is the encoder and the expanding part is the decoder,

see 2.21.

Common loss functions used in other tasks usually do not work as well for image generation tasks. The L_1 metric for example is able to capture low spatial frequency differences in the image but fails to quantify the high frequency differences [42]. To tackle the problem of image similarity quantification or put differently, to quantify the quality and perceptual differences that classical loss functions cannot address different methods have been proposed for NNs.

GANs employ an architecture that has two basic parts, namely

- a generator, that generates an image given a vector (e.g. a noise vector), and
- a discriminator, which is trained to distinguish the generated image from a real image. The discriminator serves as a term in the loss function of the generator.

The training of this network usually first updates the weights of the generator and then the weights of the discriminator per batch. Due to this game-theoretic two-player game, the training has proven difficult in terms of convergence empirically.

An alternative to training a discriminator network is the use of a NN that was pretrained on the target domain on a variety of datasets and tasks. Given such a pretrained network, the top-most layers are discarded and both the generated and the corresponding true image are passed one after another into the network. The L_2 norm of the resulting vectors from the last non-cut layer of the top-cut pre-trained network is then calculated such that a NN based quantification of the differences in features of the images can be used as a loss fuction. This is called a perceptual metric [98].

Already before the rise of NNs' popularity, metrics have been introduced to quantify differences in images, usually for applications in photography, TV and cinematic images. The peak signal to noise ratio (PSNR) [dB] is defined as the logarithm of the maximal possible pixel value (e.g. 255 for grayscale or RGB images or 1 for normalized images) divided by the L_2 norm of the two images [66].

$$\text{PSNR} = 10 \cdot \log\left(\frac{\max}{L_2}\right)$$

Further, the structural similarity index measure (SSIM) quantifies the difference based on the luminance l, contrast c and the structure s of different windows of the images:

$$l(\overrightarrow{x}, \overrightarrow{y}) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1}$$

$$c(\overrightarrow{x}, \overrightarrow{y}) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2}$$

$$s(\overrightarrow{x}, \overrightarrow{y}) = \frac{\sigma_{xy} + c_3}{\sigma_x + \sigma_y + c_3}$$

where μ_x and σ_x are the mean and the variance of the pixels in the window, σ_{xy} the covariance and $c_i = (k_i L)^2$ is used to stabilize the denominator based on the dynamic range L of the pixel values and some constant k_i. The SSIM is then computed as

$$\text{SSIM} = l + c + s$$

and is in the range $[0, 1]$ where values closer to 1 mean more structural similarity.

2.2.5. Deep Learning-based Image-to-Image Translation for MRI

[89, 38, 91, 96] propose methods to do the image-to-image synthesis problem from MRI. All of these approaches use one of T_1-, T_2- and PD-weighted images as source and target modality each, that is a modality with a single channel. [91] and [38] use unpaired data for training where the dataset consists of a set of images, while paired datasets is a set of pairs of corresponding images. They employ coupled sparse representations [10], which is per se not a NN-based method. [89] proposes a location-sensitive deep network on a voxel basis, mainly utilizing dense layers, the sigmoid activation function and multiplicative interactions — an alternative to pooling that the authers propose. The loss function is the L_2 metric. [96] employs a conditional GAN, that generates an image from an input image instead of a noise vector. The authors generate the image slice-wise and use the PatchGAN loss, introduced in [42].

[5] generates the scalars computed from the diffusion tensor directly from 12-phase-cycle bSSFP volumes in a voxel-wise manner. The L_2 norm is used as a loss function along with an uncertainty term that is used to output the uncertainty of the predicted voxel as well. The network employs dense layers where the amount of neurons per layer and the number of layers as well as the batch size are subject to a secondary optimization algorithm. Further, not only the voxel to be predicted is passed as input to the network but also the neighboring 9 voxels. The general method employed is shown in 2.24.

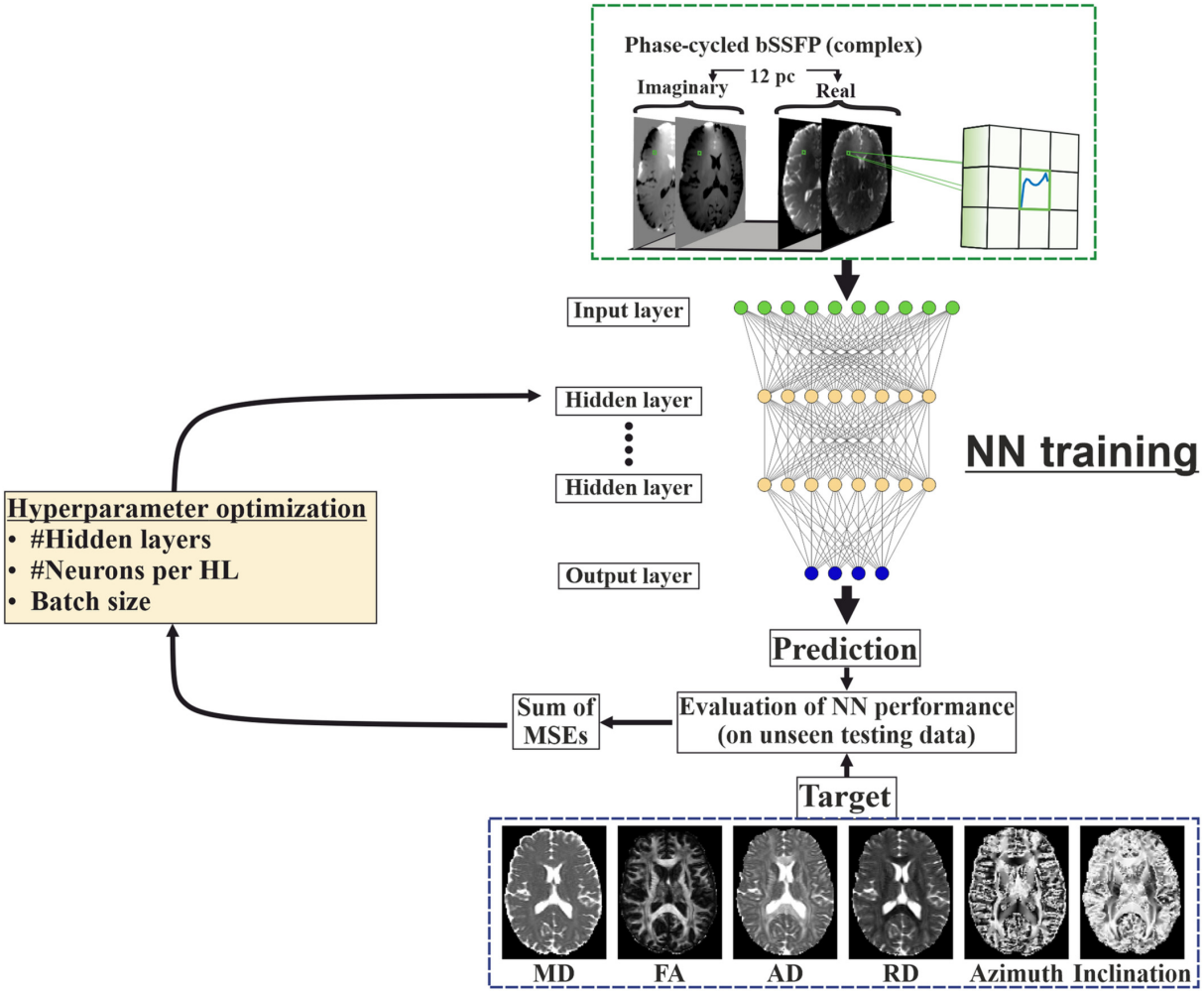

Figure 2.24. *The method employed by Birk et al. [5].*

3. Methods

In this section, the dataset used, preprocessing and details on how the estimations were generated are presented.

3.1. The Dove Dataset

The dataset acquired for the DOVE study contains a variety of modalities recorded at **3T** B_0 **field strength (Ask Flo)** using a Magnetom Prisma produced by Siemens Healthineers in Erlangen, Germany with 64-channel receive head coils. Besides our main modalities of interest — phase-cycle bSSFP and DWI — **MP2RAGE Ask Flo** [52] images were acquired as well.

The dataset contains 120 healthy subjects each of which was scanned multiple times in 3-5 sessions at different times of the day. Phase-cycle bSSFP data was acquired in 3 sessions for most participants. The whole brain was imaged in 3D sagitally with 12 phase cycles with phase increments uniformly distributed between $[0°, 360°]$. A flip angle of 15°, an isotropic resolution of 1.4 mm^3, a T_E of 2.4 ms, T_R of 4.8 ms was used with a field of view of 224 mm and a bandwith of 401 Hz per voxel. To the best of the authors knowledge, this is the largest collection of **bSSFP images Aks Flo** that has been acquired so far.

DWI was acquired in 4 sessions using an interleaved multi-slice single-shot SE-EPI DWI sequence with a b value of 1000 s/mm^2, a resolution of 2mm isotropic, a T_E of 62ms, a T_R of 3500ms, a field of view of 220mm, fat supression using saturation.

Finally, magnetization-prepared two rapid gradient-echo T_1-weighted images were acquired in one session sagitally at a resolution of 1mm^3, with a T_E of 2.98 ms, a T_R of 5000ms in a single shot fashion with a field of view of 256mm and flip angles of 4° and 5°.

3.2. Preprocessing

Regarding preprocessing, BIDScoin was used to convert the dataset from a collection of DICOMs per scan to one NIfTI file with a folder structure complying to the BIDS standard [100, 97, 28]. Nibabel was used to read and write data [7]. All modalities were then coregistered to the T_1-weighted image using SPM [20].

For **bSSFP**, the phase was rescaled from the scanner-produced range $[0, 4095]$ to radians $[-\pi, \pi]$ and phase correction was applied by multiplying the phase image of each of the 12 phase-cycles by a complex exponential with the phase of the sum of the complex data in the exponent.

$$\text{Phase corrected}_i = \text{Phase}_i \cdot \exp\left(-j\angle\left(\sum_{x,y,z} \text{Complex}_{i,x,y,z}\right)\right)$$

Next, artifacts due to the Gibbs phenomenon were removed using DiPy [22]. The complex bSSFP data were resampled to the resolution of the DTI data — 2mm^3 — using FSL [79] and a mask generated from the DTI images was applied. Finally, phase and magnitude were extracted,

normalized to be in the range $[0, 1]$ for each phase and magnitude across the whole dataset and aligned into one channel alternatingly, such that the image has 24 channels, each contianing one 3D volume. One sample of the first two channels is shown in 3.1, 3.2. In another experiment,

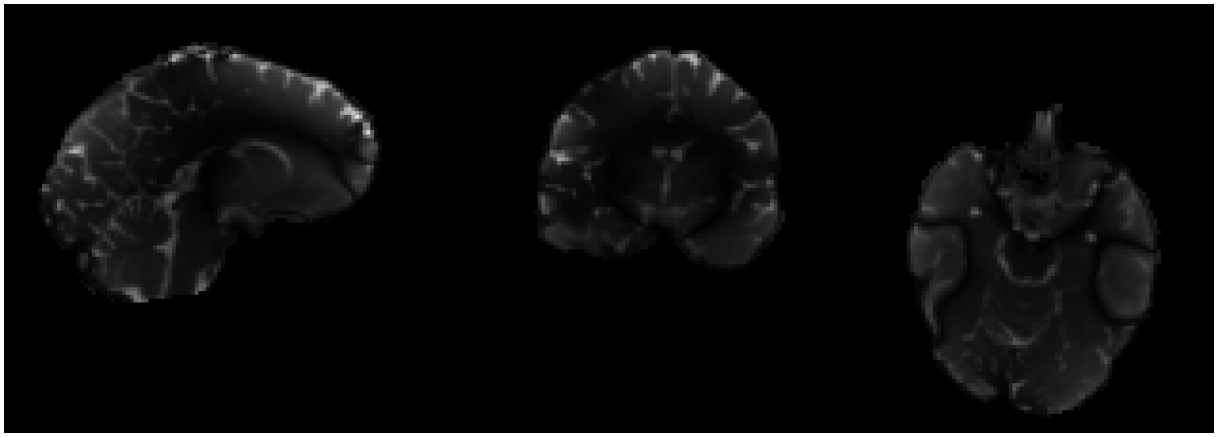

Figure 3.1. *A data sample of the first channel containing the magnitude of the first phase cycle of the preprocessed pc-bSSFP images that is fed into the neural network.*

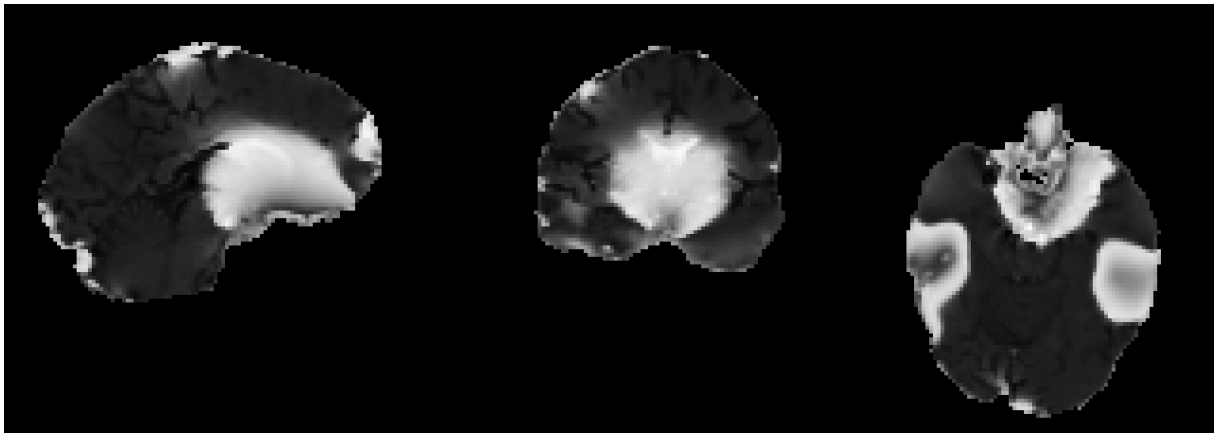

Figure 3.2. *A data sample of the second channel containing the phase of the first pahse cycle of the preprocessed pc-bSSFP images that is fed into the neural network.*

a derived image is used: Instead of using all 12 phase cycles, only one phase-cycle is repeated 12 times to generate an image that have the information content of a single phase cylce only while having the same shape.

DWI data was distortion corrected using FSL's [79] `TOPUP`, the brain was extracted using `BET` and Eddy currents were corrected with `eddy`. Then `DTIFIT` was applied to estimate the diffusion tensor from the images, resulting in 6 channels corresponding to the upper triangular tensor elements per voxel. Channels were then normalized to the range $[0, 1]$ across the whole dataset where the diagonal elements of the tensor and the off-diagonal elements were normalized separately. A sample of the first two channels is depicted in 3.3 and 3.4.

MP2RAGE T_1-weighted images are masked and resampled to match the DTI resolution. Then data was normalized to the range $[0, 1]$ and repeated to 6 channels such that it matches the DTI input dimensionality. A sample of this is shown in 3.5

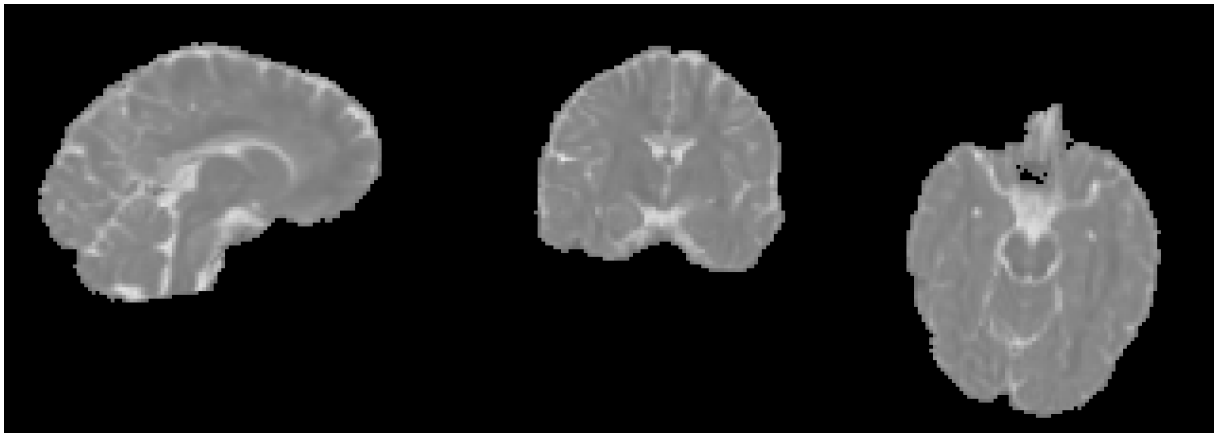

Figure 3.3. *Sample of the first channel of a preprocessed DTI corresponding to the D_{xx} element of the tensor.*

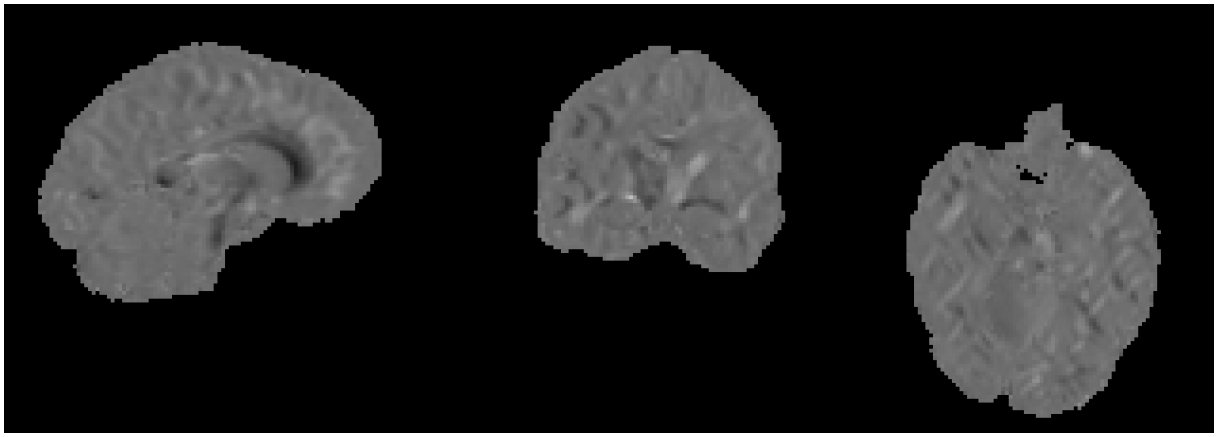

Figure 3.4. *Sample of the second channel of a preprocessed DTI corresponding to the D_{xy} element of the tensor.*

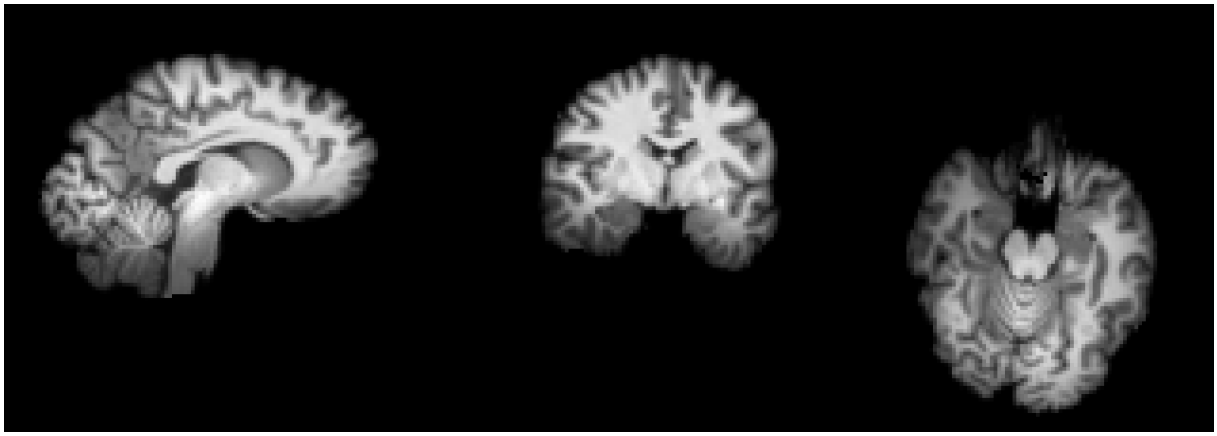

Figure 3.5. *Sample of a preprocessed T_1-weighted image fed to the NN.*

3.3. Machine Learning-based DWI Tensor estimation from bSSFP Data

Several libraries and frameworks are used to implement the neural network that predicts the diffusion tensor from the inputs:

- PyBIDS [97] provides means to query a BIDS-compliant dataset to mitigate the overhead of traversing the folder structure.
- TorchIO [61] is a library for MRI-based data loading pipelines that supports extended data augmentation.
- PyTorch [60] is a framework for programming neural networks providing layers like convolutions and transpose convolutions.
- PyTorch Lightning [18] is an extension built on top of the Torch framework to ease distributed training, finding the pareto-optimal learning rate with respect to convergence speed and quality and extended logging facilities.
- MONAI [11] offers a set of pre-defined blocks, architectures, metrics and loss functions for deep learning in medical applications.
- NumPy [31] provides basic matrix and tensor functions in an efficient manner to post-process data for example computing Eigenvectors and -values.
- Pandas [68] provides data structures and functions to construct and aggregate tabular data for evaluation of e.g. relative errors per voxel per image and aggregate them into average errors per voxel per segmentation mask.
- Seaborn [93] and Matplotlib [40] are libraries for plotting data.

3.3.1. Data Sampling and Augmentation

Deep Learning requires many data samples to produces results of acceptable quality. As there are mostly only two sessions per subject where both DWI and bSSFP were recorded, all bSSFP samples are paired with all DWI samples per subject across sessions. The goal is to infer the diffusion tensor for a subject, not necessarily specific or sensitive to variations in the time of the day, daily differences and the like. If only the two sessions would be used, there would only be about 250 data samples which is little compared to the thousands or even milions of samples that NNs usualy require. When combining all images across sessions, we get $4 \cdot 3 = 12$ samples per participant instead of just two, resulting in approximately $12 \cdot 120 = 1440$ samples. However as not for all subject, all sessions were recorded either due to subjects compliance to the scanning schedules or other reasons, 1077 samples are available.

To further increase the non-identical data samples during training, data augmentation was applied using TorchIO. Data augmentation is the act of slightly introducing noise and other pertubations into a data sample, that are characteristic to the source of the data, such that the NN is not trained on the exact same image but slight variations of it. This leads to better generalization and robustness to noise.

The augmentations added to the training inputs is two-fold:

1. a random bias field is added with a probability of 25% to mimic magnetic field inhomogenieties. It is modeled using a linear combination of polynomial basis functions as described in [88]. In this work the maximal order is 3 and the coefficitents are sampled uniformly between $[0, 0.25]$. An example of a bias field augmentation with greater coefficients than used in this work is shown in 3.6.

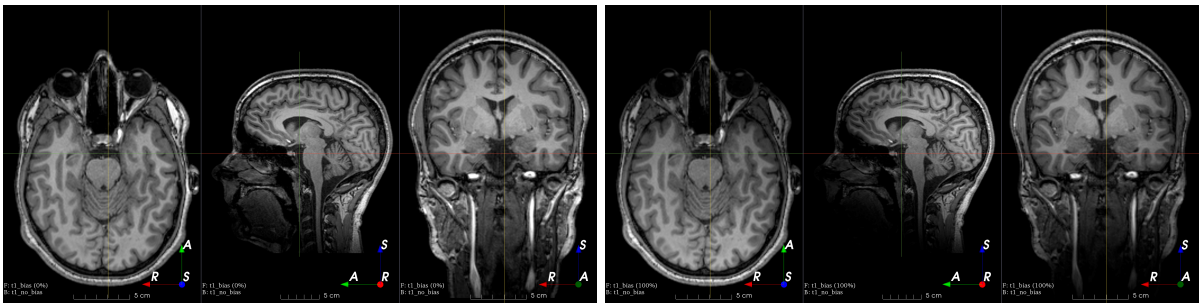

Figure 3.6. ***Left:*** *Example image before a random bias field was added.* ***Right:*** *Example image after a random bias field was added [1].*

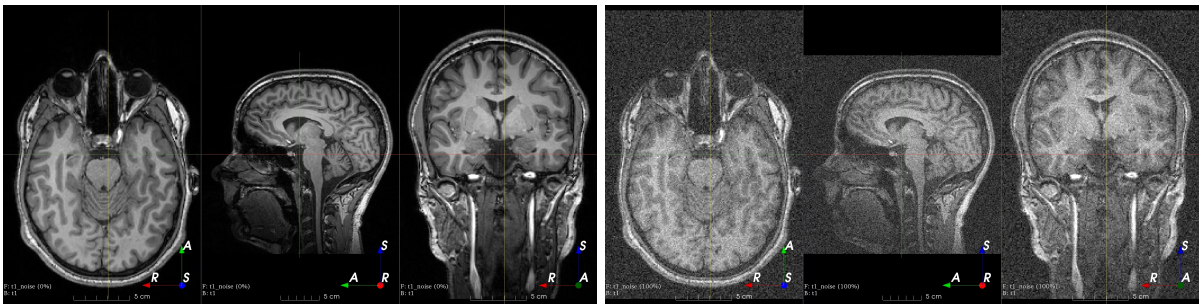

Figure 3.7. ***Left:*** *Example image before a Gaussian noise was added.* ***Right:*** *Example image after a Gaussian noise was added [1].*

2. gausian random noise is added to the image with a probability of 25%, zero mean and a standard deviation in the range of $[0, 0.01]$. An example of strong gaussian noise augmentation is shown in 3.7

Effectively, the augmentation is minimal as can be seen in 3.8. **Ask flo**

3.3.2. Architecture

The backbone architecture of the model follows the standard UNet architecture and bases on the implementation of BasicUNet of the MONAI package. It makes the following modifications:

- normalization and dropout is added at each block to increase generalization and cause faster convergence,
- 3D inputs are supported as the task is to train on the full 12-pc-bSSFP,
- input and output dimensions are adapted to the task,
- Instead of ReLU, PReLU is used, to add more flexibility to the model,
- the number of blocks and features are adapted to match the input dimensions. In this work 5 blocks for each contracting and expanding paths are used with the number of feature maps per block: $48, 96, 192, 384, 768, 384, 192, 96, 48, 24$

Besides the backbone, input heads for each modality have been added. They consist of a ResNet block as shown in 2.20 with 6 input channels, 24 output channels and 3 convolutionals layers with ReLU activations, and batch normalization with a batch size of 1, which boils down to instance normalization. A basic overview of the modified architecture is shown in 3.9. A more detiled plot can be found in the appendix .1.

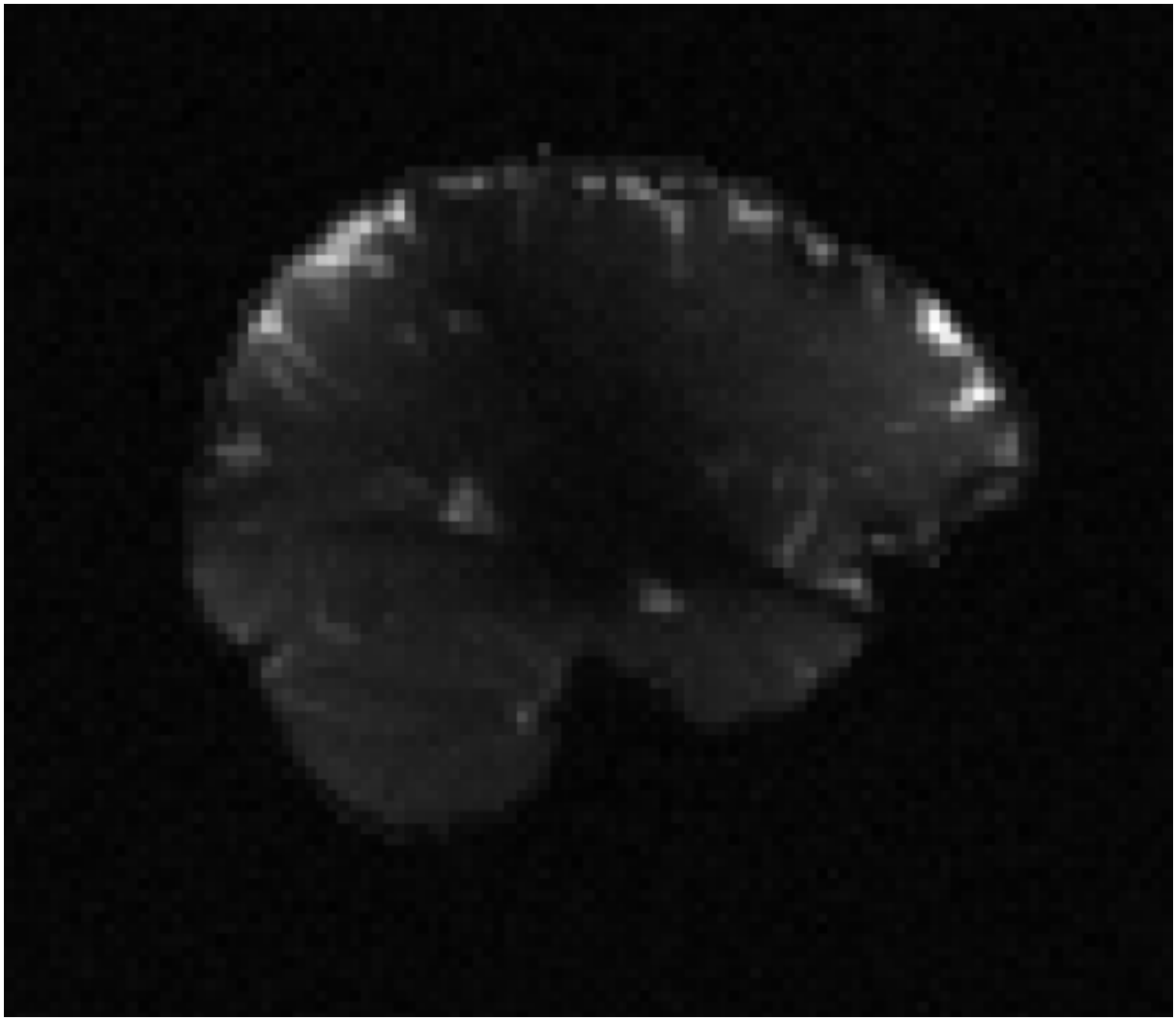

Figure 3.8. *Samples of the training data after data augmentation was applied (with a probability of* 1*).*

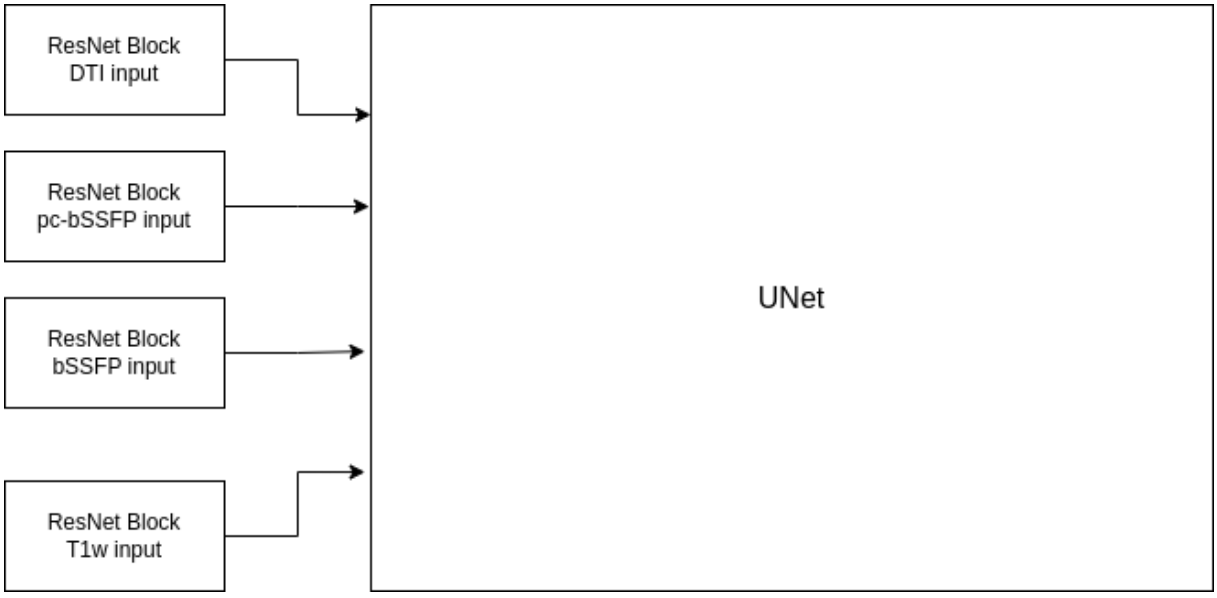

Figure 3.9. *The architecture of the NN used to predict DT from bSSFP.*

3.3.3. Training

The loss function that is optimized is a sum of three different loss functions:

1. the L_1 loss function to capture low frequency features and provide an optimization criterion which minimizes the differences between source and target based on the actual voxel value.
2. the SSIM loss, which is defined as $1 - \text{SSIM}$.
3. the perceptual loss function with MedicalNet as pretrained netork [13]. MedicalNet is based on ResNet-10 [33] and has been trained on a large dataset containing different organs, modalities and pathologies.

AdamW is used as optimization algorithm with a leanring rate $\epsilon = 1e - 4$, running moment coefficients 0.9 and 0.999, and a weight decay of 0.01. Further, the training was distributed on 4 NVidia GeForce RTX 5000 GPUs using the distributed data parallel strategy, that split the dataset into 4 and performs training on each subset to merge the losses at the end of each epoch and aggregate the gradients accordingly. The update is then executed on a single GPU and the next epoch is branched to 4 GPUs again. Early Stopping is used as termination condition with a patience of 5 epochs.

Two approaches have been explored. First, training the modalities with randomly initialized parameters directly on the DT generation task. The second approach is to pre-train an auto-encoder, i.e. a network that takes the DT as an input and generates it. This helps to store information about the task and output modality in the parameters of the model. Next, we transfer the NN to the new task of predicting the DT from a different modality by training theResNet input block, while the rest of the parameters in the NN remain frozen. Then, all weights are set to trainable and the learning rate is decreased to $1e-5$ for the finetuning stage.

After training, the 108 unseen data samples are used to evaluate the model. For that, the model is used to infer a prediction and the loss is calculated . Each predicted normalized diffusion tensor is divided into 3 ROIs corresponding to CSF, GM and WM. The average relative error per voxel is then computed for each of the diffusion tensor elements and ROIs. Then the DT is de-normalized by applying the inverse transformation, i.e. $x \cdot (\max - \min) + \min$ where x is the value of the voxel and $\min, \max$ are the minimum and maximum value of the DTs across the whole dataset. From that, the scalars introduced in 2.1.3 are computed. For each of these scalars, again the average relative error per voxel, ROI and modality is computed.

4. Results

When directly training the task to predict the DT from a given modality, the training time is a lot smaller compared to the pre-trained version. The qualitative and quantitative differences between modalities are shown in this section.

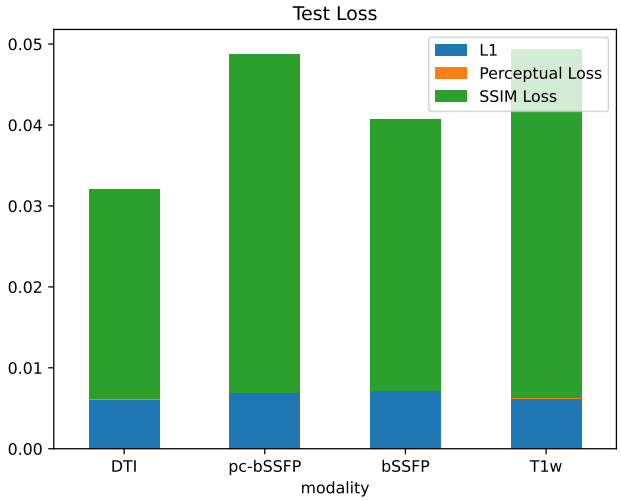

Figure 4.1. *Test loss of the models for each modality when training the task without additional initialization steps.*

4.1. Single-Stage Training

In 4.1 we can see the test loss for each modality, split by the sub part of the loss. For all modalities, the perceptual loss is very small. The majority of the loss is based on the SSIM loss.

While all losses are close in magnitude, the aut-encoding task works best, while pc-bSSFP performs worst without additional initialization steps.

However — as can be seen in 4.2 — all predictions contain artifacts of different sizes hinting, that the direct training approach does not suffice to generate results of acceptable quality. Still, the scalars were computed and evaluated to see how robust each scalar is.

Quantitatively, the diagonal elements of the diffusion tensor are predicted with a relatively small error between 5% and 30%. The off-diagonal elements of the diffusion tensor vary greatly between modalities: 10% for DT up to 3000% for pc-bSSFP. Surprisingly, bSSFP data with just the first phase cycle repeated performs astonishingly well.
The data is shown in 4.3.

The mean relative errors of mean diffusivity is — given the artifact and the bad prediction quality of off-diagonal elements — very robust as shown in 4.4. Errors are in the range of 20% to 50% which however still renders them too imprecise for real-world applications. For the fractional anisotropy, the case is a lot clearer: not a single modality performs better than 100% relative error. Even the previously rather acceptable result of the auto-encoder is not confirmed as this performs wost with errors of up to 600%.

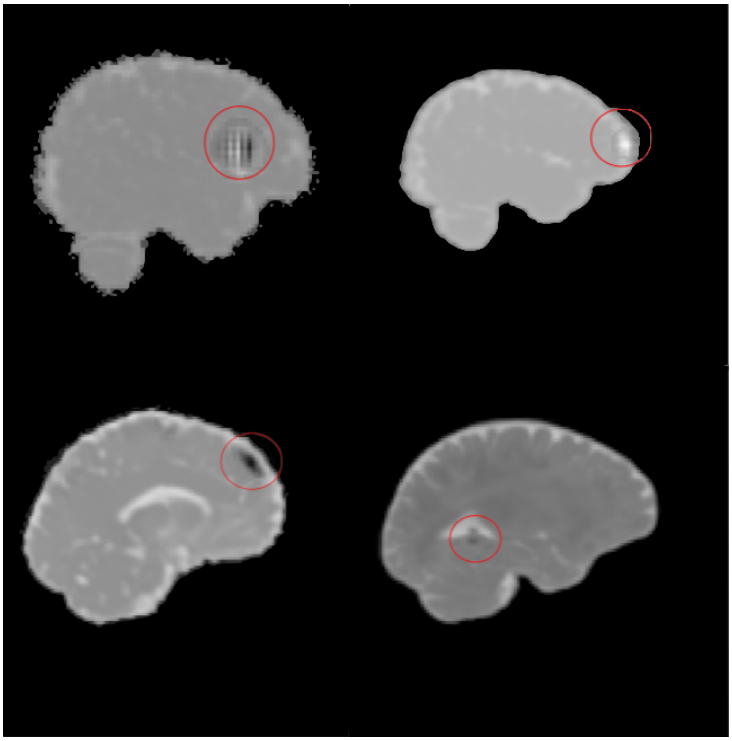

Figure 4.2. *Prediction examples for all modalities: left top — DT, right top — pc-bSSFP, left bottom — bSSFP, right bottom — T_1-weighted.*

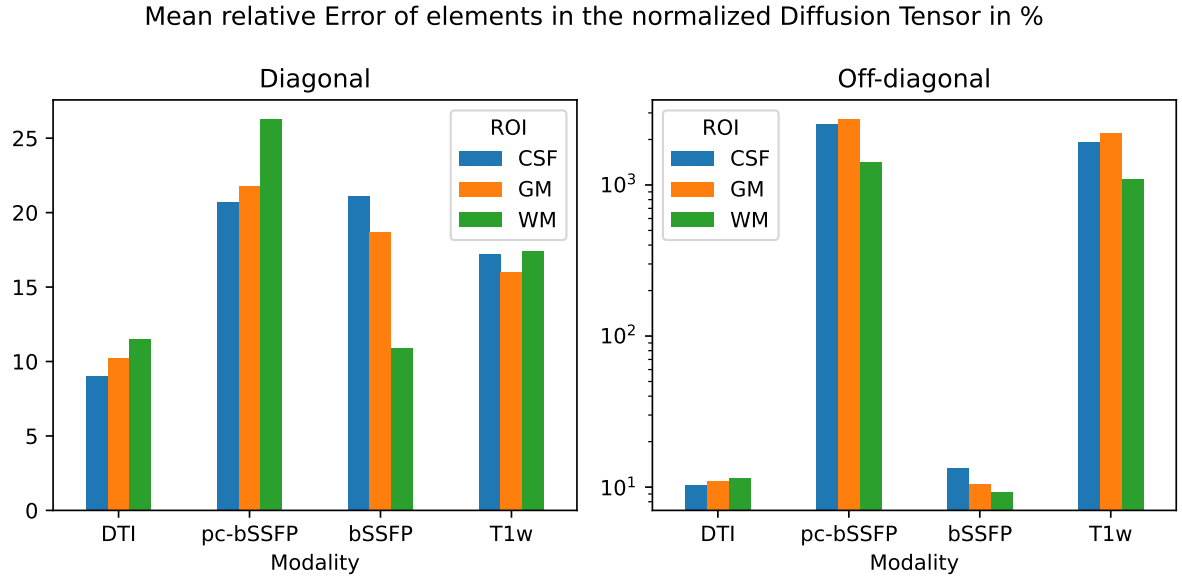

Figure 4.3. *Mean relative error of the DT elements for each modality and group, split by diagonal and off-diagonal elements for directly trained models.*

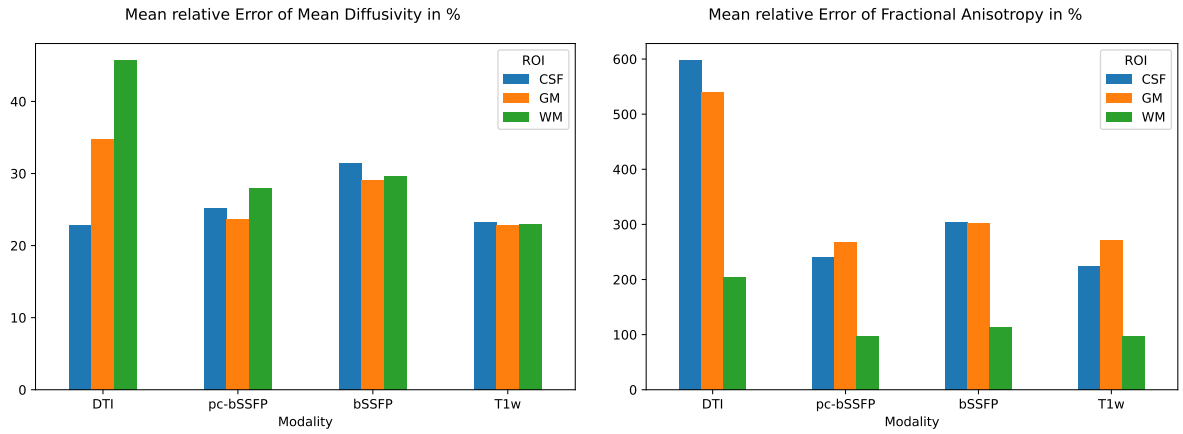

Figure 4.4. ***Left:*** *Mean relative error of the mean diffusivity for directly trained models.* ***Right:*** *Mean relative error of the fractional anisotropy for directly trained models.*

This is another indicator, that the chosen training strategy does not suffice for the taks.

4.2. Multi-Stage Training

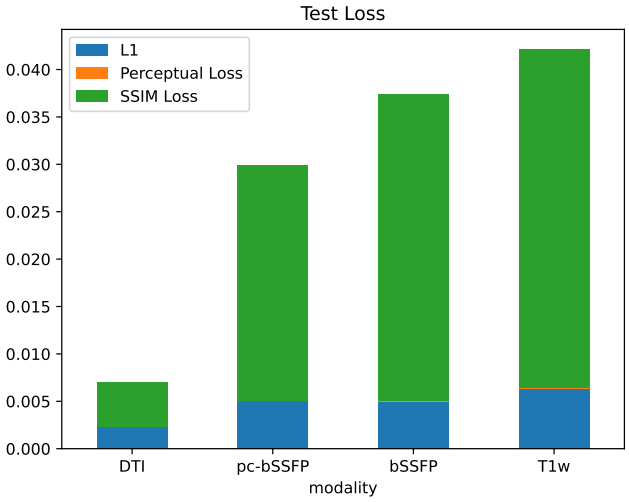

Figure 4.5. *losses*

Therefore, pre-training has been applied such that information about the output modality is encoded in the weights on initialization of the training for the actual task. The losses are comparably in magnitude to the direct training. However, DT performs much better, pc-bSSFP as an input is superior to bSSFP and T_1-weighted in terms of the loss.

4.2.1. Quantitative Results

When comparing the mean relative error of the elements in the normalized diffusion tensor, we can see in 4.6 that while the diagonal elements are predicted with comparable performance to the direct training, the off-diagonal elements improve order of magnitude for mos modalities. According with the loss, pc-bSSFP input outperforms both bSSFP with the first phase cycle repeated and T_1-weighted inputs. Off-diagonal elements in the white matter still have a rather

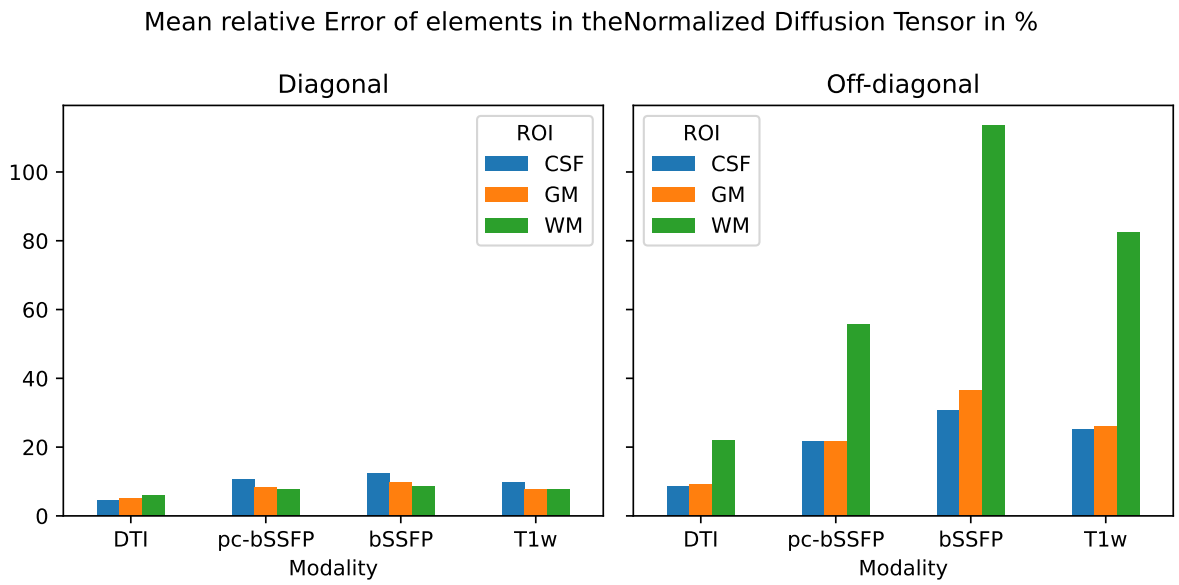

Figure 4.6. *Mean relative error of the DT elements for each modality and group, split by diagonal and off-diagonal elements for multi-stage-training models.*

large relative error compared to the other ROIs over all modalities.

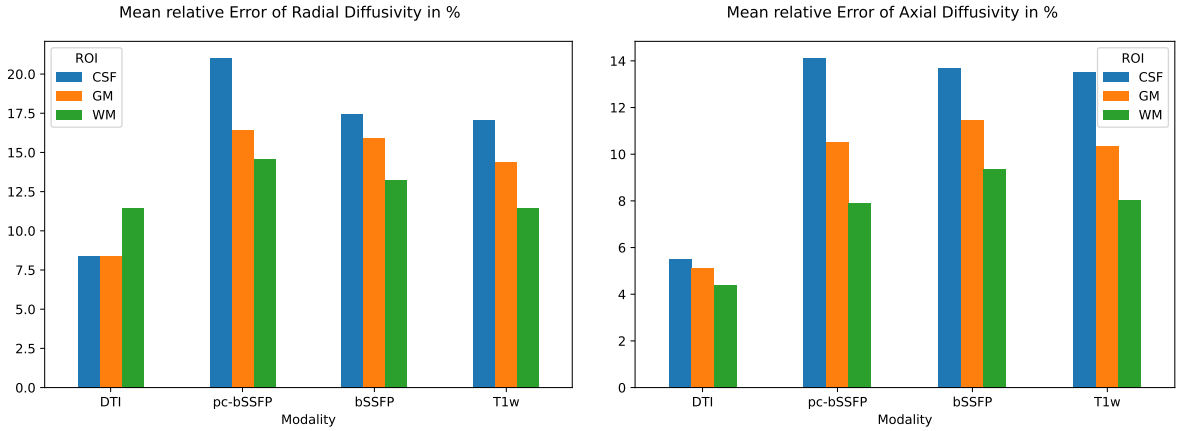

Figure 4.7. ***Left:*** *Mean relative error of the radial diffusivity for multi-stage-training models.* ***Right:*** *Mean relative error of the axial diffusivity for multi-stage-training models.*

Fir both radial and axial diffusivity, the relative errors are up to 25%. Here, the CSF is the most difficult region of interest to predict with the exception of RD when using DT as input. All non-DT inputs perform comparably within a margin of approximately 5% as shown in 4.7.

Similarly for MD and FA, we can see that DT outperforms all other inputs in terms of prediction error. Again, we can see that the CSF ROI has the largest errors with the exception of DT as an input.

Finally, inclination and azimuth are very robust with absolute errors below 0.5° for inclination and 2° for azimuth.

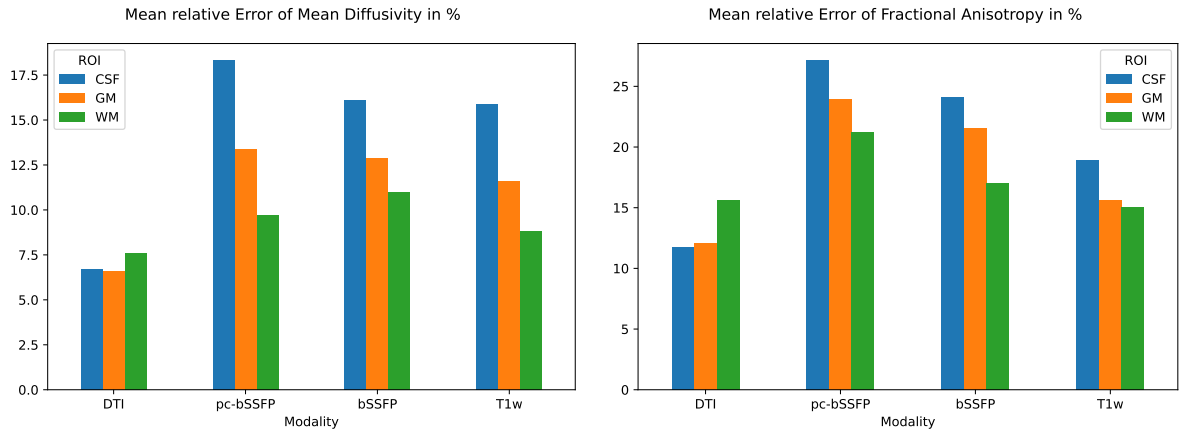

Figure 4.8. ***Left:*** *Mean relative error of the mean diffusivity for multi-stage-training models.* ***Right:*** *Mean relative error of the fractional anisotropy for multi-stage-training models.*

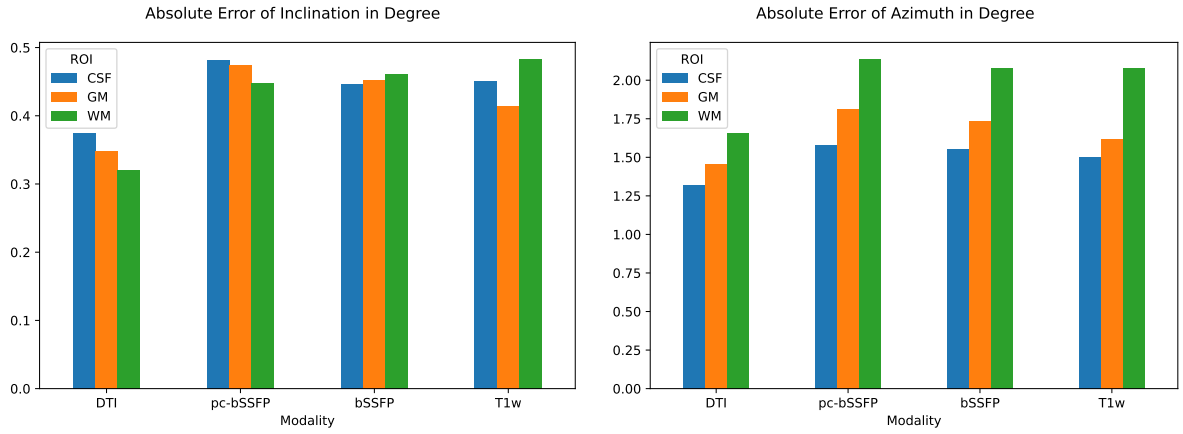

Figure 4.9. ***Left:*** *Mean absolute error of the inclination angle in degree for multi-stage-training models.* ***Right:*** *Mean absolute error of the azimuth angle in degree for multi-stage-training models.*

4.2.2. Qualitative Results

As the main goal of this thesis is to explore full volume-based inference of the DT from pc-bSSFP data, the focus is in the qualitative evaluation will be set on this input modality. Predictions for all modalities will be made available on the server of the institute. All of the following figures contains the ground truth on the left hand side and the prediction on the right handside.

One example for each diagonal and off diagonal were chosen for comparison. The diagonal element (D_{xx}, see 4.10) is predicted well in terms of detail, while the contrast is slightly different. No atrifacts are visible.

Regarding the off-diagonal example (D_{yz}, see 4.11), we can see that the contrast is mostly corresponding. The level of detail is perceivably lower than in the ground truth. Especially the high frequency components of the off-diagonal elements seem to be difficult to learn for the model with the current architecture.

For the MD, the predictions seem to be matching quite well both in terms of contrast as well as in level of detail, thus corresponding to the diagonal elements of the diffusion tensor in terms

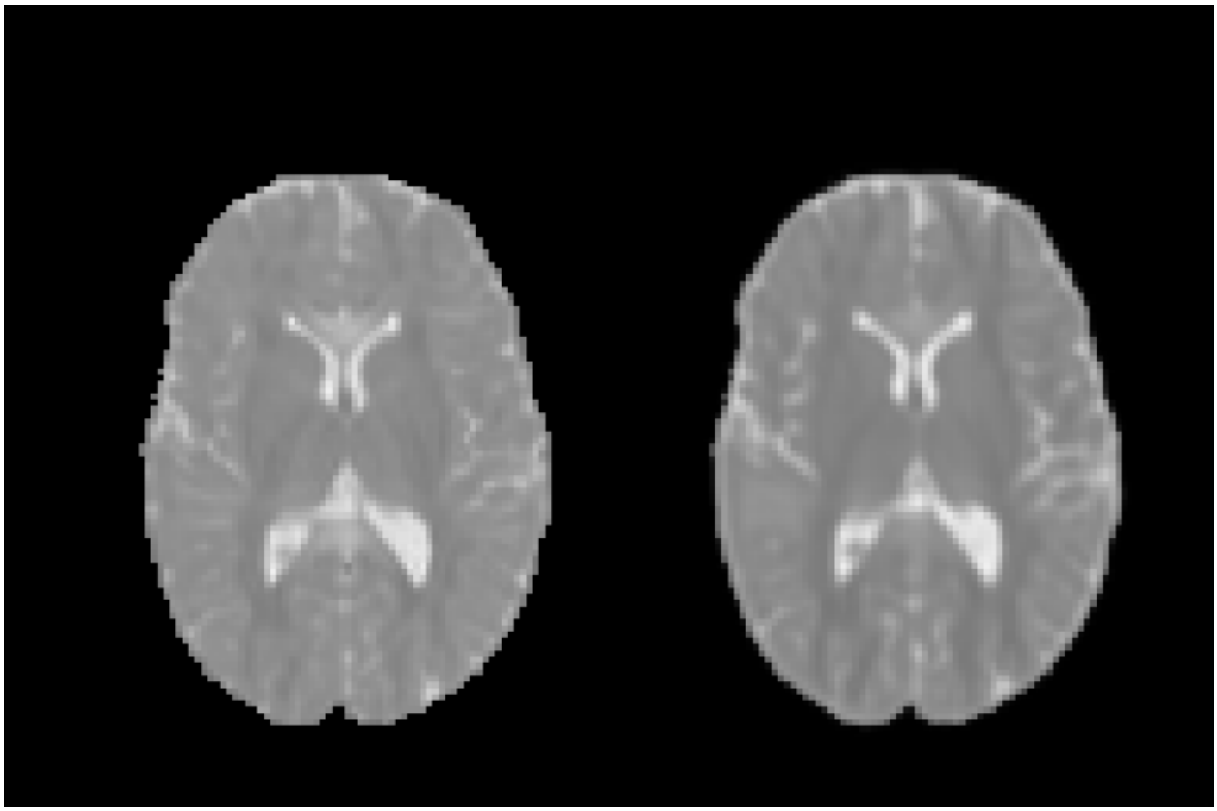

Figure 4.10. *Comparison between D_{xx} of original DT and predicted DT using pc-bSSFP as input modality.*

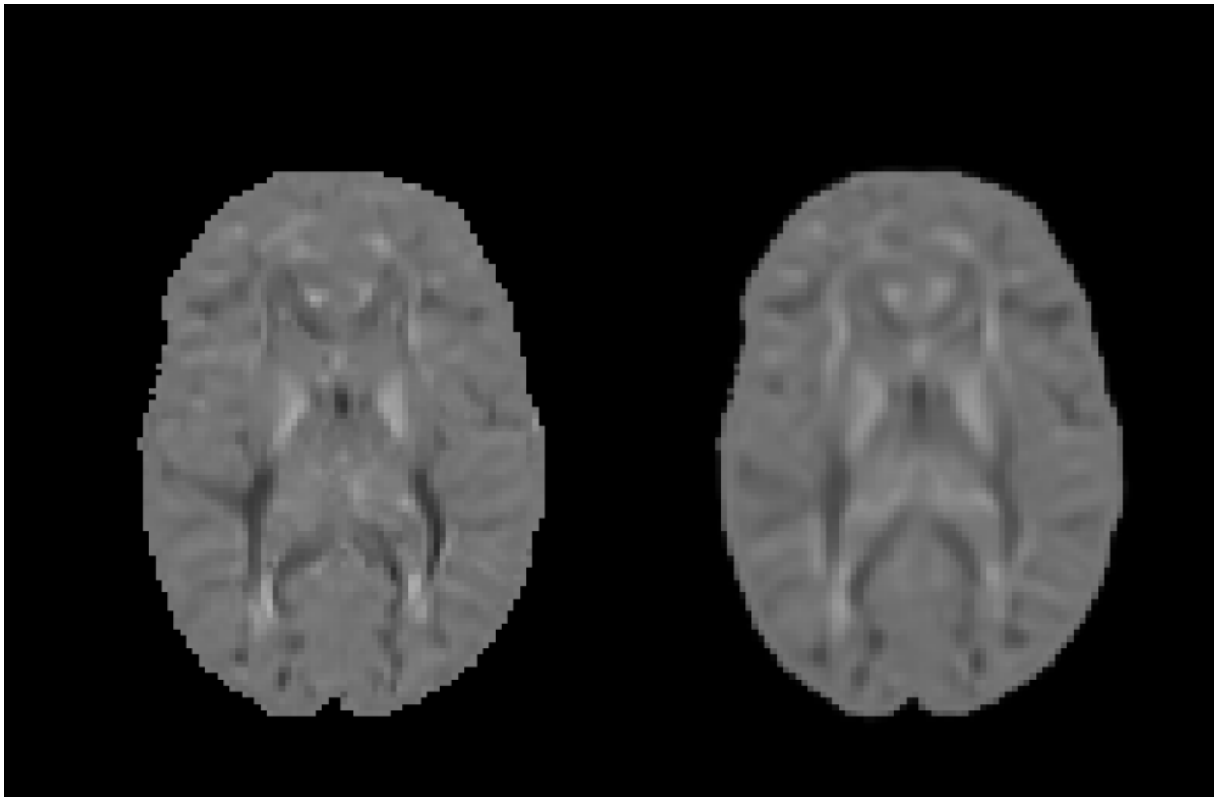

Figure 4.11. *Comparison between D_{yz} of original DT and predicted DT using pc-bSSFP as input modality.*

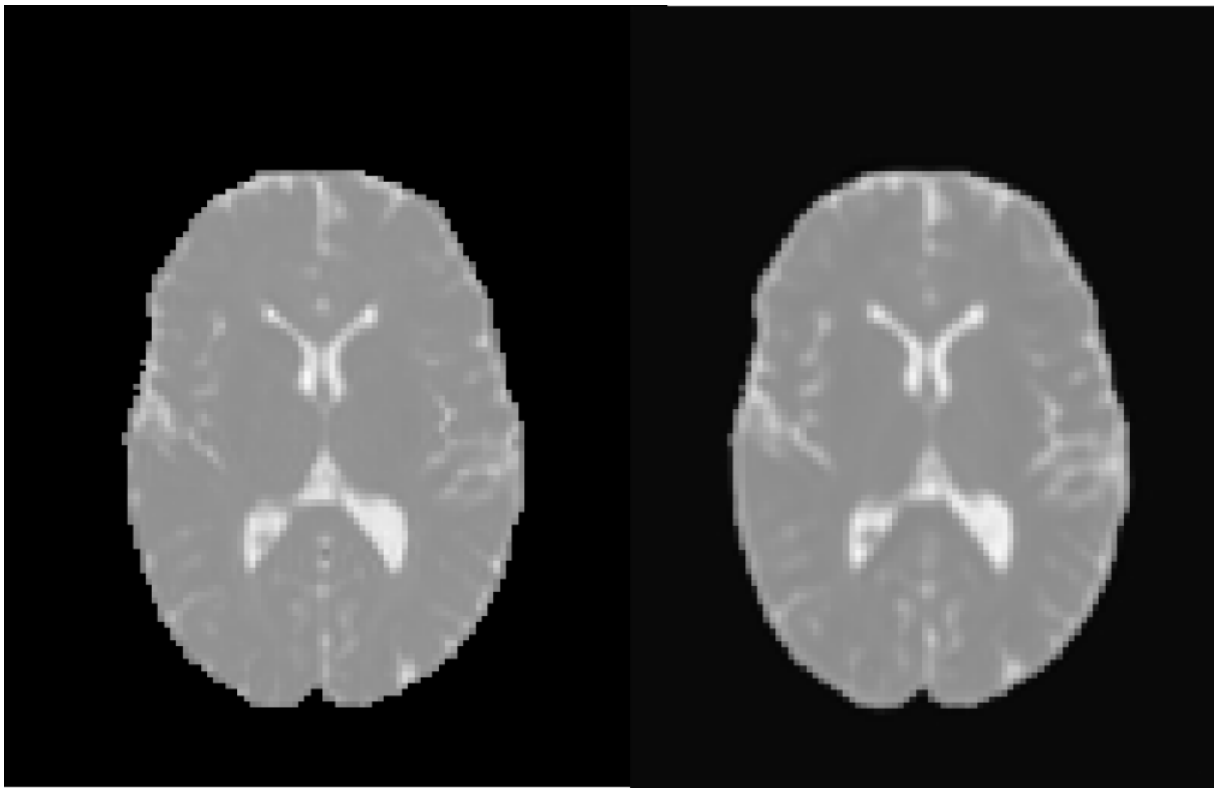

Figure 4.12. *Comparison between MD calculated from the original DT and the predicted DT using pc-bSSFP as input modality.*

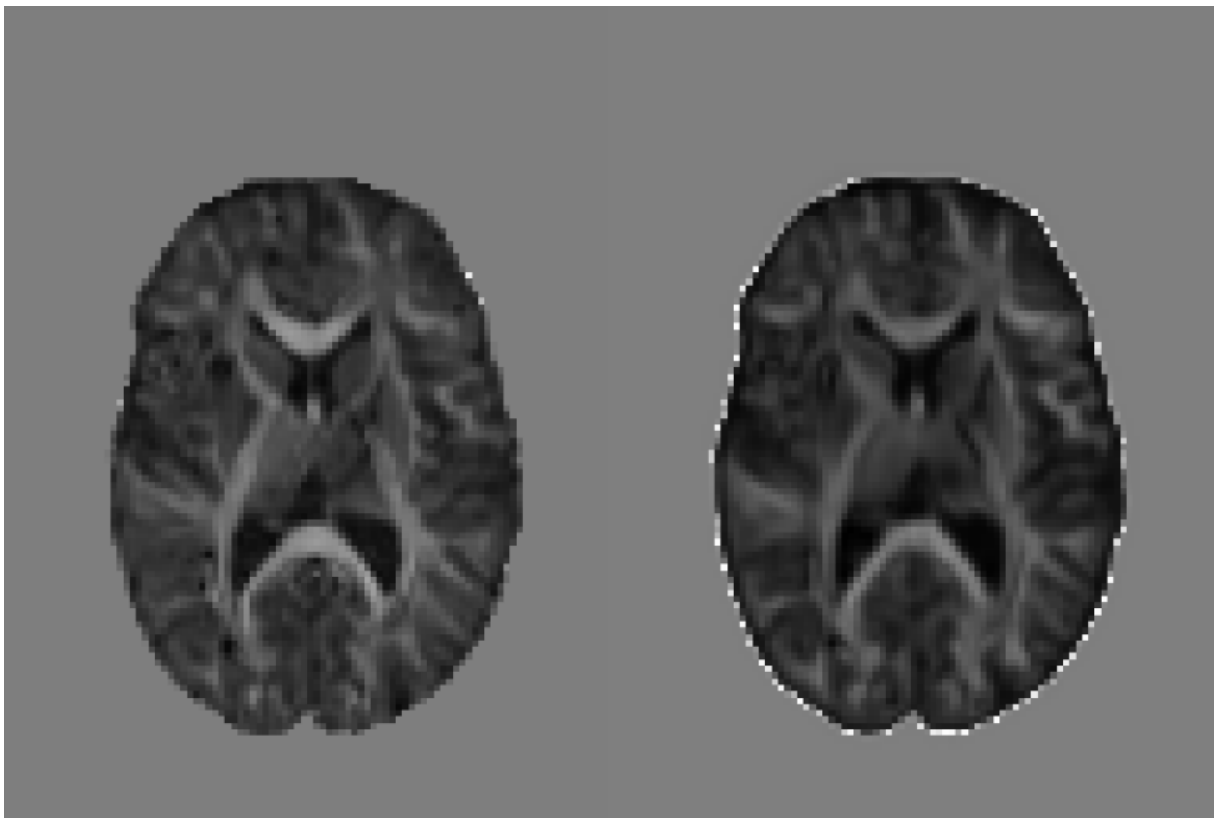

Figure 4.13. *Comparison between FA calculated from the original DT and the predicted DT using pc-bSSFP asinput modality.*

of prediction quality. Fractional anisotropy corresponds more closely to the off-diagonal tensor elements in terms of prediction quality. However, the contrast is slightly darker in the predicted FA map than in the ground truth. Further, the level of detail is lower as well.

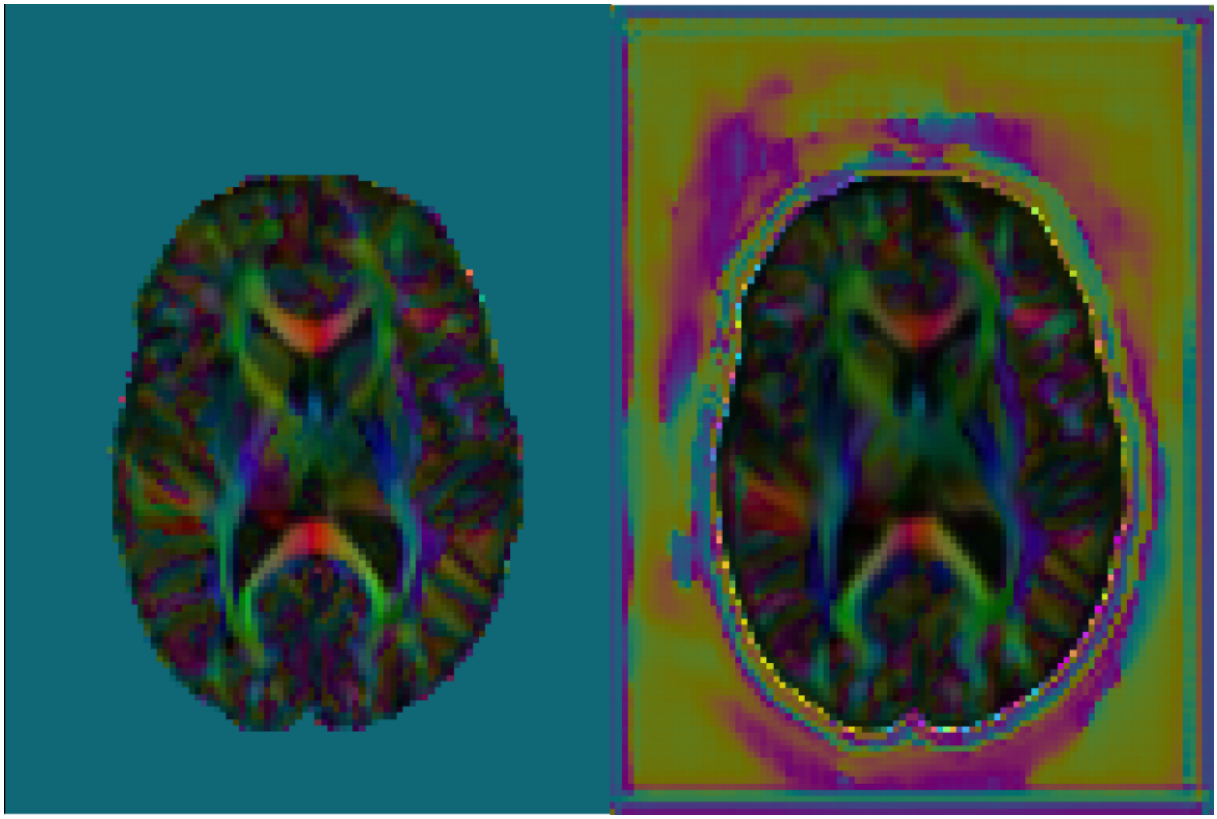

Figure 4.14. *Comparison between axial RGB images calculated from the original DT and the predicted DT using pc-bSSFP as input modality.*

To compare directionality information in a visually evaluatable way, RGB maps are shown in **??**. The main white matter tracts seem to be captured in the prediction and even some details are preserved. Nevertheless, the contrast is slightly darker and the prediction is not as detailed or crisp as the ground truth.

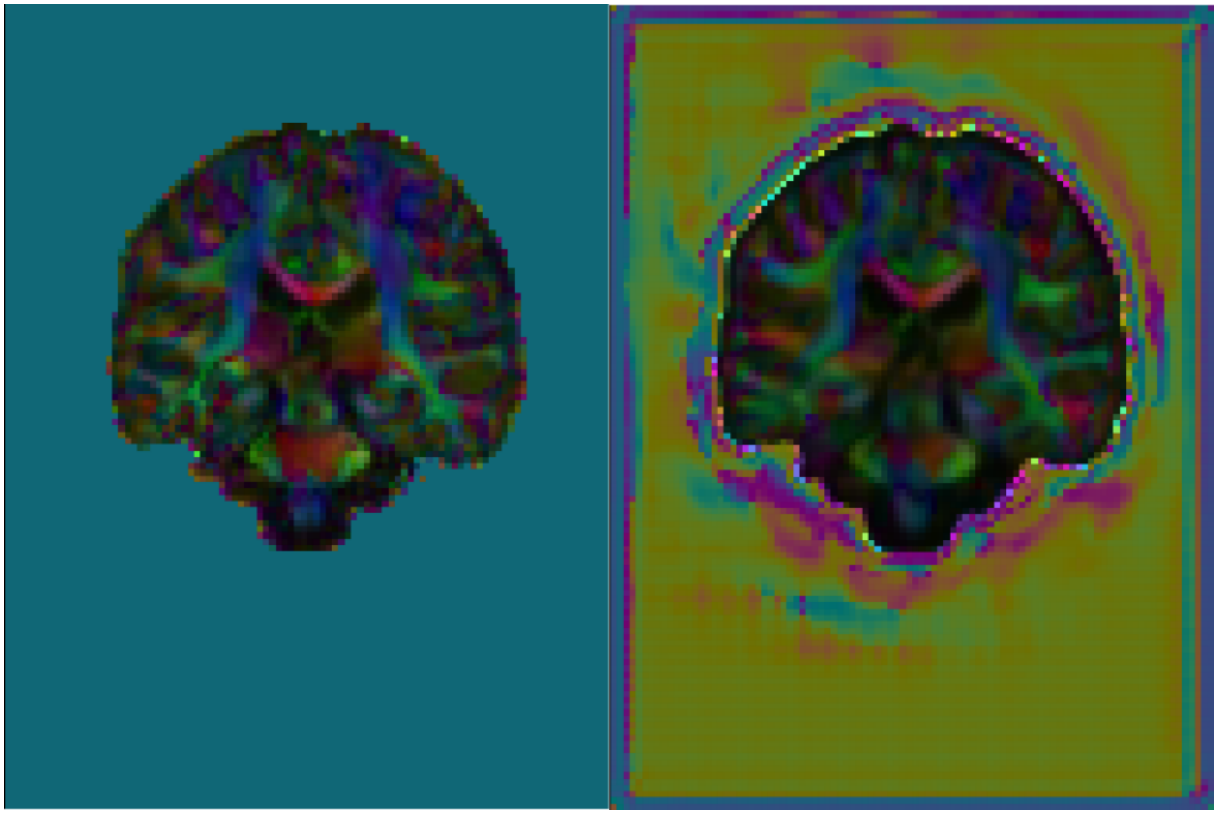

Figure 4.15. *Comparison between coronal RGB images calculated from the original DT and the predicted DT using pc-bSSFP as input modality.*

5. Discussion

As we could see in the results, the predictions are estimates within a reasonable error margin. At the same time the level of detail is not as good as in the ground truths. Thus, different strategies are described to decrease model size such that more memory demanding data types can be used for the parameters, such that the processing corresponds more to the nature of data. In this work the inputs have been treated like 3D images instead of a sequence of complex representations of bulk magnetizations. Then strategies to adapt the model architectures are discussed to not only match the weights to the complex representation but the processing as a sequence as well — instead of treating the sequence as channels in an image.

5.1. Limitations

The model described in the methods section has $51 = 5.1 \cdot 10^7$ milion trainable parameters. This equals to approximately 14 GiB of VRAM on a GPU. Several ways exist to shrink this

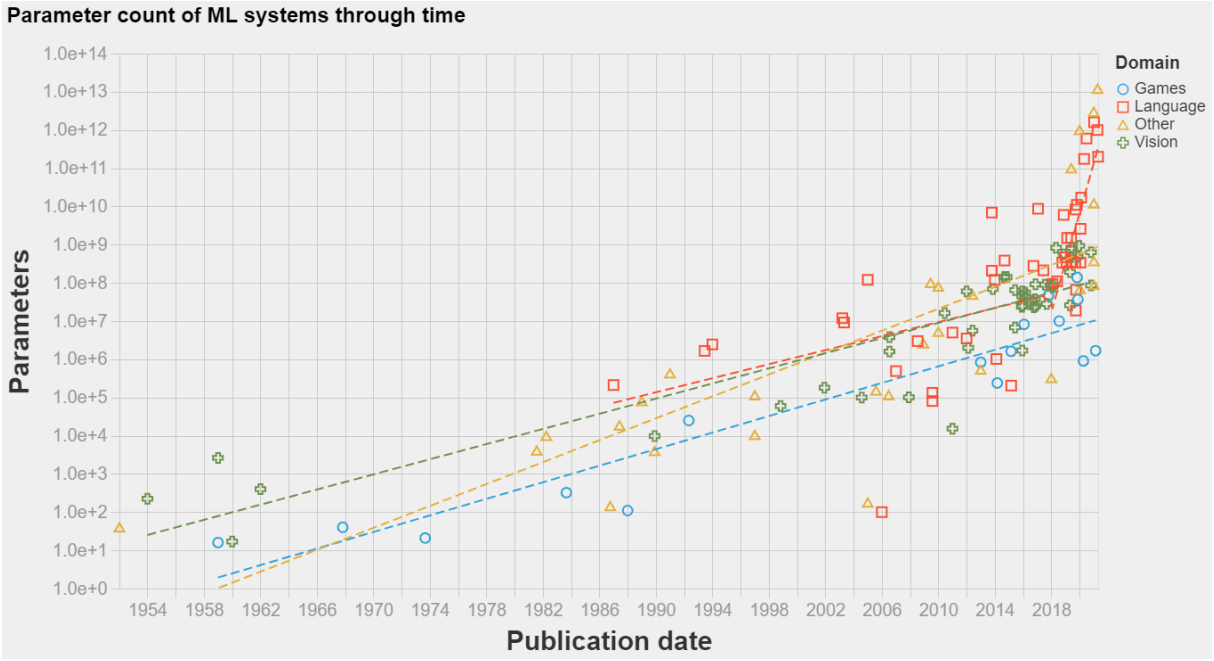

Figure 5.1. *Comparison of the number of parameters of NN models across years.*

rather large model to a smaller one. The input size is of critical importance, as it co-defines the number of weights in the NN. Smaller input dimensions result in a smaller amount of weights and thus trainable parameters. Voxel-wise regression has been performed in [5] with a rather small multi-layer perceptron architecture, with rather small relative errors, but some impreciseness in the overall structure of the predictions, e.g. in and around the ventricles. In this work, the full volumes have been input to the NN with smaller errors but a lot more computational effort both in terms of calulcations neccessary for one update as well as training duration. Those two approaches are extremes on a spectrum. A common approach in deep learning for MRI is to feed the image patch-wise [61]. With this, the amount of structural information passed to the NN can

be chosen by the user via the size of the patch as well as the amount of parameters to be trained.

Currently, the complex data is flattened into channels as well as the temporal dimension of the input. Complex weights would allow the model to accept complex inputs a priori. To take the temporal dimension into account directly — rather than indirectly, computing maps from each time step in the contracting branch and merging them in the expanding branch — either of 1D temporal convolutions, recurrent neural networks like LSTMs or vision transformers can be used [78, 47].

Here we presented one instance of a neural network per modality. This lacks the statistical power draw conclusions about which modality is best suitable to predict the diffusion tensor. To make such a statement, the model needs to be trained with different initializations (as these are random) to make specific statements. Further, the architecture was chosen based on experiments conducted by the author and common practices and can not be assumed to be the optimal architecture. Approaches like neural architecture search provide means to infer architectures in a more structured and empirical manner [94].

A NN-based approach enables the predictions of the desired quantity from a given input, however, the resulting method is not easily interpretable and does not provide many insights on or an analytical formulation of the relationship between input and output. Inital work on bSSFP and white matter tracts has been carried out by [55, 54].

The perceptual loss is used to capture high frequency components of the target, however this approach is limited. It is able to catch some higher frequency components, but fails to caputure all of them. A patch-based GAN loss as described in [42] has shown to be suitable for this purpose, however it adds additional trainable parameter to the model and slows training considerably.

The data augmentation applied in this work was quite minimal. Stronger data augmentation can help to increase robustness and generalization of the NN to unseen inputs further [26].

The masks used in this work and especially the probabilistic segementations generated by FSL for evaluation is of poor quality. Svenja K. is currently regenerating these masks and probabilistic segementations using FreeSurfer improving the quality of the evaluation of the generated results.

5.2. Future Work

Besides the above mentioned limitations to be addressed, there are more starting points for future research. Pretraining has demonstrated to improve the generation results [92].

Regarding generative AI for visual modalities, Diffusion Models have shown stunning results, but are even more expensive computationally [92]. An extension of this is the Latent Diffusion Model, which promises to provide both, the high quality generation of targets and reduced computationalcomplexity compared to diffusion models [71]. Further, instead of using stable diffusion, flow matching — a more efficient method using neural ordinary differential equations — can be used to further increase efficiency of the more costly model [83].

Instead of predicting the diffusion tensor, one option would be to predict the T_2-weighted images used for the estimation of the diffusion tensor or the scalars extracted from the diffusion tensor directly as done in [5].

Finally, instead of predicting a single modality from another single modality, a more basic model can be constructed, that is able to do translations from various to various modalities. Strategies to achieve that can be found in [74, 63]. This would require a foundational model which is then finetuned to a variety of specifc generative tasks. In the recent years, transformers that work on image data using convolutions have been proposed and may be a good architecture for the foundation model [16, 32].

5.3. Conclusion

6. Acknowledgements

Rahel, Flo, Qi, Klaus, josh, max,

Bibliography

[1] *Augmentation*. en. URL: https://torchio.readthedocs.io/transforms/transforms/augmentation.html (visited on 05/12/2024).

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. *Layer Normalization*. July 2016. DOI: 10.48550/arXiv.1607.06450. URL: http://arxiv.org/abs/1607.06450 (visited on 05/09/2024).

[3] Matt A. Bernstein, Kevin F. King, and Xiaohong Joe Zhou. *Handbook of MRI Pulse Sequences*. en. Elsevier, Sept. 2004. ISBN: 978-0-08-053312-4.

[4] Oliver Bieri and Klaus Scheffler. "Fundamentals of balanced steady state free precession MRI". en. In: *Journal of Magnetic Resonance Imaging* 38.1 (2013), pp. 2–11. ISSN: 1522-2586. DOI: 10.1002/jmri.24163. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/jmri.24163 (visited on 04/10/2024).

[5] Florian Birk et al. "High-resolution neural network-driven mapping of multiple diffusion metrics leveraging asymmetries in the balanced steady-state free precession frequency profile". en. In: *NMR in Biomedicine* 35.6 (2022), e4669. ISSN: 1099-1492. DOI: 10.1002/nbm.4669. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/nbm.4669 (visited on 04/11/2024).

[6] Stevo Bozinovski. "Reminder of the First Paper on Transfer Learning in Neural Networks, 1976". en. In: *Informatica* 44.3 (Sept. 2020). ISSN: 1854-3871. DOI: 10.31449/inf.v44i3.2828. URL: https://www.informatica.si/index.php/informatica/article/view/2828 (visited on 04/10/2024).

[7] Matthew Brett et al. *nipy/nibabel: 5.2.1*. Feb. 2024. DOI: 10.5281/zenodo.10714563. URL: https://doi.org/10.5281/zenodo.10714563.

[8] Brian M. Dale, Mark A. Brown, and PhD„ Richard C. Semelka. "Relaxation". en. In: *MRI Basic Principles and Applications*. John Wiley & Sons, Ltd, 2015, pp. 17–25. ISBN: 978-1-119-01306-8. DOI: 10.1002/9781119013068.ch3. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119013068.ch3 (visited on 04/18/2024).

[9] Robert W. Brown et al. "Introduction to MRI Coils and Magnets". en. In: *Magnetic Resonance Imaging*. John Wiley & Sons, Ltd, 2014, pp. 823–857. ISBN: 978-1-118-63395-3. DOI: 10.1002/9781118633953.ch27. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118633953.ch27 (visited on 05/08/2024).

[10] Tian Cao et al. "Robust Multimodal Dictionary Learning". en. In: *Advanced Information Systems Engineering*. Ed. by David Hutchison et al. Vol. 7908. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 259–266. ISBN: 978-3-642-38708-1 978-3-642-38709-8. DOI: 10.1007/978-3-642-40811-3_33. URL: http://link.springer.com/10.1007/978-3-642-40811-3_33 (visited on 05/10/2024).

[11] M. Jorge Cardoso et al. *MONAI: An open-source framework for deep learning in healthcare*. Nov. 2022. DOI: 10.48550/arXiv.2211.02701. URL: http://arxiv.org/abs/2211.02701 (visited on 04/10/2024).

[12] Junyi Chai et al. “Deep learning in computer vision: A critical review of emerging techniques and application scenarios”. In: *Machine Learning with Applications* 6 (Dec. 2021), p. 100134. ISSN: 2666-8270. DOI: 10.1016/j.mlwa.2021.100134. URL: https://www.sciencedirect.com/science/article/pii/S2666827021000670 (visited on 05/14/2024).

[13] Sihong Chen, Kai Ma, and Yefeng Zheng. *Med3D: Transfer Learning for 3D Medical Image Analysis*. July 2019. DOI: 10.48550/arXiv.1904.00625. URL: http://arxiv.org/abs/1904.00625 (visited on 04/10/2024).

[14] Stuart Clare. “Functional MRI : Methods and Applications”. In: 1997. URL: https://api.semanticscholar.org/CorpusID:8441926.

[15] Dan Dale. *Fine-Tuning Scheduler*. Mar. 2024. DOI: 10.5281/zenodo.10780386. URL: https://zenodo.org/records/10780386 (visited on 04/10/2024).

[16] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. en. June 2021. URL: http://arxiv.org/abs/2010.11929 (visited on 05/14/2024).

[17] Vincent Dumoulin and Francesco Visin. *A guide to convolution arithmetic for deep learning*. en. Jan. 2018. URL: http://arxiv.org/abs/1603.07285 (visited on 05/09/2024).

[18] William A. Falcon. “Pytorch lightning”. In: *GitHub* 3 (2019). URL: https://cir.nii.ac.jp/crid/1370013168774120069 (visited on 04/10/2024).

[19] *File:Praezession.svg - Wikimedia Commons*. Apr. 23, 2024. URL: https://commons.wikimedia.org/wiki/File:Praezession.svg (visited on 05/07/2024).

[20] K. J. Friston et al. “Statistical parametric maps in functional imaging: A general linear approach”. en. In: *Human Brain Mapping* 2.4 (1994), pp. 189–210. ISSN: 1097-0193. DOI: 10.1002/hbm.460020402. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/hbm.460020402 (visited on 04/10/2024).

[21] Carl Ganter. “Steady state of gradient echo sequences with radiofrequency phase cycling: Analytical solution, contrast enhancement with partial spoiling”. en. In: *Magnetic Resonance in Medicine* 55.1 (2006), pp. 98–107. ISSN: 1522-2594. DOI: 10.1002/mrm.20736. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.20736 (visited on 05/08/2024).

[22] Eleftherios Garyfallidis et al. “Dipy, a library for the analysis of diffusion MRI data”. English. In: *Frontiers in Neuroinformatics* 8 (Feb. 2014). ISSN: 1662-5196. DOI: 10.3389/fninf.2014.00008. URL: https://www.frontiersin.org/articles/10.3389/fninf.2014.00008 (visited on 04/10/2024).

[23] R. Gencay and Min Qi. “Pricing and hedging derivative securities with neural networks: Bayesian regularization, early stopping, and bagging”. In: *IEEE Transactions on Neural Networks* 12.4 (July 2001), pp. 726–734. ISSN: 1941-0093. DOI: 10.1109/72.935086. URL: https://ieeexplore.ieee.org/abstract/document/935086 (visited on 05/10/2024).

[24] Hossein Gholamalinezhad and Hossein Khosravi. “Pooling Methods in Deep Neural Networks, a Review”. en. In: ().

[25] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks”. en. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. JMLR Workshop and Conference Proceedings, Mar. 2010, pp. 249–256. URL: https://proceedings.mlr.press/v9/glorot10a.html (visited on 05/09/2024).

[26] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. en. MIT Press, Nov. 2016. ISBN: 978-0-262-33737-3.

[27] Ian Goodfellow et al. “Generative adversarial networks”. In: *Communications of the ACM* 63.11 (Oct. 2020), pp. 139–144. ISSN: 0001-0782. DOI: 10.1145/3422622. URL: https://dl.acm.org/doi/10.1145/3422622 (visited on 05/09/2024).

[28] Krzysztof J. Gorgolewski et al. “The brain imaging data structure, a format for organizing and describing outputs of neuroimaging experiments”. en. In: *Scientific Data* 3.1 (June 2016), p. 160044. ISSN: 2052-4463. DOI: 10.1038/sdata.2016.44. URL: https://www.nature.com/articles/sdata201644 (visited on 05/12/2024).

[29] David Halliday, Robert Resnick, and Jearl Walker. *Fundamentals of Physics*. en. John Wiley & Sons, Aug. 2013. ISBN: 978-1-118-23071-8.

[30] Brian Hargreaves. “Rapid Gradient-Echo Imaging”. In: *Journal of magnetic resonance imaging : JMRI* 36.6 (Dec. 2012), pp. 1300–1313. ISSN: 1053-1807. DOI: 10.1002/jmri.23742. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3502662/ (visited on 05/08/2024).

[31] Charles R. Harris et al. “Array programming with NumPy”. en. In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. ISSN: 1476-4687. DOI: 10.1038/s41586-020-2649-2. URL: https://www.nature.com/articles/s41586-020-2649-2 (visited on 05/12/2024).

[32] Ali Hatamizadeh et al. *Swin UNETR: Swin Transformers for Semantic Segmentation of Brain Tumors in MRI Images*. Jan. 2022. DOI: 10.48550/arXiv.2201.01266. URL: http://arxiv.org/abs/2201.01266 (visited on 05/14/2024).

[33] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: 2016, pp. 770–778. URL: https://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html (visited on 05/09/2024).

[34] Kaiming He et al. “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification”. In: 2015, pp. 1026–1034. URL: https://openaccess.thecvf.com/content_iccv_2015/html/He_Delving_Deep_into_ICCV_2015_paper.html (visited on 05/09/2024).

[35] Johanna Helenius et al. “Diffusion-Weighted MR Imaging in Normal Human Brains in Various Age Groups”. en. In: *American Journal of Neuroradiology* 23.2 (Feb. 2002), pp. 194–199. ISSN: 0195-6108, 1936-959X. URL: https://www.ajnr.org/content/23/2/194 (visited on 05/08/2024).

[36] Rahel Heule, Carl Ganter, and Oliver Bieri. “Triple echo steady-state (TESS) relaxometry”. en. In: *Magnetic Resonance in Medicine* 71.1 (2014), pp. 230–237. ISSN: 1522-2594. DOI: 10.1002/mrm.24659. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.24659 (visited on 04/10/2024).

[37] Rahel Heule et al. “Multi-parametric artificial neural network fitting of phase-cycled balanced steady-state free precession data”. en. In: *Magnetic Resonance in Medicine* 84.6 (2020). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/mrm.28325, pp. 2981–2993. ISSN: 1522-2594. DOI: 10.1002/mrm.28325. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.28325 (visited on 05/16/2024).

[38] Yawen Huang, Ling Shao, and Alejandro F. Frangi. “Simultaneous Super-Resolution and Cross-Modality Synthesis of 3D Medical Images Using Weakly-Supervised Joint Convolutional Sparse Coding”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017, pp. 5787–5796. DOI: 10.1109/CVPR.2017.613. URL: https://ieeexplore.ieee.org/document/8100096 (visited on 05/10/2024).

[39] T.A.G.M. Huisman. “Diffusion-weighted and diffusion tensor imaging of the brain, made easy”. In: *Cancer Imaging* 10.1A (Oct. 2010), S163–S171. ISSN: 1740-5025. DOI: 10.1102/1470-7330.2010.9023. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2967146/ (visited on 04/10/2024).

[40] John D. Hunter. “Matplotlib: A 2D Graphics Environment”. In: *Computing in Science & Engineering* 9.3 (May 2007), pp. 90–95. ISSN: 1558-366X. DOI: 10.1109/MCSE.2007.55. URL: https://ieeexplore.ieee.org/document/4160265 (visited on 05/12/2024).

[41] Sergey Ioffe and Christian Szegedy. “Batch normalization: accelerating deep network training by reducing internal covariate shift”. In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML’15. Lille, France: JMLR.org, July 2015, pp. 448–456. (Visited on 05/09/2024).

[42] Phillip Isola et al. “Image-To-Image Translation With Conditional Adversarial Networks”. In: 2017, pp. 1125–1134. URL: https://openaccess.thecvf.com/content_cvpr_2017/html/Isola_Image-To-Image_Translation_With_CVPR_2017_paper.html (visited on 04/10/2024).

[43] Bernd André Jung and Matthias Weigel. “Spin echo magnetic resonance imaging”. en. In: *Journal of Magnetic Resonance Imaging* 37.4 (2013), pp. 805–817. ISSN: 1522-2586. DOI: 10.1002/jmri.24068. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/jmri.24068 (visited on 05/08/2024).

[44] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. en. Jan. 2017. URL: http://arxiv.org/abs/1412.6980 (visited on 05/10/2024).

[45] Diederik P. Kingma and Max Welling. *Auto-Encoding Variational Bayes*. en. Dec. 2022. URL: http://arxiv.org/abs/1312.6114 (visited on 05/10/2024).

[46] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems*. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html (visited on 05/09/2024).

[47] Thomas Küstner et al. “CINENet: deep learning-based 3D cardiac CINE MRI reconstruction with multi-coil complex-valued 4D spatio-temporal convolutions”. en. In: *Scientific Reports* 10.1 (Aug. 2020), p. 13710. ISSN: 2045-2322. DOI: 10.1038/s41598-020-70551-8. URL: https://www.nature.com/articles/s41598-020-70551-8 (visited on 05/15/2024).

[48] National High Magnetic Field Laboratory. *MRI: A Guided Tour - Magnet Academy*. May 7, 2024. URL: https://nationalmaglab.org/magnet-academy/read-science-stories/science-simplified/mri-a-guided-tour/ (visited on 05/07/2024).

[49] Yann LeCun et al. “Handwritten Digit Recognition with a Back-Propagation Network”. In: *Advances in Neural Information Processing Systems*. Vol. 2. Morgan-Kaufmann, 1989. URL: https://papers.nips.cc/paper/1989/hash/53c3bce66e43be4f209556518c2fcb54-Abstract.html (visited on 05/10/2024).

[50] Zhi-Pei Liang and Paul C. Lauterbur. *Principles of Magnetic Resonance Imaging*. SPIE Optical Engineering Press Bellingham, 2000. URL: https://cds.cern.ch/record/1480847/files/0780347234_TOC.pdf (visited on 04/17/2024).

[51] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. Jan. 2019. DOI: 10.48550/arXiv.1711.05101. URL: http://arxiv.org/abs/1711.05101 (visited on 04/10/2024).

[52] José P. Marques et al. “MP2RAGE, a self bias-field corrected sequence for improved segmentation and T1-mapping at high field”. In: *NeuroImage* 49.2 (Jan. 2010), pp. 1271–1281. ISSN: 1053-8119. DOI: 10.1016/j.neuroimage.2009.10.002. URL: https://www.sciencedirect.com/science/article/pii/S1053811909010738 (visited on 05/12/2024).

[53] Warren S. McCulloch and Walter Pitts. "A logical calculus of the ideas immanent in nervous activity". en. In: *The bulletin of mathematical biophysics* 5.4 (Dec. 1943), pp. 115–133. ISSN: 1522-9602. DOI: 10.1007/BF02478259. URL: https://doi.org/10.1007/BF02478259 (visited on 05/09/2024).

[54] Karla L. Miller. "Asymmetries of the balanced SSFP profile. Part I: Theory and observation". en. In: *Magnetic Resonance in Medicine* 63.2 (2010), pp. 385–395. ISSN: 1522-2594. DOI: 10.1002/mrm.22212. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.22212 (visited on 05/14/2024).

[55] Karla L. Miller, Stephen M. Smith, and Peter Jezzard. "Asymmetries of the balanced SSFP profile. Part II: White matter". en. In: *Magnetic Resonance in Medicine* 63.2 (2010), pp. 396–406. ISSN: 1522-2594. DOI: 10.1002/mrm.22249. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.22249 (visited on 05/14/2024).

[56] Vinod Nair and Geoffrey E. Hinton. "Rectified linear units improve restricted boltzmann machines". In: *Proceedings of the 27th International Conference on International Conference on Machine Learning*. ICML'10. Madison, WI, USA: Omnipress, June 2010, pp. 807–814. ISBN: 978-1-60558-907-7. (Visited on 05/09/2024).

[57] Damien Nguyen and Oliver Bieri. "Motion-insensitive rapid configuration relaxometry". en. In: *Magnetic Resonance in Medicine* 78.2 (2017), pp. 518–526. ISSN: 1522-2594. DOI: 10.1002/mrm.26384. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.26384 (visited on 04/10/2024).

[58] Dwight Nishimura. *Principles of Magnetic Resonance Imaging*. English. Stanford Univ, Feb. 2010.

[59] Yingxue Pang et al. *Image-to-Image Translation: Methods and Applications*. en. July 2021. URL: http://arxiv.org/abs/2101.08629 (visited on 05/10/2024).

[60] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html (visited on 04/10/2024).

[61] Fernando Pérez-García, Rachel Sparks, and Sébastien Ourselin. "TorchIO: A Python library for efficient loading, preprocessing, augmentation and patch-based sampling of medical images in deep learning". In: *Computer Methods and Programs in Biomedicine* 208 (Sept. 2021), p. 106236. ISSN: 0169-2607. DOI: 10.1016/j.cmpb.2021.106236. URL: https://www.sciencedirect.com/science/article/pii/S0169260721003102 (visited on 04/10/2024).

[62] Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. "To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks". In: *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*. Ed. by Isabelle Augenstein et al. Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 7–14. DOI: 10.18653/v1/W19-4302. URL: https://aclanthology.org/W19-4302 (visited on 04/10/2024).

[63] Jonas Pfeiffer et al. *Modular Deep Learning*. Jan. 2024. DOI: 10.48550/arXiv.2302.11529. URL: http://arxiv.org/abs/2302.11529 (visited on 04/10/2024).

[64] Donald B. Plewes and Walter Kucharczyk. "Physics of MRI: A primer". en. In: *Journal of Magnetic Resonance Imaging* 35.5 (2012), pp. 1038–1054. ISSN: 1522-2586. DOI: 10.1002/jmri.23642. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/jmri.23642 (visited on 04/10/2024).

[65] Alec Radford et al. "Improving Language Understanding by Generative Pre-Training". en. In: ().

[66] Kamisetty Ramam Rao and Patrick C. Yip. *The Transform and Data Compression Handbook*. en. CRC Press, Oct. 2018. ISBN: 978-1-4200-3738-8.

[67] Garvesh Raskutti, Martin J. Wainwright, and Bin Yu. *Early stopping and non-parametric regression: An optimal data-dependent stopping rule*. en. June 2013. URL: http://arxiv.org/abs/1306.3574 (visited on 05/10/2024).

[68] Jeff Reback et al. "pandas-dev/pandas: Pandas 1.0.5". In: *Zenodo* (June 2020). DOI: 10.5281/zenodo.3898987. URL: https://ui.adsabs.harvard.edu/abs/2020zndo...3898987R (visited on 05/12/2024).

[69] *Relaxation longitudinal magnetization - Spinlattice relaxation - Wikipedia*. May 1, 2024. URL: https://en.wikipedia.org/wiki/Spin%E2%80%93lattice_relaxation#/media/File:Relaxation_longitudinal_magnetization.svg (visited on 05/07/2024).

[70] *Relaxation transverse magnetization - Spinspin relaxation - Wikipedia*. May 1, 2024. URL: https://en.wikipedia.org/wiki/Spin%E2%80%93spin_relaxation#/media/File:Relaxation_transverse_magnetization.svg (visited on 05/07/2024).

[71] Robin Rombach et al. "High-Resolution Image Synthesis with Latent Diffusion Models". en. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, June 2022, pp. 10674–10685. ISBN: 978-1-66546-946-3. DOI: 10.1109/CVPR52688.2022.01042. URL: https://ieeexplore.ieee.org/document/9878449/ (visited on 05/10/2024).

[72] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". en. 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. DOI: 10.1007/978-3-319-24574-4_28.

[73] F. Rosenblatt. "The perceptron: a probabilistic model for information storage and organization in the brain". eng. In: *Psychological Review* 65.6 (Nov. 1958), pp. 386–408. ISSN: 0033-295X. DOI: 10.1037/h0042519.

[74] Sebastian Ruder. *An Overview of Multi-Task Learning in Deep Neural Networks*. June 2017. DOI: 10.48550/arXiv.1706.05098. URL: http://arxiv.org/abs/1706.05098 (visited on 04/10/2024).

[75] Klaus Scheffler and Stefan Lehnhardt. "Principles and applications of balanced SSFP techniques". en. In: *European Radiology* 13.11 (Nov. 2003), pp. 2409–2418. ISSN: 1432-1084. DOI: 10.1007/s00330-003-1957-x. URL: https://doi.org/10.1007/s00330-003-1957-x (visited on 04/10/2024).

[76] *Segmentation of neuronal structures in EM stacks challenge - ISBI 2012*. URL: https://imagej.github.io/events/isbi-2012-segmentation-challenge (visited on 05/10/2024).

[77] R. N. Sener. "Diffusion MRI: apparent diffusion coefficient (ADC) values in the normal brain and a classification of brain disorders based on ADC values". In: *Computerized Medical Imaging and Graphics* 25.4 (July 2001), pp. 299–326. ISSN: 0895-6111. DOI: 10.1016/S0895-6111(00)00083-5. URL: https://www.sciencedirect.com/science/article/pii/S0895611100000835 (visited on 05/08/2024).

[78] Alex Sherstinsky. "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network". In: *Physica D: Nonlinear Phenomena* 404 (Mar. 2020), p. 132306. ISSN: 0167-2789. DOI: 10.1016/j.physd.2019.132306. URL: https://www.sciencedirect.com/science/article/pii/S0167278919305974 (visited on 05/15/2024).

[79] Stephen M. Smith et al. “Advances in functional and structural MR image analysis and implementation as FSL”. In: *NeuroImage*. Mathematics in Brain Imaging 23 (Jan. 2004), S208–S219. ISSN: 1053-8119. DOI: 10.1016/j.neuroimage.2004.07.051. URL: https://www.sciencedirect.com/science/article/pii/S1053811904003933 (visited on 04/10/2024).

[80] Jascha Sohl-Dickstein et al. “Deep unsupervised learning using nonequilibrium thermodynamics”. In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML’15. Lille, France: JMLR.org, July 2015, pp. 2256–2265. (Visited on 05/10/2024).

[81] Sho Sonoda and Noboru Murata. “Neural network with unbounded activation functions is universal approximator”. In: *Applied and Computational Harmonic Analysis* 43.2 (Sept. 2017), pp. 233–268. ISSN: 1063-5203. DOI: 10.1016/j.acha.2015.12.005. URL: https://www.sciencedirect.com/science/article/pii/S1063520315001748 (visited on 05/09/2024).

[82] Nitish Srivastava et al. “Dropout: A Simple Way to Prevent Neural Networks from Overfitting”. In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. ISSN: 1533-7928. URL: http://jmlr.org/papers/v15/srivastava14a.html (visited on 05/09/2024).

[83] Alexander Tong et al. *Improving and generalizing flow-based generative models with minibatch optimal transport*. en. Mar. 2024. URL: http://arxiv.org/abs/2302.00482 (visited on 05/14/2024).

[84] H. C. Torrey. “Bloch Equations with Diffusion Terms”. In: *Physical Review* 104.3 (Nov. 1956), pp. 563–565. DOI: 10.1103/PhysRev.104.563. URL: https://link.aps.org/doi/10.1103/PhysRev.104.563 (visited on 05/08/2024).

[85] Jacques-Donald Tournier, Susumu Mori, and Alexander Leemans. “Diffusion Tensor Imaging and Beyond”. In: *Magnetic Resonance in Medicine* 65.6 (June 2011), pp. 1532–1556. ISSN: 0740-3194. DOI: 10.1002/mrm.22924. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3366862/ (visited on 04/10/2024).

[86] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. *Instance Normalization: The Missing Ingredient for Fast Stylization*. Nov. 2017. DOI: 10.48550/arXiv.1607.08022. URL: http://arxiv.org/abs/1607.08022 (visited on 05/09/2024).

[87] C. Van Der Malsburg. “Frank Rosenblatt: Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms”. en. In: *Brain Theory*. Ed. by Günther Palm and Ad Aertsen. Berlin, Heidelberg: Springer, 1986, pp. 245–248. ISBN: 978-3-642-70911-1. DOI: 10.1007/978-3-642-70911-1_20.

[88] K. Van Leemput et al. “Automated model-based tissue classification of MR images of the brain”. In: *IEEE Transactions on Medical Imaging* 18.10 (Oct. 1999), pp. 897–908. ISSN: 1558-254X. DOI: 10.1109/42.811270. URL: https://ieeexplore.ieee.org/abstract/document/811270 (visited on 05/12/2024).

[89] Hien Van Nguyen, Kevin Zhou, and Raviteja Vemulapalli. “Cross-Domain Synthesis of Medical Images Using Efficient Location-Sensitive Deep Network”. en. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab et al. Cham: Springer International Publishing, 2015, pp. 677–684. ISBN: 978-3-319-24553-9. DOI: 10.1007/978-3-319-24553-9_83.

[90] Ashish Vaswani et al. “Attention is all you need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS’17. Red Hook, NY, USA: Curran Associates Inc., Dec. 2017, pp. 6000–6010. ISBN: 978-1-5108-6096-4. (Visited on 05/09/2024).

[91] Raviteja Vemulapalli, Hien Van Nguyen, and Shaohua Kevin Zhou. "Unsupervised Cross-Modal Synthesis of Subject-Specific Scans". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. Dec. 2015, pp. 630–638. DOI: 10.1109/ICCV.2015.79. URL: https://ieeexplore.ieee.org/document/7410436 (visited on 05/10/2024).

[92] Tengfei Wang et al. *Pretraining is All You Need for Image-to-Image Translation*. May 2022. DOI: 10.48550/arXiv.2205.12952. URL: http://arxiv.org/abs/2205.12952 (visited on 04/10/2024).

[93] Michael L. Waskom. "seaborn: statistical data visualization". en. In: *Journal of Open Source Software* 6.60 (Apr. 2021), p. 3021. ISSN: 2475-9066. DOI: 10.21105/joss.03021. URL: https://joss.theoj.org/papers/10.21105/joss.03021 (visited on 05/12/2024).

[94] Colin White et al. *Neural Architecture Search: Insights from 1000 Papers*. en. Jan. 2023. URL: http://arxiv.org/abs/2301.08727 (visited on 05/14/2024).

[95] Yuxin Wu and Kaiming He. "Group Normalization". In: 2018, pp. 3–19. URL: https://openaccess.thecvf.com/content_ECCV_2018/html/Yuxin_Wu_Group_Normalization_ECCV_2018_paper.html (visited on 05/09/2024).

[96] Qianye Yang et al. "MRI Cross-Modality Image-to-Image Translation". en. In: *Scientific Reports* 10.1 (Feb. 2020), p. 3753. ISSN: 2045-2322. DOI: 10.1038/s41598-020-60520-6. URL: https://www.nature.com/articles/s41598-020-60520-6 (visited on 04/10/2024).

[97] Tal Yarkoni et al. "PyBIDS: Python tools for BIDS datasets". In: *Journal of open source software* 4.40 (2019), p. 1294. ISSN: 2475-9066. DOI: 10.21105/joss.01294. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7409983/ (visited on 05/12/2024).

[98] Richard Zhang et al. *The Unreasonable Effectiveness of Deep Features as a Perceptual Metric*. Apr. 2018. DOI: 10.48550/arXiv.1801.03924. URL: http://arxiv.org/abs/1801.03924 (visited on 04/10/2024).

[99] Zhou and Chellappa. "Computation of optical flow using a neural network". In: *IEEE 1988 International Conference on Neural Networks*. July 1988, 71–78 vol.2. DOI: 10.1109/ICNN.1988.23914. URL: https://ieeexplore.ieee.org/document/23914 (visited on 05/09/2024).

[100] Marcel Peter Zwiers, Stefano Moia, and Robert Oostenveld. "BIDScoin: A User-Friendly Application to Convert Source Data to Brain Imaging Data Structure". English. In: *Frontiers in Neuroinformatics* 15 (Jan. 2022). ISSN: 1662-5196. DOI: 10.3389/fninf.2021.770608. URL: https://www.frontiersin.org/articles/10.3389/fninf.2021.770608 (visited on 05/12/2024).

Appendix

A. Supplementary Material

Figure .1. *The architecture of the model used in this work. Please zoom in to explore in detail.*

Detailed Model Architecture

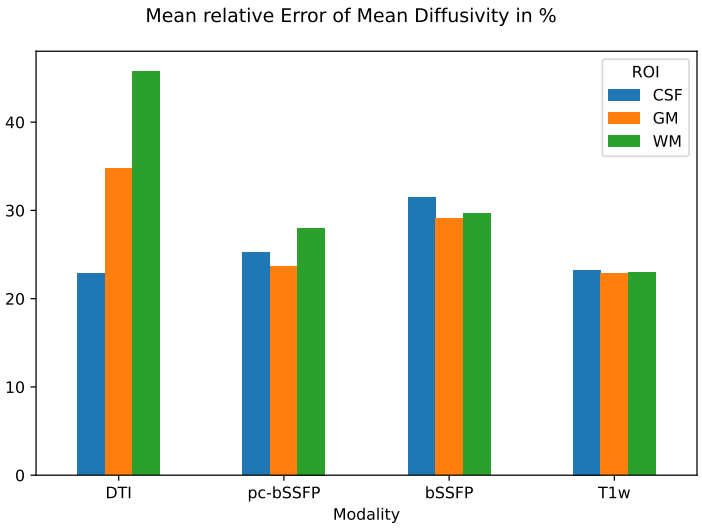

Figure .2. *md*

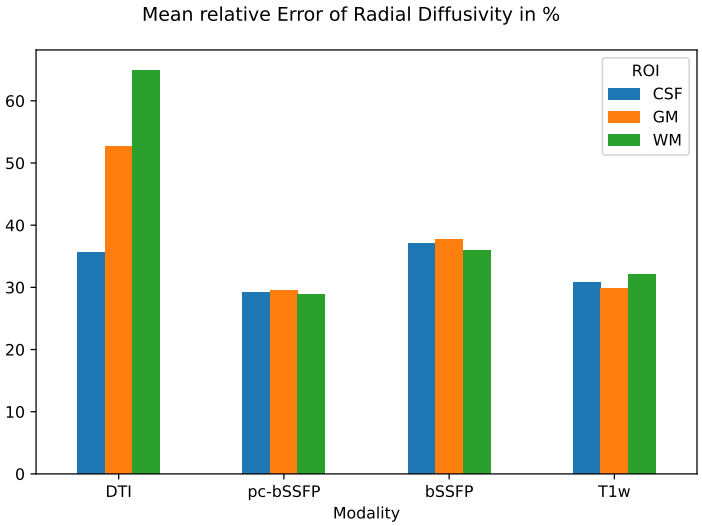

Figure .3. *rd*

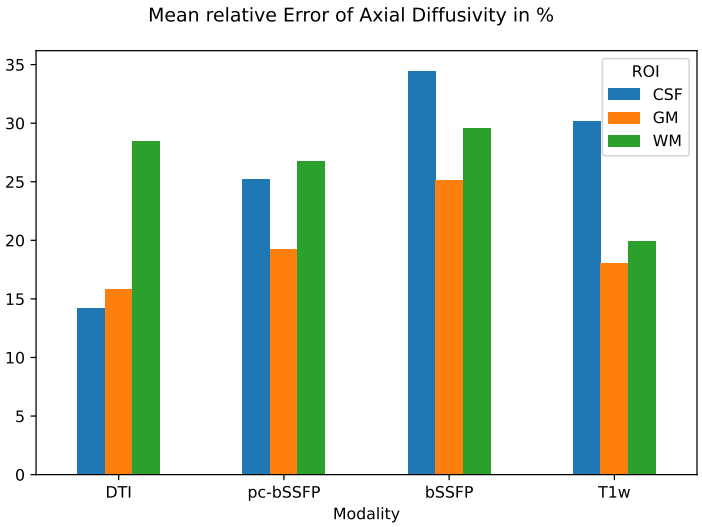

Figure .4. *ad*

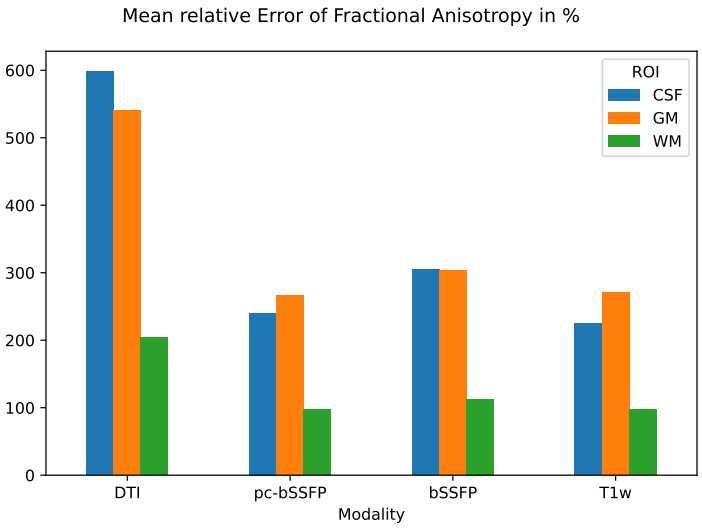

Figure .5. *af*

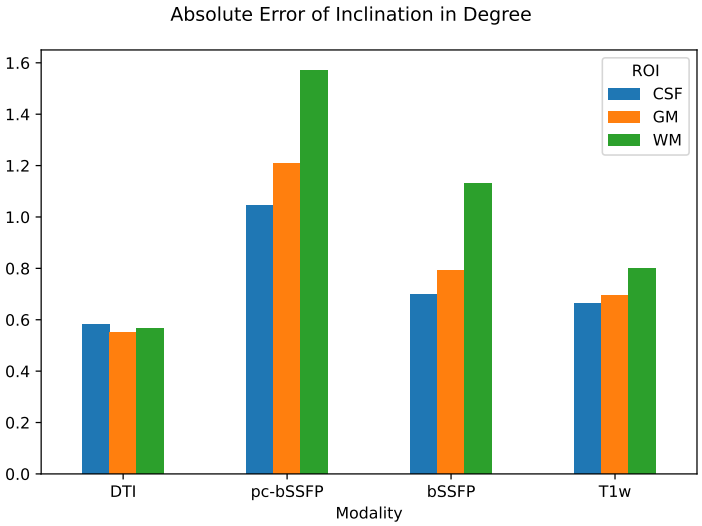

Figure .6. *incl*

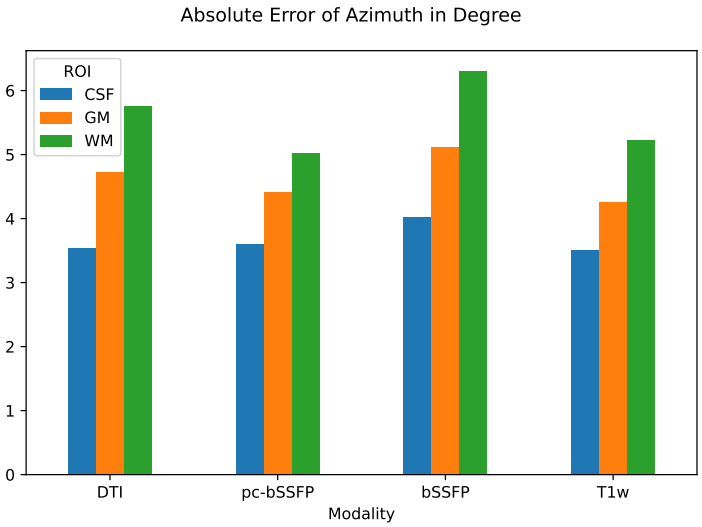

Figure .7. *azi*

B. Direct Training: Qualitative Plots for all Scalars

C. Multi-Stage Training: Quantitative Plots for all Tensor Elements and Scalars

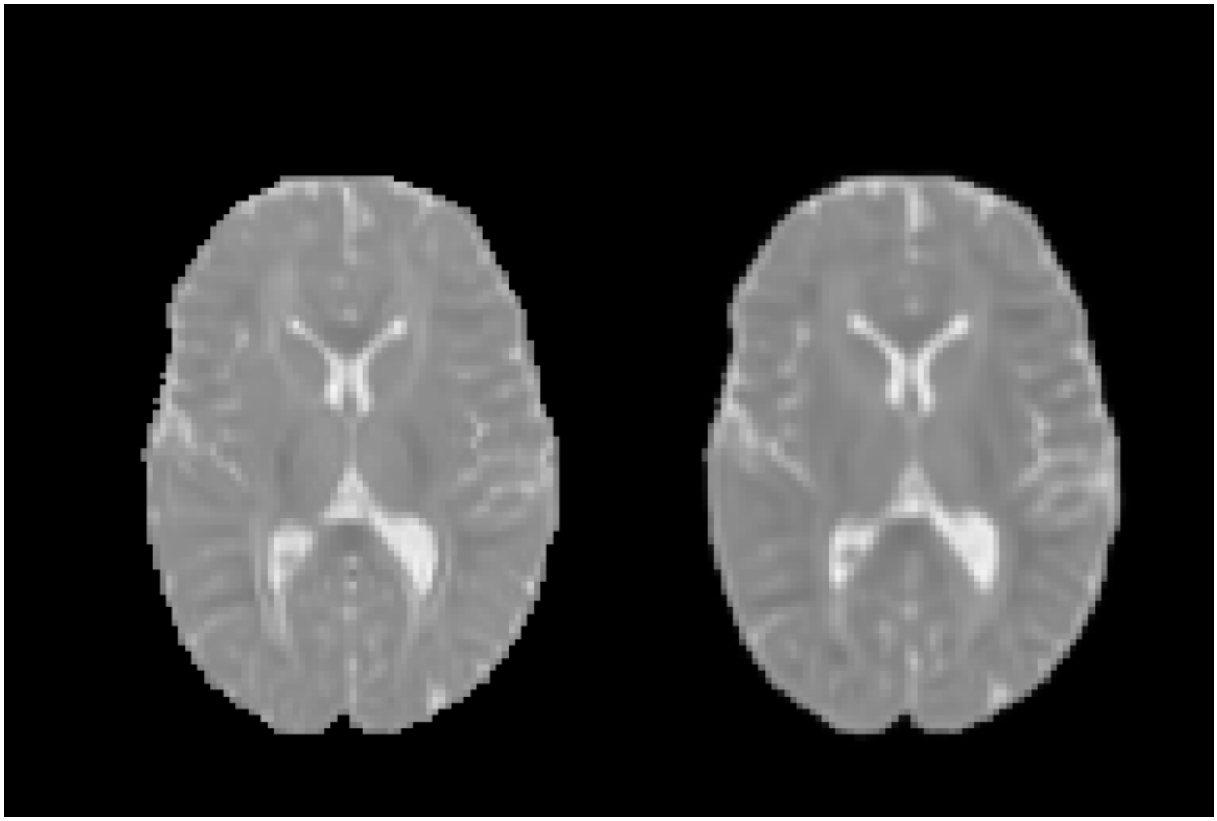

Figure .8. *Comparison between D_{yy} of original DT and predicted DT using pc-bSSFP as input modality.*

D. Direct Training Evaluation Statistics

Modality	ROI	Scalar	Median	1%	25%	75%	99%
bssfp	CSF	dxx	22.98	6.05	8.52	28.17	38.05
dwi-tensor	CSF	dxx	9.76	4.64	5.79	27.18	38.67
pc-bssfp	CSF	dxx	21.07	5.02	6.71	26.49	32.75
t1w	CSF	dxx	17.18	4.25	7.23	24.62	83.23
bssfp	GM	dxx	22.76	5.04	6.77	27.18	46.05
dwi-tensor	GM	dxx	10.67	6.43	7.76	38.13	54.55
pc-bssfp	GM	dxx	22.24	4.68	5.58	25.06	42.99

t1w	GM	dxx	16.62	3.40	5.74	23.03	47.92
bssfp	WM	dxx	9.88	4.61	5.82	26.12	44.61
dwi-tensor	WM	dxx	11.29	7.55	8.63	44.94	56.83
pc-bssfp	WM	dxx	24.19	4.92	5.56	26.18	28.64
t1w	WM	dxx	18.81	3.83	5.62	25.80	47.36
bssfp	CSF	dxy	12.02	3.36	6.29	2988.29	6577.62
dwi-tensor	CSF	dxy	9.02	5.88	7.73	5589.47	10 579.18
pc-bssfp	CSF	dxy	2512.40	4.31	5.78	3393.45	6022.24
t1w	CSF	dxy	1894.97	3.54	6.74	3074.58	8235.55
bssfp	GM	dxy	10.48	3.52	6.79	3365.76	9672.37
dwi-tensor	GM	dxy	9.70	7.50	8.48	5653.00	10 503.94
pc-bssfp	GM	dxy	2531.02	5.26	6.22	3620.04	7964.46
t1w	GM	dxy	2153.02	3.89	7.02	3887.60	9851.58
bssfp	WM	dxy	9.11	5.16	7.39	1822.72	5892.20
dwi-tensor	WM	dxy	10.13	8.05	9.26	2494.32	9002.88
pc-bssfp	WM	dxy	1529.50	6.80	7.39	1871.34	3031.21
t1w	WM	dxy	1209.74	4.65	7.04	1647.99	6134.25
bssfp	CSF	dxz	17.81	4.99	8.54	4917.17	9621.07
dwi-tensor	CSF	dxz	10.69	7.59	9.42	6687.19	19 404.34
pc-bssfp	CSF	dxz	2527.38	4.10	5.79	3780.94	6709.16
t1w	CSF	dxz	1885.15	3.71	6.63	3885.35	8265.64
bssfp	GM	dxz	12.54	5.26	8.48	4545.06	7355.37
dwi-tensor	GM	dxz	11.55	9.22	10.24	6651.75	14 392.92
pc-bssfp	GM	dxz	2687.94	4.96	6.04	3657.43	5151.54
t1w	GM	dxz	2243.68	3.99	6.99	3457.69	10 889.91
bssfp	WM	dxz	10.91	6.49	8.76	2169.77	7522.11

dwi-tensor	WM	dxz	12.26	10.03	11.16	2904.39	8825.20
pc-bssfp	WM	dxz	1320.54	6.54	7.16	1748.27	5851.08
t1w	WM	dxz	1025.22	4.92	6.69	1616.59	7163.57
bssfp	CSF	dyy	16.74	6.22	8.72	28.72	42.27
dwi-tensor	CSF	dyy	8.55	3.57	4.64	20.45	36.34
pc-bssfp	CSF	dyy	20.33	5.09	6.78	26.51	43.43
t1w	CSF	dyy	17.19	4.47	7.29	24.99	33.96
bssfp	GM	dyy	10.18	5.30	6.80	28.19	41.68
dwi-tensor	GM	dyy	10.16	5.11	6.42	31.60	47.75
pc-bssfp	GM	dyy	21.37	4.52	5.44	24.06	30.48
t1w	GM	dyy	16.19	3.55	5.67	23.14	38.13
bssfp	WM	dyy	11.19	4.91	7.02	30.34	47.48
dwi-tensor	WM	dyy	11.70	7.35	8.53	41.62	55.84
pc-bssfp	WM	dyy	27.33	5.72	6.28	28.81	35.01
t1w	WM	dyy	17.11	3.47	5.49	24.01	45.41
bssfp	CSF	dyz	10.04	3.36	5.72	2759.52	5590.09
dwi-tensor	CSF	dyz	11.01	8.03	9.83	6717.22	10 275.61
pc-bssfp	CSF	dyz	2547.44	4.47	5.87	3865.43	6699.57
t1w	CSF	dyz	1959.50	3.32	6.70	3085.37	8316.22
bssfp	GM	dyz	8.30	3.17	5.77	3077.10	6232.45
dwi-tensor	GM	dyz	11.87	9.73	10.63	7039.82	23 977.82
pc-bssfp	GM	dyz	3033.35	5.35	6.45	3880.19	7209.39
t1w	GM	dyz	2181.29	4.03	6.86	3154.72	6403.68
bssfp	WM	dyz	7.94	4.48	6.41	1403.89	4991.09
dwi-tensor	WM	dyz	12.13	10.37	11.26	2580.10	8085.27

pc-bssfp	WM	dyz	1385.95	6.44	7.11	1852.83	4389.51
t1w	WM	dyz	1069.09	5.24	6.83	1469.13	7113.53
bssfp	CSF	dzz	23.69	6.36	8.90	30.99	68.43
dwi-tensor	CSF	dzz	8.80	3.65	5.05	22.15	38.87
pc-bssfp	CSF	dzz	20.80	4.96	6.48	26.65	92.00
t1w	CSF	dzz	17.30	4.28	6.88	24.59	87.07
bssfp	GM	dzz	23.15	5.28	7.28	30.86	44.64
dwi-tensor	GM	dzz	9.92	5.11	6.77	33.41	47.68
pc-bssfp	GM	dzz	21.74	4.63	5.39	23.69	34.25
t1w	GM	dzz	15.26	3.16	5.28	24.46	42.76
bssfp	WM	dzz	11.62	5.09	8.13	31.89	50.08
dwi-tensor	WM	dzz	11.51	7.27	8.79	42.04	56.14
pc-bssfp	WM	dzz	27.25	5.61	6.37	29.44	32.42
t1w	WM	dzz	16.36	3.15	5.55	24.35	45.72
bssfp	CSF	fa	304.67	153.60	225.60	402.28	568.62
dwi-tensor	CSF	fa	598.26	413.11	509.66	668.20	744.90
pc-bssfp	CSF	fa	240.16	187.28	214.12	288.09	326.35
t1w	CSF	fa	224.36	139.36	174.00	336.87	524.00
bssfp	GM	fa	303.00	101.05	227.91	410.36	548.41
dwi-tensor	GM	fa	540.10	432.63	496.93	607.78	672.01
pc-bssfp	GM	fa	267.23	210.45	258.03	280.47	308.74
t1w	GM	fa	271.24	129.89	177.61	380.76	514.21
bssfp	WM	fa	112.91	45.38	77.77	151.16	190.04
dwi-tensor	WM	fa	203.60	157.76	188.85	221.82	236.35
pc-bssfp	WM	fa	97.05	83.41	90.23	101.19	112.41
t1w	WM	fa	97.40	54.79	70.54	140.32	189.99

bssfp	CSF	md	31.44	22.15	27.48	34.32	56.60
dwi-tensor	CSF	md	22.88	15.46	19.82	31.97	38.37
pc-bssfp	CSF	md	25.25	19.05	22.92	28.70	51.44
t1w	CSF	md	23.23	14.34	19.50	27.42	47.33
bssfp	GM	md	29.08	21.27	26.14	34.12	49.95
dwi-tensor	GM	md	34.74	27.15	32.16	44.55	49.28
pc-bssfp	GM	md	23.70	20.54	22.27	25.23	36.65
t1w	GM	md	22.87	13.10	17.49	27.76	41.82
bssfp	WM	md	29.66	15.37	25.14	41.48	53.43
dwi-tensor	WM	md	45.75	37.78	42.15	52.41	59.23
pc-bssfp	WM	md	27.99	24.14	26.55	28.50	32.69
t1w	WM	md	22.95	13.16	17.68	31.95	45.46
bssfp	CSF	ad	34.47	21.48	27.[illegible]	[illegible]	862.30
dwi-tensor	CSF	ad	14.23	9.08	[illegible]	[illegible]	366.00
pc-bssfp	CSF	ad	25.20	15.27	[illegible]	[illegible]	505.60
t1w	CSF	ad	30.17	11.66	[illegible]	[illegible]	936.00
bssfp	GM	ad	25.16	18.22	[illegible]	[illegible]	108.10
dwi-tensor	GM	ad	15.81	10.19	[illegible]	[illegible]	148.70
pc-bssfp	GM	ad	19.21	14.37	[illegible]	[illegible]	105.90
t1w	GM	ad	18.02	9.88	[illegible]	[illegible]	452.00
bssfp	WM	ad	29.57	19.56	[illegible]	[illegible]	427.10
dwi-tensor	WM	ad	28.50	18.70	[illegible]	[illegible]	005.16
pc-bssfp	WM	ad	26.75	24.44	[illegible]	[illegible]	650.70
t1w	WM	ad	19.96	11.28	[illegible]	[illegible]	666.30
bssfp	CSF	rd	37.06	22.74	31.72	43.60	84.34

dwi-tensor	CSF	rd	35.62	25.81	31.12	47.33	62.15
pc-bssfp	CSF	rd	29.26	22.27	26.82	35.09	82.49
t1w	CSF	rd	30.80	17.62	22.59	36.11	81.62
bssfp	GM	rd	37.69	24.64	31.58	45.48	62.04
dwi-tensor	GM	rd	52.73	43.00	49.38	62.24	71.21
pc-bssfp	GM	rd	29.58	26.10	27.76	32.75	42.02
t1w	GM	rd	29.83	18.05	21.99	40.21	53.86
bssfp	WM	rd	35.95	22.47	28.88	49.16	66.13
dwi-tensor	WM	rd	64.93	53.66	58.07	72.77	83.83
pc-bssfp	WM	rd	28.85	24.94	27.96	30.29	34.15
t1w	WM	rd	32.01	20.07	21.88	47.66	67.58
bssfp	CSF	inclination	69.76	53.42	64.62	74.65	88.60
dwi-tensor	CSF	inclination	58.27	52.14	56.09	60.68	70.92
pc-bssfp	CSF	inclination	104.32	83.45	95.61	108.58	124.36
t1w	CSF	inclination	66.27	54.60	60.53	69.86	74.47
bssfp	GM	inclination	79.20	59.02	72.38	95.13	136.60
dwi-tensor	GM	inclination	54.97	49.90	51.52	57.64	68.82
pc-bssfp	GM	inclination	120.78	88.26	107.50	135.08	155.96
t1w	GM	inclination	69.56	58.11	63.90	75.41	102.79
bssfp	WM	inclination	112.87	66.92	80.07	130.67	173.03
dwi-tensor	WM	inclination	56.82	53.54	56.15	61.55	65.71
pc-bssfp	WM	inclination	157.15	82.10	129.35	169.95	184.30
t1w	WM	inclination	80.02	59.32	74.87	85.66	106.67
bssfp	CSF	azimuth	403.04	291.39	353.79	483.60	2062.73
dwi-tensor	CSF	azimuth	353.69	258.31	315.11	409.58	1468.24
pc-bssfp	CSF	azimuth	360.32	273.79	329.44	406.40	1416.85

t1w	CSF	azimuth	351.25	262.37	315.10	406.34	1386.04
bssfp	GM	azimuth	511.23	281.30	390.24	605.82	1684.69
dwi-tensor	GM	azimuth	473.36	337.85	392.82	549.41	1469.08
pc-bssfp	GM	azimuth	442.01	315.66	383.72	518.75	1460.95
t1w	GM	azimuth	426.05	325.85	392.90	527.27	1605.71
bssfp	WM	azimuth	630.79	268.55	445.85	736.33	1382.23
dwi-tensor	WM	azimuth	575.90	416.73	496.42	674.39	3397.96
pc-bssfp	WM	azimuth	502.32	335.52	445.79	589.29	2383.49
t1w	WM	azimuth	523.02	360.03	442.02	660.57	3689.50

E. Fine-Tuning Evaluation Statistics

Modality	**ROI**	**Scalar**	**Median**	1%	25%	75%	99%
bssfp	CSF	dxx	11.90	3.68	5.62	15.63	25.94
dwi-tensor	CSF	dxx	4.39	1.75	2.21	6.04	10.06
pc-bssfp	CSF	dxx	11.48	3.61	6.33	18.90	27.53
t1w	CSF	dxx	9.89	3.32	4.95	16.00	26.04
bssfp	GM	dxx	9.79	2.55	3.89	12.92	25.71
dwi-tensor	GM	dxx	4.75	1.82	2.13	7.33	13.53
pc-bssfp	GM	dxx	8.70	2.93	4.25	13.01	22.61
t1w	GM	dxx	7.52	2.19	3.42	10.52	25.93
bssfp	WM	dxx	8.29	2.34	3.49	12.07	32.16
dwi-tensor	WM	dxx	5.94	1.83	2.68	10.05	18.86
pc-bssfp	WM	dxx	6.57	2.25	3.17	9.88	13.92
t1w	WM	dxx	7.80	2.25	3.52	11.10	30.00

bssfp	CSF	dxy	29.65	4.07	5.73	41.81	77.24
dwi-tensor	CSF	dxy	9.15	1.91	2.53	18.72	65.81
pc-bssfp	CSF	dxy	22.76	4.13	6.33	46.52	77.43
t1w	CSF	dxy	25.01	3.22	4.65	35.61	69.91
bssfp	GM	dxy	37.63	4.09	5.29	47.72	152.80
dwi-tensor	GM	dxy	9.89	1.80	2.32	23.45	189.53
pc-bssfp	GM	dxy	23.02	4.20	6.22	49.83	219.69
t1w	GM	dxy	27.54	3.31	4.25	44.26	431.51
bssfp	WM	dxy	111.34	5.34	6.39	163.32	415.98
dwi-tensor	WM	dxy	25.52	1.95	2.83	74.82	389.93
pc-bssfp	WM	dxy	65.48	5.18	7.53	184.63	522.54
t1w	WM	dxy	87.04	4.35	5.18	158.24	727.03
bssfp	CSF	dxz	32.86	4.09	6.37	42.19	89.31
dwi-tensor	CSF	dxz	7.98	1.80	2.25	17.42	58.40
pc-bssfp	CSF	dxz	22.17	4.15	6.14	44.39	117.73
t1w	CSF	dxz	23.41	3.01	4.98	38.01	189.80
bssfp	GM	dxz	37.27	4.38	5.58	44.89	82.24
dwi-tensor	GM	dxz	8.99	1.76	2.16	19.43	67.76
pc-bssfp	GM	dxz	21.54	4.23	5.72	45.50	71.65
t1w	GM	dxz	24.55	3.16	4.16	45.51	108.77
bssfp	WM	dxz	119.28	5.41	6.62	156.36	5331.02
dwi-tensor	WM	dxz	20.12	1.87	2.56	60.22	2741.50
pc-bssfp	WM	dxz	54.23	5.46	7.32	138.93	372.35
t1w	WM	dxz	81.95	4.30	5.01	129.99	493.32
bssfp	CSF	dyy	12.63	3.62	5.60	16.13	22.53
dwi-tensor	CSF	dyy	4.75	1.62	2.26	7.00	11.79

pc-bssfp	CSF	dyy	10.50	3.66	6.62	18.93	27.38
t1w	CSF	dyy	10.14	3.53	5.13	15.01	23.46
bssfp	GM	dyy	9.57	2.65	4.00	13.41	18.83
dwi-tensor	GM	dyy	4.65	1.53	2.15	6.36	13.46
pc-bssfp	GM	dyy	7.65	3.10	4.61	13.50	23.03
t1w	GM	dyy	7.75	2.21	3.53	9.72	20.06
bssfp	WM	dyy	9.04	2.49	3.81	13.40	23.24
dwi-tensor	WM	dyy	5.29	1.66	2.00	6.87	16.17
pc-bssfp	WM	dyy	8.25	2.55	3.90	11.72	15.39
t1w	WM	dyy	7.44	2.14	2.67	9.03	19.89
bssfp	CSF	dyz	29.42	3.89	5.52	40.83	179.10
dwi-tensor	CSF	dyz	8.52	1.75	2.34	23.35	51.38
pc-bssfp	CSF	dyz	19.95	4.09	6.21	46.26	198.39
t1w	CSF	dyz	26.98	3.01	4.72	45.24	212.14
bssfp	GM	dyz	34.82	4.02	5.37	52.14	121.45
dwi-tensor	GM	dyz	8.74	1.70	2.11	23.52	54.22
pc-bssfp	GM	dyz	20.41	4.37	5.82	49.33	105.23
t1w	GM	dyz	25.93	3.25	4.23	47.45	374.28
bssfp	WM	dyz	110.38	5.34	6.85	164.82	398.97
dwi-tensor	WM	dyz	19.91	2.07	2.66	67.16	207.49
pc-bssfp	WM	dyz	47.65	5.43	8.34	145.83	539.81
t1w	WM	dyz	78.07	4.81	5.70	138.22	425.40
bssfp	CSF	dzz	12.32	3.89	5.69	17.09	58.93
dwi-tensor	CSF	dzz	4.37	1.63	2.40	6.87	11.94
pc-bssfp	CSF	dzz	9.70	3.71	6.06	19.50	60.48

t1w	CSF	dzz	9.63	3.22	5.14	17.32	80.81
bssfp	GM	dzz	10.27	2.63	4.12	13.49	30.05
dwi-tensor	GM	dzz	5.51	1.81	2.19	7.57	16.91
pc-bssfp	GM	dzz	8.60	2.98	4.16	12.81	30.13
t1w	GM	dzz	7.86	2.18	3.54	11.39	28.28
bssfp	WM	dzz	8.83	2.44	3.48	14.50	25.67
dwi-tensor	WM	dzz	6.32	1.78	2.90	10.95	21.34
pc-bssfp	WM	dzz	7.94	2.36	3.64	11.59	18.20
t1w	WM	dzz	8.21	2.07	3.54	12.90	27.52
bssfp	CSF	fa	24.14	14.64	21.16	25.62	30.65
dwi-tensor	CSF	fa	11.71	7.65	10.53	16.00	35.33
pc-bssfp	CSF	fa	27.17	15.25	21.73	34.28	44.12
t1w	CSF	fa	18.92	13.15	16.01	23.91	39.01
bssfp	GM	fa	21.55	14.22	17.88	23.52	38.12
dwi-tensor	GM	fa	12.11	8.55	9.48	21.18	42.45
pc-bssfp	GM	fa	23.95	14.64	19.03	30.33	38.34
t1w	GM	fa	15.62	12.13	13.85	24.02	49.69
bssfp	WM	fa	17.05	14.64	16.45	23.59	42.57
dwi-tensor	WM	fa	15.59	9.28	11.19	24.60	39.85
pc-bssfp	WM	fa	21.21	15.27	18.22	24.62	30.75
t1w	WM	fa	15.00	11.45	13.16	21.35	49.03
bssfp	CSF	md	16.12	9.85	13.62	19.32	42.98
dwi-tensor	CSF	md	6.70	4.52	5.82	9.23	11.25
pc-bssfp	CSF	md	18.34	10.17	14.34	21.31	51.21
t1w	CSF	md	15.90	8.34	12.25	18.82	44.37
bssfp	GM	md	12.86	6.67	9.46	16.43	27.70

dwi-tensor	GM	md	6.59	5.01	5.90	9.43	14.46
pc-bssfp	GM	md	13.36	7.55	10.96	17.87	23.25
t1w	GM	md	11.57	6.08	7.53	16.24	27.40
bssfp	WM	md	10.99	4.89	5.88	17.74	28.02
dwi-tensor	WM	md	7.62	4.53	6.58	13.35	18.35
pc-bssfp	WM	md	9.71	3.96	7.70	11.47	25.68
t1w	WM	md	8.84	4.09	5.50	17.03	26.45
bssfp	CSF	ad	13.69	8.18	11.04	18.37	258.12
dwi-tensor	CSF	ad	5.51	3.91	4.90	7.00	12.37
pc-bssfp	CSF	ad	14.13	8.71	12.15	20.63	267.30
t1w	CSF	ad	13.50	7.44	9.48	18.36	282.04
bssfp	GM	ad	11.46	6.43	8.32	13.94	47.14
dwi-tensor	GM	ad	5.14	3.82	4.25	6.29	13.67
pc-bssfp	GM	ad	10.53	7.05	8.93	13.48	46.49
t1w	GM	ad	10.36	5.63	7.07	14.08	46.95
bssfp	WM	ad	9.36	6.39	6.89	13.59	39.24
dwi-tensor	WM	ad	4.39	3.19	3.82	8.27	10.91
pc-bssfp	WM	ad	7.93	6.44	7.13	9.54	40.32
t1w	WM	ad	8.02	5.26	5.96	13.14	44.03
bssfp	CSF	rd	17.47	11.48	15.01	22.43	44.62
dwi-tensor	CSF	rd	8.38	5.30	6.79	10.38	18.98
pc-bssfp	CSF	rd	21.04	11.81	18.06	25.59	37.23
t1w	CSF	rd	17.09	9.28	14.00	21.37	130.92
bssfp	GM	rd	15.91	8.95	11.58	19.10	47.44
dwi-tensor	GM	rd	8.36	6.29	7.06	12.66	19.04

pc-bssfp	GM	rd	16.45	9.63	13.20	21.84	38.37
t1w	GM	rd	14.37	7.16	9.07	20.36	102.32
bssfp	WM	rd	13.23	7.09	8.79	22.16	42.23
dwi-tensor	WM	rd	11.47	7.41	9.69	18.17	24.60
pc-bssfp	WM	rd	14.58	7.05	10.52	17.01	49.78
t1w	WM	rd	11.46	6.27	7.70	20.65	69.33
bssfp	CSF	inclination	44.59	40.74	42.76	46.20	70.68
dwi-tensor	CSF	inclination	37.50	30.85	34.46	38.43	83.76
pc-bssfp	CSF	inclination	48.14	42.88	46.09	49.69	64.08
t1w	CSF	inclination	45.04	40.70	43.62	47.20	62.75
bssfp	GM	inclination	45.17	39.87	43.08	47.39	74.33
dwi-tensor	GM	inclination	34.78	29.57	33.32	36.55	91.29
pc-bssfp	GM	inclination	47.45	42.31	45.80	49.03	67.65
t1w	GM	inclination	41.47	36.76	40.38	43.72	66.75
bssfp	WM	inclination	46.04	37.40	40.56	50.79	69.06
dwi-tensor	WM	inclination	31.99	24.87	27.08	38.17	78.80
pc-bssfp	WM	inclination	44.72	39.06	42.10	50.82	66.81
t1w	WM	inclination	48.34	35.75	41.57	50.82	64.35
bssfp	CSF	azimuth	155.15	143.25	149.07	46[illegible]2900	497.38
dwi-tensor	CSF	azimuth	132.05	114.56	126.85	5[illegible]8637	251.62
pc-bssfp	CSF	azimuth	158.04	148.12	153.43	1[illegible]0044	855.89
t1w	CSF	azimuth	150.24	138.36	141.96	3[illegible]5651	540.34
bssfp	GM	azimuth	173.49	148.50	162.65	19[illegible]9414	761.57
dwi-tensor	GM	azimuth	145.53	114.30	135.69	16[illegible]7505	779.92
pc-bssfp	GM	azimuth	181.15	150.28	169.04	19[illegible]2815	425.24
t1w	GM	azimuth	161.71	138.66	153.40	17[illegible]5989	377.51

bssfp	WM	azimuth	207.70	159.43	180.22	25[illegible]7040	963.12
dwi-tensor	WM	azimuth	165.78	115.17	145.90	20[illegible]071	801.10
pc-bssfp	WM	azimuth	213.83	162.21	188.67	24[illegible]533	922.42
t1w	WM	azimuth	208.12	149.64	181.52	26[illegible]771	398.94

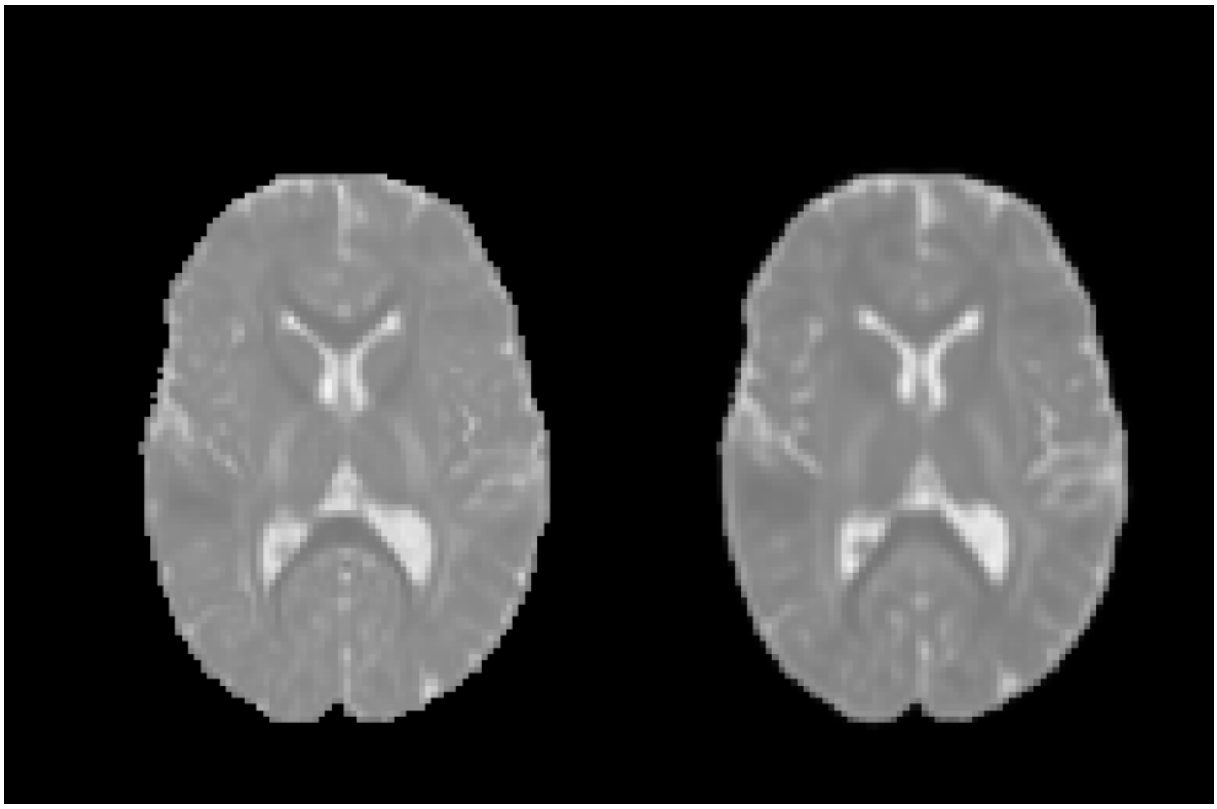

Figure .9. *Comparison between D_{zz} of original DT and predicted DT using pc-bSSFP as input modality.*

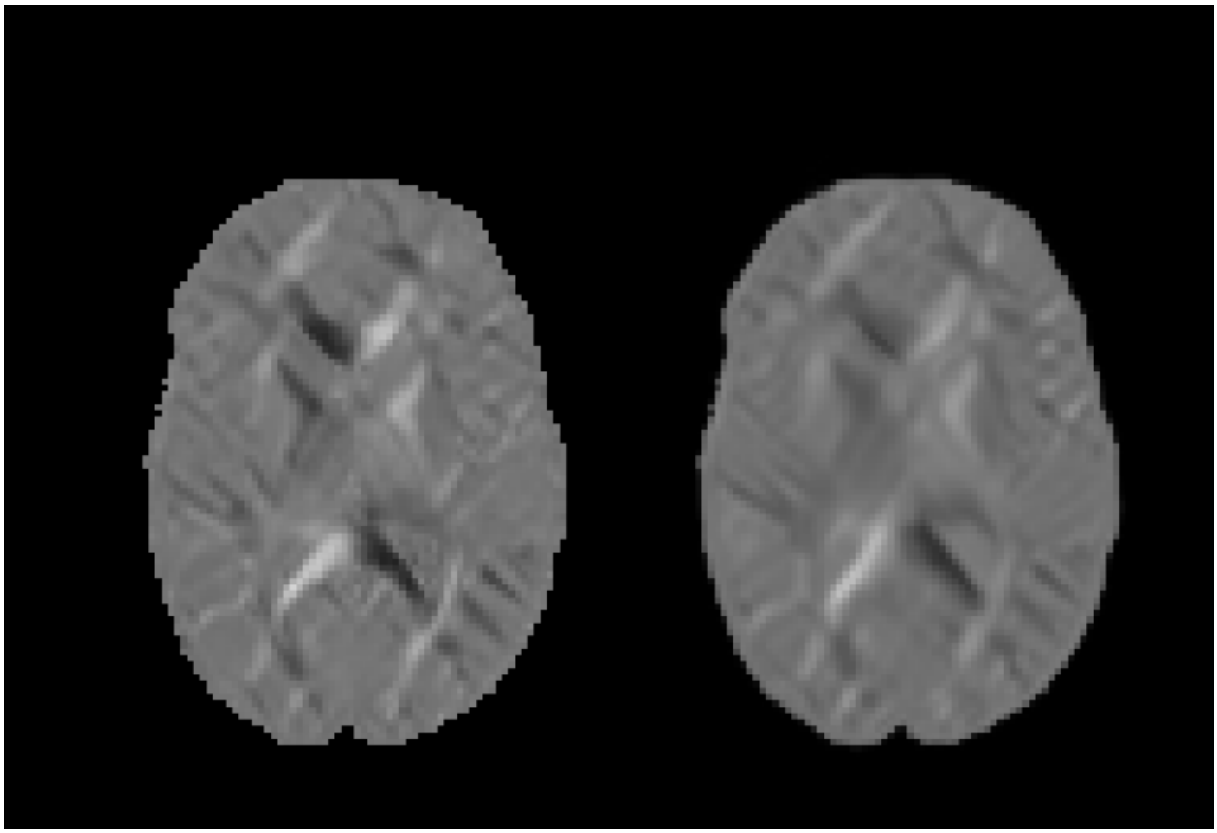

Figure .10. *Comparison between D_{xy} of original DT and predicted DT using pc-bSSFP as input modality.*

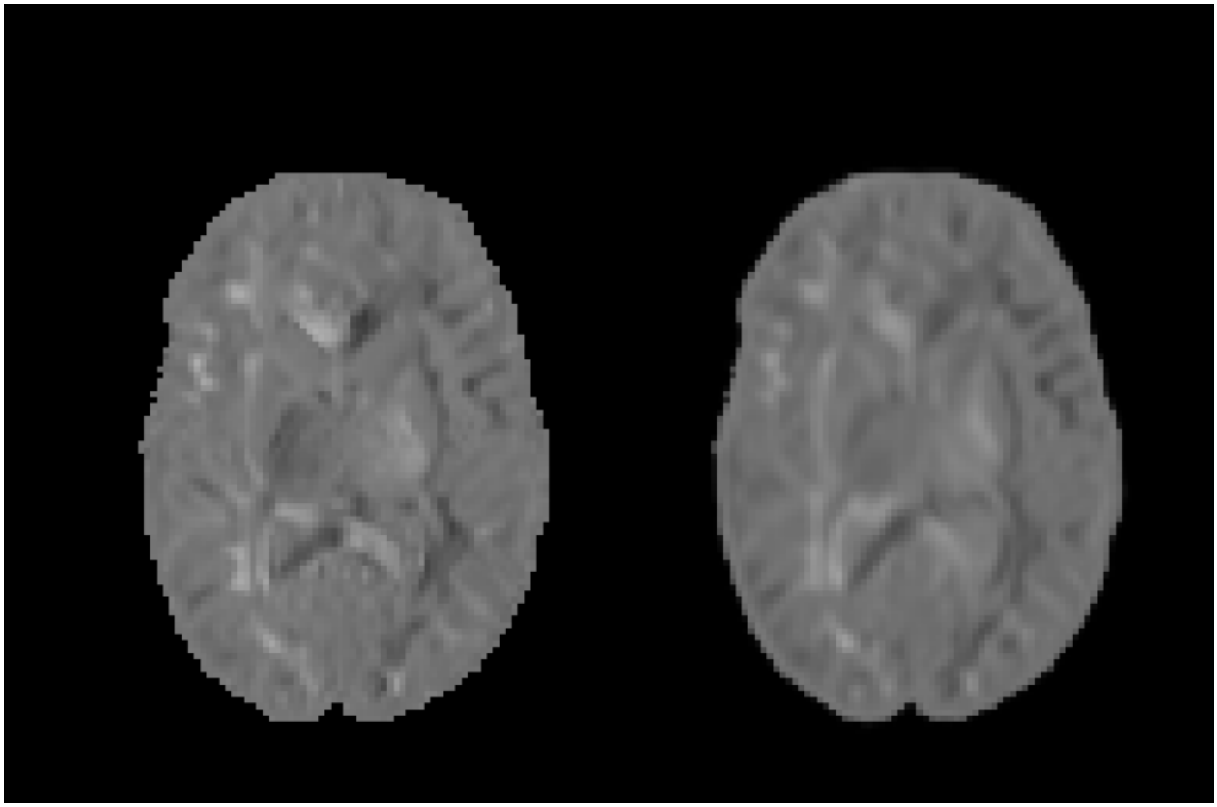

Figure .11. *Comparison between D_{xz} of original DT and predicted DT using pc-bSSFP as input modality.*

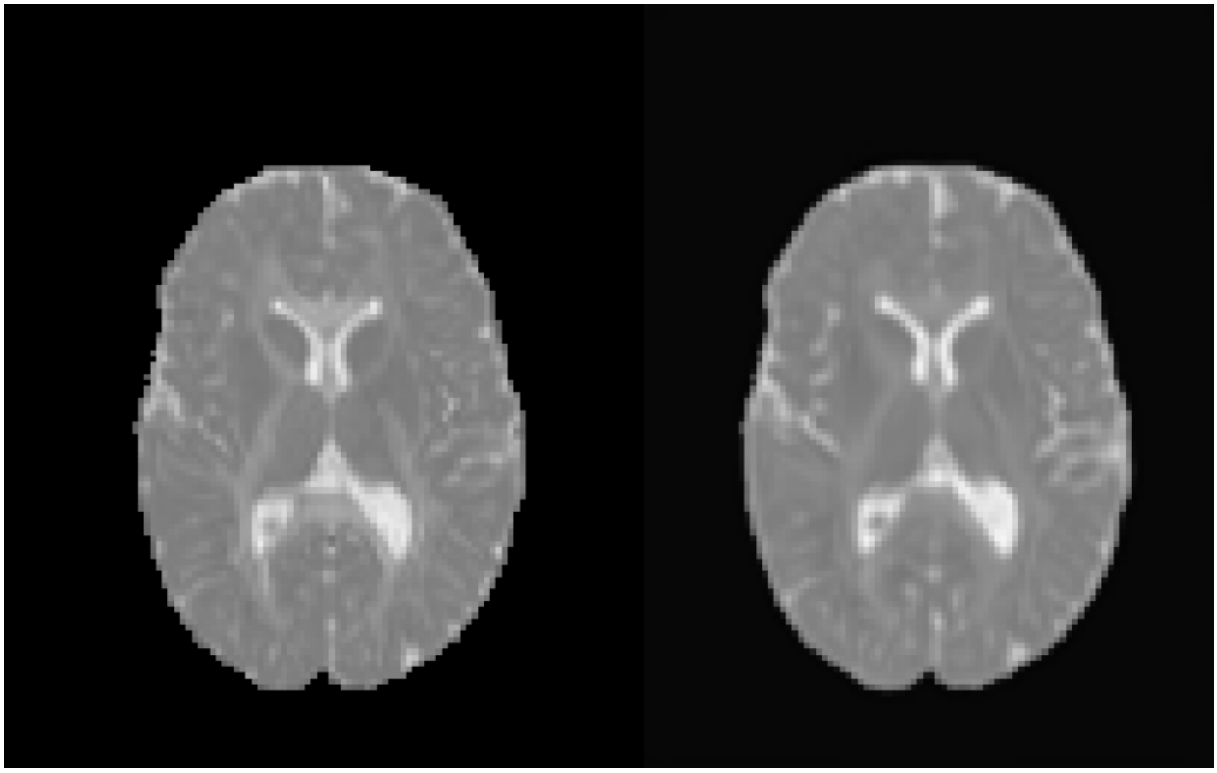

Figure .12. *Comparison between RD calculated from the original DT and the predicted DT using pc-bSSFP as input modality.*

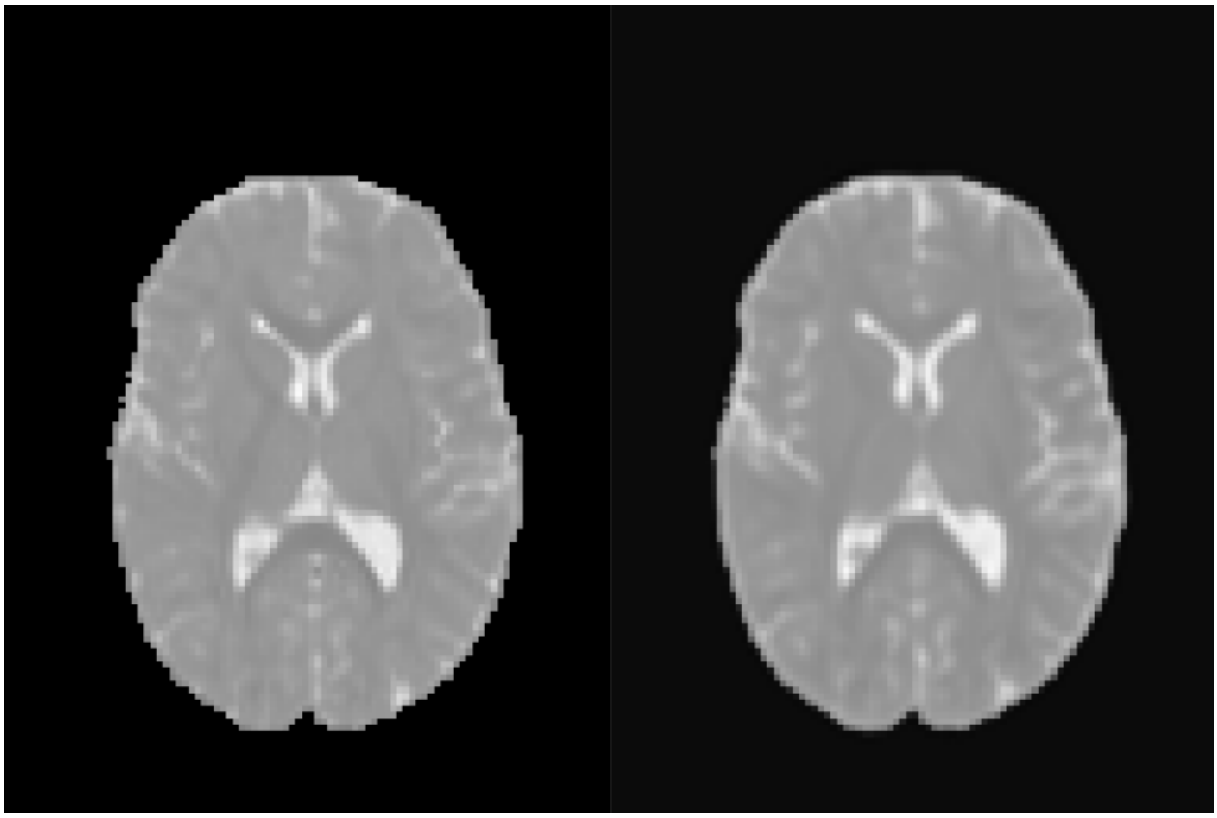

Figure .13. *Comparison between AD calculated from the original DT and the predicted DT using pc-bSSFP as input modality.*