Vrije Universiteit Amsterdam
Faculty of Science

Amsterdam UMC
Physics-MRI group
Quantitative MRI group

MASTER THESIS

Deep Learning for Outlier Detection in 4D IVIM MRI

A study on real-time Automated Artifact Detection

Author: Floor Deben (*2601450*)

1st daily supervisor: Daan Kuppens (Amsterdam UMC)
2nd daily supervisor: Dr. Oliver Gurney-Champion (Amsterdam UMC)
1st supervisor: Dr. Rikkert Hindriks (VU)
Second reader: Dr. Peter Bloem (VU)

A thesis submitted in fulfillment of the requirements for the Master of Science degree in Business Analytics at the Vrije Universiteit Amsterdam.

June 28, 2024

Deep Learning for Outlier Detection in 4D IVIM-MRI Data

Floor Deben

Internship Report

Vrije Universiteit Amsterdam
Faculty of Science
Business Analytics
De Boelelaan 1081a
1081 HV Amsterdam

Host organization:
Amsterdam UMC (location AMC)
Department of Radiology and Nuclear Medicine
Quantitative MRI group - MRI-Physics group
Meibergdreef 9
1105 AZ Amsterdam

June 2024

Preface

This research is conducted for the Master Project in Business Analytics at the Vrije Universiteit Amsterdam. It is an exploratory research focused on automated outlier detection in 4D IVIM MRI. To this day this remains a challenging task that lacks a golden standard solution as it is an insufficiently recognized issue within the medical field.

By contributing to the development of an automated outlier detection model in 4D IVIM MRI data, this research endeavors to provide healthcare professionals with a framework to resolve severe image quality issues as well as providing useful insights on which methods are suited for this. Ultimately, this improvement in image quality is crucial for unlocking the full potential of quantitative MRI in patient care.

This research was conducted at the quantitative MRI group of the MRI-physics group, which is part of the Department of Radiology and Nuclear Medicine of the Amsterdam UMC. This group researches and develops new approaches to AI-driven quantitative MRI. Advanced deep learning methods are deployed in order to develop these approaches.

First and foremost, I would like to express my gratitude to Daan Kuppens from the Amsterdam UMC for his supervision throughout this research and the weekly meetings. His guidance and expertise have been invaluable during this research. I would also like to thank Oliver Gurney-Champion for his expertise and for providing me the opportunity to undertake my graduation internship at the Quantitative MRI research group of the Amsterdam UMC.

Moreover, I would like to thank Rikkert Hindriks from the Vrije Universiteit Amsterdam for his support as my supervisor. Additionally, I extend my appreciation to Peter Bloem for being my second reader from the Vrije Universiteit Amsterdam.

Summary

Context. Intravoxel Incoherent Motion (IVIM) MRI provides a unique and powerful tool for unraveling the complexities of tissue micro-structure and perfusion dynamics, thereby advancing our understanding of physiological processes and enhancing clinical management strategies. However, despite the methodological advancements and the potential of IVIM MRI to revolutionize medical diagnostics and treatment monitoring, the technique faces significant challenges and is therefore rarely used clinically. A significant portion of studies indicate that the noise within quantitative MRI datasets poses significant challenges to data acquisition, processing, and interpretation, which testifies of the need for substantial improvement in both quality and reliability.

Goal. Therefore, in a clinical setting, we want to detect artifactual measurements, such that these outliers can be filtered out or, ideally, re-measured. By contributing to the development of an automated outlier detection model in 4D IVIM MRI data, this research endeavors to provide healthcare professionals with a framework to resolve severe image quality issues as well as providing useful insights on which methods are suited for this. Ultimately, this improvement in image quality is crucial for unlocking the full potential of quantitative MRI in patient care.

Method. To achieve the research goal, we firstly conduct a thorough literature review. Hereafter, we propose our first model - an algorithm for automatically labeling our data using absolute z-scores. By manually validating its performance, it functioned as a proof-of-concept prior to developing the second model - a deep learning solution to automated 4D outlier detection. As deep learning solution, we test four different fully-connected Neural Networks (the base models) and three Convolutional Neural Networks (CNNs). All models are evaluated on in-vivo data. The architectures' performance as well as their suitability to the problem at hand are analysed both quantitatively and qualitatively.

Results. The results of our proposed data labeling algorithm exhibit a promising potential in detecting outliers. By manually performing a visual inspection of these outliers, we can conclude that especially interleaved motion artifacts are detected very well. Although, the model is able to capture local artifacts too. The outcomes of this model were sufficient to function as proof-of-concept and to provide pseudo-labels for the deep learning models. The results of the deep learning models show the potential suitability of these models to this problem by being able to achieve perfect training performances. However, the models strongly suffer from overfitting. Therefore, the

validation performances are uncertain and unstable. Moreover, there is a large discrepancy between the achieved training and validation performance. Our preferred model is the CNN_{reps}, which uses one input channel per repeated measurement and is trained batch-wise. This model suffers least from overfitting, shows the greatest potential, is least uncertain and offers the greatest flexibility in terms of restrictions to input data and model tuning. Lastly, the effects of regularising the CNNs are examined. For the CNN_{reps}, adding regularisation reduces the overfit, but slightly increased the model's uncertainty.

Conclusions. In conclusion, we introduced a promising and novel approach to automatically label outliers in IVIM MRI data based on absolute z-scores. Since these outliers highly correlated with the presence of artifacts, this approach functioned as a proof-of-concept and therefore, the resulting labels could be used as pseudo-labels to facilitate the development of deep learning alternatives. These deep learning alternatives, in the form of fully-connected Neural Networks and CNNs, showed potential to function as automated outlier detection model for IVIM MRI data, but need significant improvement in order to ensure reliability. The main limitation of the current models is the significant overfit. Further research should therefore prioritise resolving the overfitting issue to enable an in-depth analysis of the reliability and real-time applicability of the models.

Contents

1 Introduction

Diffusion-Weighted Imaging (DWI) (Le Bihan and Breton, 1985; Merboldt et al., 1985; Taylor and Bushell, 1985; Le Bihan et al., 1986) has become a widely adopted imaging technique over the past few years (Baliyan et al., 2016), as it is able to provide a unique insight into white matter architecture. Unlike traditional MRI methods that primarily focus on anatomical imaging, diffusion MRI (dMRI) provides insights into the dynamic behavior of water molecules across tissue types at a microscopic level - diffusion. Its unparalleled sensitivity to microstructural changes makes it primarily suited for diagnosing and monitoring conditions such as tumors, strokes, traumatic brain injury, neurodegenerative and neuropsychiatric disorders (Soares et al., 2013; Sundgren et al., 2004; Mori and Zhang, 2006).

A notable advancement within dMRI is Intra Voxel Incoherent Motion MRI (IVIM MRI), which is an advanced dMRI technique designed to capture both diffusion and perfusion phenomena within biological tissues (le Bihan et al., 1988). During an IVIM MRI scan, the tissue is subjected to a series of magnetic field gradients, known as diffusion gradients, in a particular direction. This sensitizes the MRI signal to the random motion of water molecules - diffusion - in this direction. Additionally, IVIM MRI captures the signal changes caused by the movement of blood within the microvasculature, known as perfusion (le Bihan et al., 1992).

By acquiring multiple images at varying diffusion sensitivities, typically quantified by a parameter called the b-value (Le Bihan et al., 2001), a series of diffusion-weighted images (DWIs) is obtained. The IVIM MRI acquisition protocol facilitates the separation of diffusion and perfusion effects within each voxel, the three-dimensional units of imaging (Lemke et al., 2010). At lower b-values, where the diffusion sensitization is relatively low, the acquired signal is predominantly influenced by the perfusion-related motion of water molecules (le Bihan et al., 1986). At higher b-values, diffusion effects become more prominent and therefore, the acquired signal primarily represents the diffusion-related motion of water molecules within the tissue (le Bihan et al., 1986; le Bihan et al., 2008).

Through sophisticated mathematical modeling and analyzing the signal decay across multiple b-values, IVIM MRI enables the extraction of quantitative parameters that

characterize tissue micro-structure and perfusion properties. These parameters include the diffusion coefficient (D), which describes the rate of water diffusion within the tissue, the pseudo-diffusion coefficient (D^*), which represents perfusion-related motion, and the perfusion fraction (f), which quantifies the proportion of perfusion-related signal contribution within the voxel (Callaghan et al., 1991).

The integration of diffusion and perfusion information within a single imaging modality enables IVIM MRI to provide valuable insights into tissue micro-structure and vascular dynamics. This comprehensive assessment holds immense potential for clarifying physiological processes, diagnosing pathological conditions, and monitoring treatment responses in various clinical settings, particularly in oncology (le Bihan et al., 2008; Detre et al., 1992).

In essence, IVIM MRI provides a unique and powerful tool for unraveling the complexities of tissue micro-structure and perfusion dynamics, thereby advancing our understanding of physiological processes and enhancing clinical management strategies.

However, despite the methodological advancements and the potential of IVIM MRI to revolutionize medical diagnostics and treatment monitoring (Cho et al., 2017; Zhu et al., 2017; Ma et al., 2018; Klaassen et al., 2020), the technique faces significant challenges and is therefore rarely used clinically (Kaandorp et al., 2021). A significant portion of studies indicate that the noise within quantitative MRI datasets poses significant challenges to data acquisition, processing, and interpretation, which testifies of the need for substantial improvement in both quality and reliability. Accurate parameter estimation is challenged by various sources of uncertainty, primarily caused by the fact that dMRI has low Signal-to-Noise Ratio (SNR) and resolution and is very susceptible to motion (Farrell et al., 2007; Choi et al., 2011; Polders et al., 2011).

Therefore, dMRI data is often contaminated by various sources of artifacts. Without any exclusion or correction of these artifacts, the results of any subsequent analysis could be biased, making their interpretation unreliable (Bammer et al., 2003; Van Dijk et al., 2012; Reuter et al., 2015). Hence, quality control (QC) is an essential step before dMRI goes into further processing (Bastiani et al., 2019; Le Bihan et al., 2008; Tournier et al., 2011; Pierpaoli, 2010; Soares et al., 2013; Ahmad et al., 2023).

In practice, these quality issues have led to under-utilization of advanced models like IVIM-DWI, which could provide more accurate measures of underlying physiology than conventional models. Furthermore, most studies have been limited to basic parameters due to data quality issues and in many cases, patients had to be excluded. A solution to tackle these problems is a necessity in order to deploy the full potential of the promising technique, IVIM MRI. Therefore, in a clinical setting, we want to detect artifactual measurements, such that these anomalies can be filtered out or, ideally, re-measured.

However, to this day quality control in dMRI remains an insufficiently recognized issue within the dMRI research community as there are no commonly available, user-friendly tools specifically engineered to address the issue of dMRI QC comprehensively. As a result, current dMRI studies often perform a poor job at dMRI QC (Oguz et al., 2014). It is still a challenging and extremely laborious process, since quality control and artifact identification are undertaken mostly by visual inspection (Samani et al., 2020) of all volumes, or even slices, by experts. This process becomes infeasible when large datasets are involved and quickly becomes a bottleneck for any dMRI study. Alternatively, people opt for spot checking that may leave a lot of artifactual data in leading to incorrect results. An additionally challenging characteristic of dMRI data is the inherently low signal to noise ratio. Furthermore, it is a unavoidably subjective process, which makes it prone to inter- and intra-observer variability (Bauer et al., 2013; Victoroff et al., 1994; Samani et al., 2020). In conclusion, an efficient golden standard for quality control of dMRI data is still absent in existing literature.

This makes a normalized and automated image-based artifact detection model, that can replicate or exceed human performance, imperative for any dMRI processing pipeline (Ettehadi et al., 2022; Rizwan-i-Haque et al., 2020). Such model would enhance assessment and diagnosis efficiency, reduce variability, and improve reproducibility in order to save time for medical professionals and provide reliable analysis and monitoring.

In recent years, deep learning techniques have emerged as a promising tool for both outlier detection and image processing tasks, which makes it a natural choice for the problem at hand. However, while deep learning models have demonstrated remarkable success in solving general computer vision problems, they face specific challenges when applied to MRI images. To this day outlier detection in 4D image-based data remains a challenging task that lacks a golden standard solution.

A first challenge is the previously mentioned sparsity of annotated data, which limits resources for the validation of developed models, thereby impeding the research on automated outlier detection in medical data. Moreover, in existing literature the vast majority of models are either 2D or 3D outlier detection models, lacking solutions in 4D. This means that, in most cases, either the spatial or the temporal information is analysed, whereas we hypothesize that exploiting spatio-temporal information could enhance performance.

Despite these challenges, deep learning algorithms have shown promising results in studies on image-based problems and outlier detection. Therefore, the objective of this thesis is to devise novel approaches for automated outlier detection in IVIM MRI. Ultimately, the goal is operation in real-time, allowing for immediate rescanning. In particular, this would lead to enhanced image quality, but also increased efficiency in terms of scanning logistics, thereby improving patient experience. To achieve this, the thesis paper aims to answer the following research question: *How can deep learning techniques be used to accurately detect motion artifacts in 4D IVIM MRI?*. For that, the research is divided into the following sub-questions:

1. Can the residuals of an IVIM fitting algorithm direct us towards the presence of an artifact?
2. Can deep learning techniques provide a reliable solution for automated outlier detection in IVIM MRI?
3. Can incorporating spatio-temporal information of the data enhance our models' sensitivity to detecting artifacts?
4. What is considered essential information to facilitate reliable detection?

In order to answer these questions, we first conduct a thorough literature review of current state-of-the-art solutions to sub-problems of this research. Hereafter, we propose our first model - an algorithm for automatically labeling our data. By manually validating its performance, it can function as a proof-of-concept prior to developing the second model - a deep learning solution to automated 4D outlier detection. Both models are evaluated on in-vivo data. The architecture's performance as well as its suitability to the problem at hand will be analysed both quantitatively and qualitatively. By contributing to the development of an automated outlier detection model in

4D IVIM MRI data, this research endeavors to provide healthcare professionals with a framework to resolve severe image quality issues as well as providing useful insights on which methods are suited for this. Ultimately, this improvement in image quality is crucial for unlocking the full potential of quantitative MRI in patient care.

I will be part of the quantitative MRI group of the MRI-physics group, which is part of the Department of Radiology and Nuclear Medicine of the Amsterdam UMC. This group researches and develops new approaches to AI-driven quantitative MRI. Advanced deep learning methods are deployed in order to develop these approaches.

2 Literature Review

This literature review firstly examines the current main limitations of IVIM MRI, discussed in Section 2.1. In Section 2.2, we review literature on using deep learning techniques for outlier detection. Moreover, in Section 2.3 we examine the topic of anomaly detection in quantitative MRI.

2.1 IVIM challenges

2.1.1 Noise

There are various challenges in IVIM MRI regarding the acquisition. First of all, the scale of voxel values in MR images is not standardized (Havaei et al., 2017). Grayscale values can vary significantly depending on the type of MR machine used and the specific acquisition protocol, leading to inconsistency across different hospitals. Secondly, there is the so-called partial volume effect in MRI images, meaning that more than one class of tissue type can occupy the same image pixel or voxel, which complicates segmentation and classification tasks. These pixels or voxels are commonly called mixels (Ruan et al., 2000). But more importantly, IVIM MRI suffers from multiple sources of noise.

The nature of noise in IVIM MRI is multifaceted, encompassing various sources such as thermal noise, electronic noise, and physiological motion artifacts (le Bihan et al., 2008; Jones et al., 2013). Thermal noise, stemming from random thermal fluctuations within MRI systems, contributes to baseline signal variability and undermines the fidelity of acquired data. Electronic noise, arising from the amplification and digitization processes, further exacerbates signal distortions, particularly in low-SNR environments. Additionally, motion artifacts induced by physiological processes, such as respiration and cardiac pulsation, introduce spatio-temporal inconsistencies in IVIM MRI images, confounding accurate signal analysis (Cheng et al., 2012; Veraart et al., 2016).

Multiple studies have investigated the impact of noise on IVIM MRI data quality, revealing its detrimental effects on image resolution, signal-to-noise ratio (SNR), and quantitative parameter estimation (Parker et al., 2023; Cheng et al., 2012; McVeigh et al., 1985). High levels of noise can obscure subtle signal variations associated with tissue micro-structure and perfusion, leading to unclear textures, blurry tissue borders, and decreased diagnostic sensitivity and specificity. Moreover, noise-induced artifacts,

such as blurring and ghosting, further degrade image quality and compromise the reliability of IVIM MRI-based analyses (Lee et al., 2015; Dyvorne et al., 2013; Jones et al. 2004a). On top of that, the retrieval of true signal values from noise corrupted data is far from trivial, both theoretically and practically (Gudbjartsson et al., 1995, Sijbers et al., 1998).

Efforts to mitigate noise in IVIM MRI encompass a spectrum of techniques, ranging from acquisition optimization to post-processing algorithms. Acquisition strategies aimed at increasing SNR, such as higher field strengths, parallel imaging, and optimized gradient designs, have shown promise in enhancing image quality and mitigating the effects of noise (Barbieri et al., 2015). Furthermore, advanced denoising algorithms, including wavelet-based methods, non-local means filtering, and deep learning approaches, have been developed to suppress noise while preserving underlying signal features in IVIM MRI datasets (van de Ville et al., 2007; Veraart et al., 2016; André et al., 2014; Jones et al., 2004b; Baselice et al., 2017; Phophalia et al., 2017; Zhang et al., 2015; Salimi-Khorshidi et al., 2014; Lysaker et al., 2003). These techniques offer valuable insights into tissue microstructure and perfusion dynamics by enhancing the robustness and accuracy of IVIM MRI-based analyses.

Noise in IVIM MRI presents significant challenges to data acquisition and analysis, necessitating ongoing research efforts to develop effective noise reduction strategies. By elucidating the nature of noise sources, exploring novel acquisition techniques, and refining denoising algorithms, future studies aim to enhance the reliability and clinical utility of IVIM MRI for investigating tissue microstructure and perfusion dynamics in various physiological and pathological contexts.

2.1.2 Artifacts

Artifacts are a primary source of noise in dMRI. There are several kinds of artifacts including motion, multi-band interleaving, Gibbs ringing, low signal to noise ratio (SNR), ghosting, susceptibility, herringbone, and chemical shift (Wood et al., 1985; Smith et al., 1991; Smith et al., 2010; Simmons et al., 1994; Schenck, 1996; Heiland, 2008; Moratal et al., 2008; Krupa et al., 2015; Le Bihan et al., 2008). These artifacts stem from various sources, including hardware limitations, physiological motion, and imaging parameters.

Motion artifacts, such as ghosting and blurring, can result from patient movement during scanning. Susceptibility artifacts arise from magnetic field inhomogeneities at tissue interfaces, while geometric distortions may occur due to gradient non-linearity. Other artifacts, such as chemical shift artifacts and aliasing artifacts, can further degrade image quality and compromise data interpretation (Mesri et al., 2020; Bellon et al., 1986; Mirowitz et al., 1999; Bernstein et al., 2006; Zhuo et al., 2006; Stadler et al., 2007; Morelli et al., 2011).

Motion artifacts are very common in dMRI. The motion during scanning can result in signal dropout from individual slices in a volume, in ghosting artifacts, or in artifacts arising from stitching together misaligned data in acquisitions that use slice interleaving or multi-band acceleration (Ahmad et al., 2023). An interleaved acquisition protocol starts by acquiring, for instance, all even-numbered slices, hereafter all odd-numbered slices are acquired. If there is patient motion between the two phases of acquisition, consecutive even- and odd-numbered slices are misaligned, which causes jagged edges in the coronal dimension. This misplacement of the signal is different from signal dropout, also caused by intra-volume movement, which leads to direct loss of the signal (Graham et al., 2018).

Efforts to mitigate artifacts in IVIM MRI involve hardware optimization, protocol refinement, and post-processing techniques. Hardware improvements, such as gradient system upgrades and shimming techniques (Ratai et al., 2016), aim to reduce susceptibility artifacts and magnetic field distortions. Optimized imaging protocols, including motion correction algorithms and parallel imaging methods, help minimize motion-related artifacts and improve image quality. Furthermore, advanced image processing algorithms, such as distortion correction methods and noise reduction techniques, aid in artifact suppression and enhance data fidelity (Duffy et al., 2021; Griswold et al., 2002; Reese et al., 2003; Zaitsev et al., 2004; Weaver et al., 1992).

2.2 Deep learning for outlier detection

2.2.1 Image-based

Deep learning has dramatically transformed the landscape of outlier detection in images, providing advanced solutions to a challenge that spans numerous applications,

from security surveillance to medical imaging diagnostics. Chandola et al. (2009) highlight the application of anomaly detection across a wide range of domains and discuss domain-specific challenges and requirements, hereby the authors emphasize the versatility and necessity of anomaly detection techniques. Outlier or anomaly detection in images refers to the identification of images or regions within images that deviate significantly from the norm.

Anomaly detection methods can be categorized into different types based on the underlying approach, including statistical methods, clustering-based methods, nearest neighbor-based methods, and classification-based methods (Chandola et al., 2009). Given the vast amount of image data generated daily and the complexity of defining what constitutes an "anomaly," deep learning approaches offer a compelling advantage due to their ability to learn robust high-level features from data, their scalability and generalizability without explicit programming.

Convolutional Neural Networks (CNNs) are at the forefront of deep learning techniques for image processing, including outlier detection. Their ability to automatically and adaptively learn spatial hierarchies of image features from low- to high-level patterns makes them particularly suited for identifying atypical patterns in visual data (Krizhevsky et al., 2012; LeCun et al., 1998). In 2017, Schlegl et al. introduced AnoGAN, a groundbreaking model that uses Generative Adversarial Networks (GANs) for detecting anomalies in retinal images. AnoGAN learns to model the distribution of normal images and then identifies anomalies based on the GAN's ability to reconstruct input images.

Moreover, U-Net (Ronneberger et al., 2015) approaches are amongst the most popular image-based outlier detection architectures. The U-Net is specifically designed for image segmentation tasks and is computationally efficient because of its key feature: skip-connections. Furthermore, it can handle corrupted and missing data, learn robust feature representations, and has a good balance between accuracy and computational efficiency. In 2022, Zhang et al. presented a noteworthy approach that leverages the efficiency of skip-connections by introducing a skip connection training algorithm, achieving very promising performance. In 2020, Ibtehaz et al. introduced the MultiResUNet, which aims to enhance the feature extraction capabilities of the original U-Net architecture by incorporating multi-resolution blocks. This approach outperformed the

state-of-the-art U-Net approaches, especially on segmenting complex and heterogeneous biomedical images.

Autoencoders (AEs) are another popular tool for outlier detection in images. AEs learn to compress and then reconstruct input data, with the hypothesis that anomalies will have higher reconstruction errors. A major advantage of AEs is their fairly high reconstruction resolution thanks to a supervised training signal coming from the reconstruction objective (Baur et al., 2019). Notable is the work of Vincent et al. (2010), which introduced the denoising autoencoder. This is a variant of the standard autoencoder, which is trained to reconstruct a clean input from a corrupted version. By introducing a novel approach for learning through denoising, the authors have paved the way for advancements in unsupervised learning and the development of deep neural networks capable of capturing complex data representations. Moreover, Sakurada et al. (2014) showed the effectiveness of this approach in anomaly detection by leveraging the reconstruction error as an anomaly score. Furthermore, Gong et al. (2019) presented a memory-augmented autoencoder (MemAE), which shows great effectiveness in detecting anomalies across various different datasets.

Variational Autoencoders (VAEs) have also been extensively used for image-based outlier detection. In 2013, Kingma et al. introduced the VAE, which was a groundbreaking approach in the field of machine learning and deep learning. VAEs, through their probabilistic approach to encoding and decoding images, can effectively model the distribution of normal data. The VAE represents a significant advancement in the unsupervised learning domain, specifically in generative modeling and latent variable models. Alongside Kingma et al., Rezende et al. (2014) significantly contributed to the field of deep learning and generative modeling by presenting a general and efficient framework for variational inference and learning. Furthermore, An et al. (2015) demonstrated how VAEs could be utilized to identify anomalies by examining the reconstruction probability. Images that cannot be reconstructed well are considered outliers. In 2019, Zimmerer et al. and Guo et al. presented promising and robust approaches to unsupervised anomaly localization using VAEs.

Despite the advances, deep learning for outlier detection in images faces several challenges. Data scarcity is a primary concern, as anomalies are by definition rare events, making it difficult to obtain sufficient training data. Interpretability of deep learning

models is another significant challenge, as understanding the basis for anomaly detection decisions is crucial for trust and actionability in many applications. Furthermore, the high variability in what constitutes an anomaly across different domains complicates the development of generalized models.

Semi-supervised and unsupervised learning models present a promising avenue for addressing the data scarcity issue, allowing for the effective use of large amounts of unlabeled data. In conclusion, deep learning offers a powerful framework for outlier detection in images, capable of handling the complexity and high dimensionality of image data. As the field progresses, the focus will likely shift towards developing more robust, interpretable, and domain-adaptable models. Continued research in this area holds the promise of significant advancements in automated surveillance, medical diagnostics, and beyond.

2.2.2 Spatial, temporal, spatio-temporal

Deep learning methodologies often prioritize processing either spatial or temporal dimensions, yet the integration of spatio-temporal dynamics within model architectures could significantly augment performance. Contemporary spatio-temporal deep learning strategies predominantly employ CNNs to address spatial dimensions, diverging primarily in their treatment of temporal aspects (Asadi-Aghbolaghi et al., 2017).

A first class of models uses 3D CNNs, which extend kernels to encapsulate the temporal dimension (Liu et al., 2016). Pioneering work by Ji et al. in 2013 employed 3D CNNs for human action recognition. In 2015, Tran et al. proposed the widely adopted C3D architecture after investigating the use of 3D CNNs further on large-scale datasets. Furthermore, Varol et al. (2018) explored the efficacy of long-term convolutions across varying sequence lengths with 3D CNNs.

Given that 3D convolutions increase the model's parameter count, more computationally efficient strategies have been proposed. For instance, Sun et al. (2015) introduced factorized convolutions that decompose the 3D kernel into separate 2D spatial and 1D temporal components, applied in succession. This methodology was further advanced by Qiu et al. (2017), who examined various residual block (He et al., 2016b) variants

for distinct spatial and temporal convolution operations. In 2018, Tran et al. unveiled a 3D CNN model employing mixed convolutional layers, with 3D convolutions restricted to the model's initial layers. An example of how using convolutions for temporal processing has been applied in the medical domain is surgical video analysis (Funke et al., 2019).

Alternatively, several architectures harness recurrent neural networks to model temporal relationships, typically processing spatial dimensions with a 2D CNN before introducing a recurrent mechanism (Ordóñez et al., 2016). Donahue et al. (2015) and Yue-Hei Ng et al. (2015) used this concept by feeding features from a 2D CNN into LSTM (Hochreiter et al., 1997) layers. Pigou et al. (2018) explored various spatio-temporal models, including CNN+LSTM configurations and temporal pooling, with applications extending to medical analyses such as surgical video analysis (Jin et al., 2019) and force estimation (Gao et al., 2018). Furthermore, the development of two-stream architectures has been introduced, wherein a bifurcated convolutional pathway is employed: one pathway is dedicated to processing spatial information through the analysis of individual frames, and the other pathway receives temporal information, for instance, in the form of precomputed optical flow (Simonyan et al., 2014; Feichtenhofer et al., 2016; Wang et al., 2016).

Additionally, various works have investigated the topic of Video Object Segmentation (VOS), which is in essence a spatio-temporal task. Noteworthy are the works of Cheng et al. (2021a), Cheng et al. (2021b), Heo et al. (2021) and Oh et al. (2019). These works all introduce a different methodology, each presenting an efficient and well-performing approach to VOS. Furthermore, Kiran et al. (2018) provide an extensive overview of deep learning based methods used for anomaly detection in videos.

Despite these advances, models capable of processing 4D spatio-temporal data remain scarce. In the natural image domain, 3D spatial data captured by time-of-flight cameras is frequently represented as depth maps, avoiding the need for direct 4D processing. El Sallab et al. (2018) transformed sequences of 3D LiDAR point clouds into 2D projections for convolutional LSTM (Xingjian et al., 2015) processing, facilitating concurrent spatial and temporal analysis. Additionally, Choy et al. (2019) advocated for sparse 4D convolutions to process 4D data directly from depth sensors.

In the realm of medical imaging, 4D CNNs have been applied to tasks such as CT image reconstruction (Clark et al., 2019), segmentation (Myronenko et al., 2020) and MRI reconstruction (Küstner et al., 2020). This area of research also includes image-to-image translation (van de Leemput et al., 2020), functional MRI (fMRI) modeling (Zhao et al., 2018), and fMRI-based disease classification (Bengs et al., 2019). Although these methodologies demonstrate the potential of 4D deep learning, they have yet to exhibit a definitive advantage over lower-dimensional, and thus typically more efficient, deep learning approaches.

2.3 Anomaly detection in quantitative MRI

2.3.1 Segmentation

In recent years, deep learning techniques have emerged as a promising tool for accurately detecting and classifying anomalies in medical images (Lundervold et al., 2019). The number of publications on segmentation-based methods has grown exponentially over the last decades (Havaei et al., 2017), testifying to the need and that it is still work-in-progress. Recent studies have consistently shown the superiority of deep learning approaches over traditional methods for medical image segmentation in MRI (Akkus et al., 2017; Bernal et al., 2019; Garcia-Garcia et al., 2017; Havaei et al., 2017; Kamnitsas et al., 2017; Pereira et al., 2016; Shen et al., 2017; Zhao et al., 2018; Zhou et al., 2019).

Initially, traditional machine learning techniques such as Support Vector Machine (SVM) (Ciritsis et al., 2018; Schnell et al., 2009) and Non-Negative Matrix Factorization (NMF) (Sun et al., 2019) were extensively applied in the realm of medical image analysis. More recently, studies have started to utilize deep learning to further improve segmentation performance. Of note are studies that utilized multilayer perceptron (MLP) (Hastie et al., 2009) to perform MRI-based tissue segmentation (Bagher-Ebadian et al., 2011; Golkov et al., 2016). Specifically, Golkov et al. (2016) showcased the efficacious deployment of MLP for segmenting various tissue types using dMRI data. Furthermore, Isensee et al. (2018) presented a robust and self-adapting framework on the basis of 2D and 3D vanilla U-Nets: the nnU-Net, which achieved a high performance and generalizability.

So far CNNs have been considered as a significant tool for medical segmentation, par-

ticularly within the domain of multi-modal segmentation that integrates dMRI, demonstrating substantial potential for enhanced tissue segmentation (Nie et al., 2018; Zhang et al., 2015). However, the application of such methodologies remains dependent on the accuracy of cross-modality registration necessary for compiling the multi-modal input used for the segmentation.

Focusing on a more specific type of segmentation, namely artifact detection, the literature delineates techniques into two primary categories: those implemented during the acquisition phase (Reese et al., 2003; Jezzard et al., 1995; Andersson et al., 2003; Golan et al., 2018), which may involve alterations to the acquisition protocol or the gathering of additional data, and post-processing strategies implemented after acquisition time (Haselgrove et al., 1996; Jenkinson et al., 2001; Zimmerer et al., 2019). Post-processing methods are predominantly used. Nonetheless, the absence of an objective ground truth complicates the systematic evaluation of existing techniques, thereby inhibiting end-users from making informed decisions. This limitation equally obstructs the development of novel methodologies, as evidencing superiority over pre-existing approaches poses a considerable challenge. (Graham et al., 2016).

2.3.2 Post-processing techniques

Considering post-processing techniques, it is perceptible that computerized approaches for the quality control and identification of artifacts in dMRI data significantly alleviate the challenge of manual inspection. Throughout the years, several tools for automated quality control of dMRI data have been developed, including FSL (Jenkinson et al., 2012; Andersson et al., 2016; Bastiani et al., 2019), DTIPrep (Oguz et al., 2014), DTI Studio (Jiang et al., 2006), and TORTOISE (Pierpaoli et al., 2010). Subsequently, multiple statistical (Roalf et al., 2016) or Artificial Intelligence (AI) methodologies have been proposed, aiming at enhancing quality control and the detection of artifacts in dMRI data (Iglesias et al., 2017; Kelly et al., 2017; Alfaro-Almagro et al., 2018; Fantini et al., 2018; Graham et al., 2018; Samani et al., 2020; Ahmad et al., 2023; Ettehadi et al., 2021). The efficacy of such tools has been extensively evaluated in the works of Liu et al. (2015) and Haddad et al. (2019).

Nevertheless, a notable limitation of these approaches is their specificity to particular

artifact types, predominantly those induced by motion and eddy currents, as detailed in various studies (Liu et al., 2015; Iglesias et al., 2017; Alfaro-Almagro et al., 2018; Graham et al., 2018). This restriction is also applicable to tools like FSL EDDY, DTI Studio, DTIPrep, and TORTOISE. (Ettehadi et al., 2022; Samani et al., 2020)

Conversely, Deep Learning (DL) methods such as QC Automator (Samani et al., 2019) and Squeeze-and-Excitation CNNs (Ettehadi et al., 2022) have demonstrated excellent performance in detecting a broader spectrum of artifacts. However, their validation on patient data remains limited. Moreover, these approaches predominantly rely on 2D deep learning models and require ground-truth annotations for each slice across every volume. This imposes a significant workload on human annotators, particularly during the model fine-tuning phase where annotated subsets are essential for processing a previously unencountered dataset (Ahmad et al., 2023). This aspect underscores a potential area for improvement in the deployment and optimization of DL models for dMRI data quality control and artifact detection.

2.3.3 Models at acquisition time

Recent investigations have endeavored to develop and implement lightweight and efficient deep learning models that can be directly integrated into MRI hardware or deployed on edge devices, aiming to enable real-time artifact detection. Typically, the identification of corruption and subsequent data exclusion are executed during post-processing tensor calculation. While this strategy proves to be effective, it inherently results in a reduced SNR due to the exclusion of corrupted pixels. To mitigate the effects of post-processing quality assurance measures, the implementation of real-time monitoring for image quality, coupled with the reacquisition of images severely affected by artifacts, emerges as a preferable solution. (Li et al., 2013)

Several critical considerations must be addressed when monitoring quality in real-time. Primarily, the outlier assessment should happen on a slice-by-slice basis, diverging from the pixel-by-pixel rejection methodology of post-processing techniques. Furthermore, the establishment of an additional threshold for the permissible level of corruptness of each slice is imperative to avoid an excessive increase in scan time. Despite the efficacy of post-processing quality control measures, they are not without drawbacks, such as

the reduction in SNR. Consequently, the concept of real-time quality monitoring and the re-acquisition of corrupted images have been proposed, albeit with the acknowledgment of increased scan durations. (Li et al., 2013)

In 2022, Gudovskiy et al. introduced a novel approach for real-time, unsupervised anomaly detection and localization using conditional normalizing flows. The authors propose CFLOW-AD, a model that leverages the power of conditional normalizing flows for detecting anomalies in an unsupervised manner. CFLOW-AD had broad implications across numerous domains, including identifying anomalies in medical images. The work of Gudovskiy et al. represents a significant advancement in the field of anomaly detection, offering a practical solution that combines the efficiency of real-time processing with the flexibility of unsupervised learning.

Moreover, Tokuda et al. (2008) introduced an innovative approach to 4D MRI by utilizing navigator-based respiratory signals for adaptive imaging. This method significantly improves the utility of MRI in guided therapeutic applications by compensating for respiratory motion, which has been a major source of artifacts in thoracic and abdominal imaging. By adaptively modifying the imaging parameters based on real-time respiratory motion, their technique enhances image quality and reduces the impact of patient movement, thereby facilitating more accurate and effective MRI-guided interventions. Although the authors did not specifically apply this approach to artifact detection, it could function as an interesting approach to motion artifact identification.

Deep learning-based real-time artifact detection models represent a transformative advancement in qMRI, offering the potential to significantly improve image quality and diagnostic reliability. While challenges persist, particularly concerning data diversity, model generalization, and computational efficiency, ongoing research continues to address these issues. Future advancements are expected to further consolidate the role of DL in ensuring artifact-free qMRI acquisition, heralding a new era in medical imaging.

3 Background

3.1 IVIM MRI

3.1.1 Acquisition

MRI functions through the application of a strong magnetic field, aligning the hydrogen nuclei or protons within bodily tissues. Upon exposure to a radiofrequency pulse, these protons absorb and subsequently re-emit energy, which is detected by highly sensitive receivers. The measurement of these emitted signals enables MRI scanners to construct very detailed images that reflect the spatial distribution of these signals throughout the body.

IVIM MRI incorporates the effects of both diffusion and perfusion within a voxel, which entails the acquisition of a series of diffusion-weighted images at varying levels of diffusion weighting, quantified by the b-value. It is essential for IVIM MRI to acquire the diffusion-weighted images across a spectrum of b-values. Lower b-values, typically ranging from 0 to approximately 200 s/mm^2, predominantly capture signals related to perfusion, thereby enhancing sensitivity to microcirculation. Conversely, higher b-values primarily reflect the signals affected by diffusion. Employing varying b-values enables the separation and quantification of diffusion and perfusion effects.

Models to analyze the signal decay should accommodate the non-Gaussian distribution of noise within the image space. The signal detected by MRI scanners corresponds to a summation of frequency encoding (FE) and phase encoding (PE) waves in k-space, which is a mathematical space of spatial frequencies. By performing a Fourier transformation on the k-space, an image can be reconstructed. Given the Gaussian distribution of noise in k-space, the resultant noise in image space follows a Rician distribution. This type of noise becomes particularly evident at higher b-values, introducing a noise floor that may lead to the underestimation of the perfusion fraction (f).

3.1.2 b-values

The choice of b-value scheme in IVIM MRI highly influences image quality as well as the level of uncertainty in parameter estimation. Determining an optimal set of b-values

is a nuanced process influenced by the desired imaging outcomes, the specific tissue or pathology under investigation, and technical limitations. The primary challenge lies in balancing the need for sufficient diffusion weighting to distinguish tissue types and pathological conditions against the inherent signal-to-noise ratio (SNR) trade-offs at high b-values. Additionally, the heterogeneity of biological tissues necessitates a flexible approach to b-value selection, tailored to the specific imaging scenario.

Ongoing research into b-value optimization strategies continues to refine these techniques, expanding their applicability and improving their contribution to diagnostic imaging and tissue characterization. Despite extensive research efforts, the establishment of a universally optimal b-value scheme for IVIM MRI has not yet been achieved. However, significant progress has been made in tailoring b-value schemes to suit particular applications.

Various studies successfully derive a set of b-values that minimize error, however they are often tailored to a limited set of IVIM parameters. On the other hand, adaptive algorithms that dynamically adjust b-values based on real-time signal characteristics are emerging, promising to enhance the robustness and reliability of diffusion and IVIM MRI measurements. Furthermore, according to various studies, irregular sampling of b-values yields optimal results in minimizing error in parameter estimates. These insights highlight the significance of using a customized b-value scheme tailored to a specific scenario.

3.1.3 Quantification

The extraction of quantitative parameters happens by fitting the measured signal decay over the b-values with a suitable model. In this case, this is the bi-exponential model described by the IVIM formula shown in Equation 1. In this formula, f_p represents the perfusion fraction, D and D^* are, respectively, the diffusion and pseudo-diffusion coefficient, *S(b)* is the signal strength measured at a specific b-value and *S(0)* is the signal at b_0. By employing a fitting algorithm, the resulting IVIM curve is obtained. Because the magnitudes of D and D^* are comparable, the IVIM formula can be solved, resulting in estimates of the key parameters.

$$\frac{S(b)}{S(0)} = (1 - f_p)e^{-bD} + f_p e^{-bD^*} \tag{1}$$

Limitations to the quantification process of IVIM MRI concern technical, standardization and computational demands. For instance, the accurate fitting of the bi-exponential model requires high-quality data and is sensitive to noise, especially at high b-values. Therefore, future advancements in IVIM MRI quantification are likely to focus on the development of more robust fitting algorithms, machine learning approaches for automated parameter estimation, and standardized protocols to facilitate wider clinical adoption.

3.1.4 Fitting the curve

Typically, conventional Least-Squares (LSQ) fitting is used to estimate the parameter maps. To enhance the robustness of fitting, segmented fitting approaches first apply a low b-value threshold to separate perfusion and diffusion components, subsequently performing piecewise fitting. This method reduces the influence of noise but may introduce bias in parameter estimation.

State-of-the-art fitting algorithms include Bayesian fitting algorithms, such as Bayesian Inference (BI). Bayesian approaches incorporate prior knowledge about the expected parameter distributions to regularize the fitting process, potentially improving the stability and reliability of parameter estimates, especially in data with low signal-to-noise ratios (SNR). However, these methods are computationally exhaustive.

Recent advances have seen the application of machine learning and deep learning for IVIM parameter estimation. These models, trained on large datasets, can potentially offer robust, automatic fitting across diverse tissue types and conditions. Noteworthy is the approach presented by Barbieri et al., 2020, which is a Physics-Informed Deep Neural Network (PI-DNN) that is trained by minimizing the Mean Squared Error (MSE) between the measured signal and the signal retrieved from the estimated parameters. This approach outperforms the conventional LSQ and achieves a (marginally) better performance than the Bayesian approach, while operating substantially faster.

A major limitation of the PI-DNN is that it requires the input data to have the same

size. As there is no uniform b-value protocol, this is problematic for clinical adoption. In order to resolve this issue, the signal decay curve can be modelled as a Neural Controlled Differential Equation (NCDE), which is a continuous-time extension of Recurrent Neural Networks (RNNs) (Morrill et al., 2021). A neural network is used as learning algorithm for the CDE, which preserves the computational efficiency of neural networks but enhances user flexibility. Yet, it demonstrates consistent confidence in its predictions, which introduces a significant concern concerning the model's performance on out-of-distribution data. Such circumstances could lead to images that vary from reality, potentially resulting in incorrect clinical diagnoses.

3.2 Data

We use in-vivo data of 19 individual patients from multiple studies, which are stored in a Neuroimaging Informatics Technology Initiative (NIfTI) format, which is a common data format for this type of data. The region of interest is the abdomen. The data is acquired using a Philips 3.0T scanner and constitutes of the measured signal decay as a function of the b-values, resulting in a 3D object over the timepoints of the measurements. The 4D tensor is acquired on a per measurement per slice basis, so each slice-measurement combination is unique. The measured signal ranges from approximately 1 at $b = 0$ to near 0 for high b-values.

3.2.1 Characteristics

The studies used varying b-value protocols and x-, y- and z-dimensions. Therefore, without any preprocessing the patients are not directly comparable. To obtain some insights into the differences between studies, we explore the dimensions of each dataset, which are previewed in Table 1.

For most studies, the x- and y-dimensions, which define the shape of a slice, are around 144. However, some studies for instance have a much smaller y-dimension than x-dimension. Regarding the number of slices - the z-dimension, the value typically is around 18. Some studies, however, contain up to 27 slices. The varying dimensions lead to a highly fluctuating total number of voxels as well as the number of valid voxels,

which strongly affect the computational demands required to analyse that patient.

Furthermore, the b-value protocols were variable, as can be observed in Table 2. Both the values of the b itself as the number of repeated measurements for a specific b-value strongly differ. Therefore, the signal decay curves of different patients do not necessarily contain directly comparable information, since the type of information captured heavily relies on the magnitude of b-value at hand. On top of that, the quality of the scan also depends on the b-value scheme employed.

Lastly, the tissue region that is scanned during the acquisition varied per patient. This does not directly influence the level of flexibility needed for the model architecture, as is the case with the previously mentioned variations, yet the increased inter-patient variability could decrease model performance.

Table 1: The x/y/z dimensions of the in-vivo data from 19 different studies and corresponding numbers of (valid) voxels.

Patient	**Study**	**x**	**y**	**Nr slices**	**Total voxels**	**Valid voxels**
1	NAPAN/P1	144	144	18	373.248	85.126
2	NAPAN/P2	144	144	18	373.248	76.435
3	NAPAN/P3	144	144	18	373.248	83.112
4	NAPAN/P4	144	144	20	414.720	103.016
5	MIPA/P1	144	36	18	93.312	50.666
6	MIPA/P2	144	36	18	93.312	60.226
7	MIPA/P3	136	36	18	88.128	60.771
8	MATRIX/P1	142	36	18	92.016	44.351
9	ANCHOR/P1	256	256	27	1.769.472	653.635
10	ANCHOR/P2	256	256	27	1.769.472	617.044
11	ANCHOR/P3	256	256	27	1.769.472	720.123
12	ASAP_volunteer/P1	144	144	18	373.248	81.795
13	NEPHROX/P1	144	144	18	373.248	203.396
14	NEPHROX/P2	144	144	18	373.248	190.930
15	Pyqmri_kidney_train/P2	176	176	18	557.568	118.173
16	Pyqmri_kidney_train/P4	176	176	18	557.568	126.131
17	REMP/P1	144	36	18	93.312	44.093
18	REMP/P2	119	36	18	77.112	43.406
19	REMP/P3	141	36	18	91.368	39.514

Table 2: The b-value protocols employed in the 19 different studies.

Patient	b-values	Repetitions
1	[0, 5, 10, 25, 50, 75, 150, 450, 600]	[9, 6, 6, 6, 6, 6, 6, 6, 6]
2	[0, 5, 10, 25, 50, 75, 150, 450, 600]	[9, 6, 6, 6, 6, 6, 6, 6, 6]
3	[0, 5, 10, 25, 50, 75, 150, 450, 600]	[9, 6, 6, 6, 6, 6, 6, 6, 6]
4	[0, 5, 10, 25, 50, 75, 150, 450, 600]	[9, 6, 6, 6, 6, 6, 6, 6, 6]
5	[0, 10, 20, 30, 40, 50, 75, 100, 150, 250, 400, 600]	[15, 9, 9, 9, 9, 9, 4, 12, 4, 4, 4, 16]
6	[0, 10, 20, 30, 40, 50, 75, 100, 150, 250, 400, 600]	[15, 9, 9, 9, 9, 9, 4, 12, 4, 4, 4, 16]
7	[0, 10, 20, 30, 40, 50, 75, 100, 150, 250, 400, 600]	[15, 9, 9, 9, 9, 9, 4, 12, 4, 4, 4, 16]
8	[0, 10, 20, 30, 40, 50, 75, 100, 150, 250, 400, 600]	[15, 9, 9, 9, 9, 9, 4, 12, 4, 4, 4, 16]
9	[0, 1, 2, 5, 10, 20, 30, 40, 50, 75, 100, 150, 200, 300, 400, 500, 600, 700]	[9, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]
10	[0, 1, 2, 5, 10, 20, 30, 40, 50, 75, 100, 150, 200, 300, 400, 500, 600, 700]	[9, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]
11	[0, 1, 2, 5, 10, 20, 30, 40, 50, 75, 100, 150, 200, 300, 400, 500, 600, 700]	[9, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]
12	[0, 5, 10, 25, 50, 75, 150, 450, 600]	[9, 6, 6, 6, 6, 6, 6, 6, 6]
13	[0, 2, 4, 8, 12, 18, 24, 32, 40, 50, 75, 110, 200, 300, 450, 600]	[15, 9, 9, 9, 9, 9, 9, 9, 9, 9, 15, 9, 9, 9, 9, 15]
14	[0, 2, 4, 8, 12, 18, 24, 32, 40, 50, 75, 110, 200, 300, 450, 600]	[15, 9, 9, 9, 9, 9, 9, 9, 9, 9, 15, 9, 9, 9, 9, 15]
15	[0, 10, 20, 35, 50, 75, 100, 200, 400, 600]	[16, 6, 6, 6, 6, 6, 6, 32, 32, 32]
16	[0, 0.1, 10, 20, 35, 50, 75, 100, 200, 400, 600]	[1, 15, 6, 6, 6, 6, 6, 6, 32, 32, 32]
17	[0, 10, 20, 30, 40, 50, 75, 100, 150, 250, 400, 600]	[15, 9, 9, 9, 9, 9, 4, 12, 4, 4, 4, 16]
18	[0, 10, 20, 30, 40, 50, 75, 100, 150, 250, 400, 600]	[15, 9, 9, 9, 9, 9, 4, 12, 4, 4, 4, 16]
19	[0, 10, 20, 30, 40, 50, 75, 100, 150, 250, 400, 600]	[15, 9, 9, 9, 9, 9, 4, 12, 4, 4, 4, 16]

3.2.2 Preprocessing

Firstly, the data needs to be preprocessed. We start by excluding background voxels and noise by using a mask that filters voxels showing non-IVIM behavior. A main criterion for IVIM behavior is that S_0 is greater than the median of the signal. The resulting voxels are referred to as the valid voxels.

Since the data is stored in the form of flat arrays of the valid voxels over the measurements, the data has to be reshaped into multi-dimensional arrays. In order to do so, the valid voxel array is firstly padded to a full voxel array, containing exactly $x * y$ voxels per slice. This voxel array is then reshaped into an array with dimensions (x, y, z), where the z-dimension represents the slices. Consequently, the voxels are divided over the slices they belong to and their corresponding x- and y-coordinates are determined.

4 Methods

The main objective of this research is to investigate if and how deep learning techniques could be used to accurately and automatically detect artifacts in IVIM MRI. We divided the research into four sub-questions to examine different aspects of the primary objective. In this chapter, we elaborate on all the techniques used to retrieve answers to the following four research questions:

1. Can the residuals of an IVIM fitting algorithm direct us towards the presence of an artifact?

2. Can deep learning techniques provide a reliable solution for automated outlier detection in IVIM MRI?

3. Can incorporating spatio-temporal information of the data enhance our models' sensitivity to detecting artifacts?

4. What is considered essential information to facilitate reliable detection?

We conduct several experiments to be able to answer these questions, starting with question 1. In order to find an answer to this question, we explore multiple approaches to develop an architecture to automatically label the unlabeled IVIM MRI data. During this exploratory process, which is described in Section 4.1, we gain insights on the (potential) correlation between the data's residuals and the presence of artifacts. This first algorithm aims to function as a proof-of-concept for connecting artifacts to outliers and consequently automatically generates pseudo-labels for the input data to be used for an automated model, facilitating the development a more advanced and efficient second model. This second model is based on neural networks and aims to investigate the potential benefits of using deep learning for 4D outlier detection in IVIM MRI.

To answer question 3, we compare the outcomes and performance of different versions of the first model to the performances of spatial, temporal and spatio-temporal versions of the second model, which are all based on neural networks. These models are described in Sections 4.2.3 and 4.2.4.

In order to answer questions 2 and 4, we develop multiple versions of the deep learning-based outlier detection model. By comparing these different variants, we obtain insights

in which architecture is most suited for this task, how suitable the architectures actually are and what information is essential to secure reliability. In Section 4.2, we extensively elaborate on all versions of the outlier detection model and their evaluation criteria.

4.1 Data labeling

The goal in this first part is to develop a method to automatically label our data. Specifically, the labels in question indicate whether a certain slice (in the z-dimension) acquired during a given measurement is corrupted by artifacts or not. The aim is to identify and flag problematic slice-timepoint pairs, which will be labeled as problematic whenever they contain artifacts. These pairs are unique, which allows for direct re-scanning of the specific affected measurement. This would enable to reduce the level of distortion and improve the quality of the data.

In order to develop such a model, the relationship between outliers and artifacts should be investigated first. Hence, the model firstly performs a voxel-wise Least Squares fit to the in-vivo data in order to learn IVIM MRI specific parameters (D, D*, f and S0). As a second fitting option, the voxel-wise IVIM-Net, which is a Physics-Informed Deep Neural Network (PI-DNN) presented by Kaandorp et al., is used as fitting algorithm. Because it is less prone to outliers, it could provide a more robust approach. Furthermore, the measured signal intensities are divided by the value found at $b = 0$, which exhibits minor fluctuations around 1, in order to normalize the data. This normalization is common for the bi-exponential IVIM model in order to secure accurate parameter estimation, as the signal intensities depend on the b-values used, tissue type and specific imaging protocols.

After estimating the model parameters per voxel, as detailed in Section 3.1.4, the residuals are computed per voxel per measurement. The resulting residuals are used to direct us towards outliers, and hence potential artifacts. The performance of the two fitting models on detecting outliers will be compared by manually inspecting and assessing the high residual regions.

4.1.1 Fitting methods

Least Squares (LSQ)

The goal of the Least Squares (LSQ) fitting is to find the best-fitting curve or line through a set of points by minimizing the sum of the squares of the vertical deviations (residuals) from each data point to the curve. The parameters of the IVIM model, which determine the expected curve, are chosen to minimize the sum of squared residuals (SSR), of which the equation is shown in Formula 2. In this formula, y_i is the observed value and $\hat{y}_i$ is the predicted value of measurement i, which is computed using Formula 3, where b_i is the b-value of measurement i. By summing over the squared residuals, the value of SSR is obtained, which is minimized by using the $curve\_fit$ function from the $SciPy.optimize$ package. During the optimization, the parameters are constrained to physiologically plausible ranges of [0, 0.005], [0, 0.7], [0.005, 0.2] and [0.7, 1.3], for D, f_p, D^* and $S(0)$ respectively.

$$SSR = \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{2}$$

where

$$\hat{y}_i = \frac{S(b_i)}{S(0)} = (1 - f_p)e^{-b_i D} + f_p e^{-b_i D^*} \tag{3}$$

A downside of LSQ is its sensitivity to outliers since it squares the residuals. An outlier with a large deviation from the fitted model will have a disproportionately large squared residual, heavily influencing the overall fit. Neural networks could provide a more robust alternative, since they can be trained to recognize and minimize the influence of outliers through mechanisms like dropout, which reduces the dependency on any single data point, thus enhancing robustness. Moreover, generalization techniques add penalties for large coefficients, discouraging the model from fitting outliers excessively and ensuring that the model generalizes better to unseen data. Therefore, we consider IVIM-Net.

IVIM-Net

IVIM-Net is an unsupervised Physics-Informed Deep Neural Network (PI-DNN) initially proposed by Barbieri et al. in 2020, outperforming state-of-the-art IVIM fitting approaches at that time. In 2021, Kaandorp et al. proposed a substantially improved

version, the IVIM-Net$_{optim}$. In vivo, it showed significantly less noisy parameter maps than conventional fitting algorithms. Therefore, it is more robust to outliers, which makes it a more accurate and consistent IVIM fitting algorithm.

In Figure 1, the IVIM-Net architecture is depicted for different hyperparameter options. Here, the DWI signal data serves as the input and is processed through two distinct network architectures. In Design A, the network features a parallel configuration where each IVIM parameter is individually predicted by dedicated fully connected layers. Conversely, Design B uses a traditional single fully connected network approach. Blue circles within the diagram represent neurons selected at random for dropout, which helps prevent overfitting by omitting some network connections during training phases. The output layer is comprised of four neurons, equipped with either absolute or sigmoid activation functions that map directly to the IVIM parameters. The network then uses these outputs to predict the IVIM signal, a crucial step that informs the computation of the loss function.

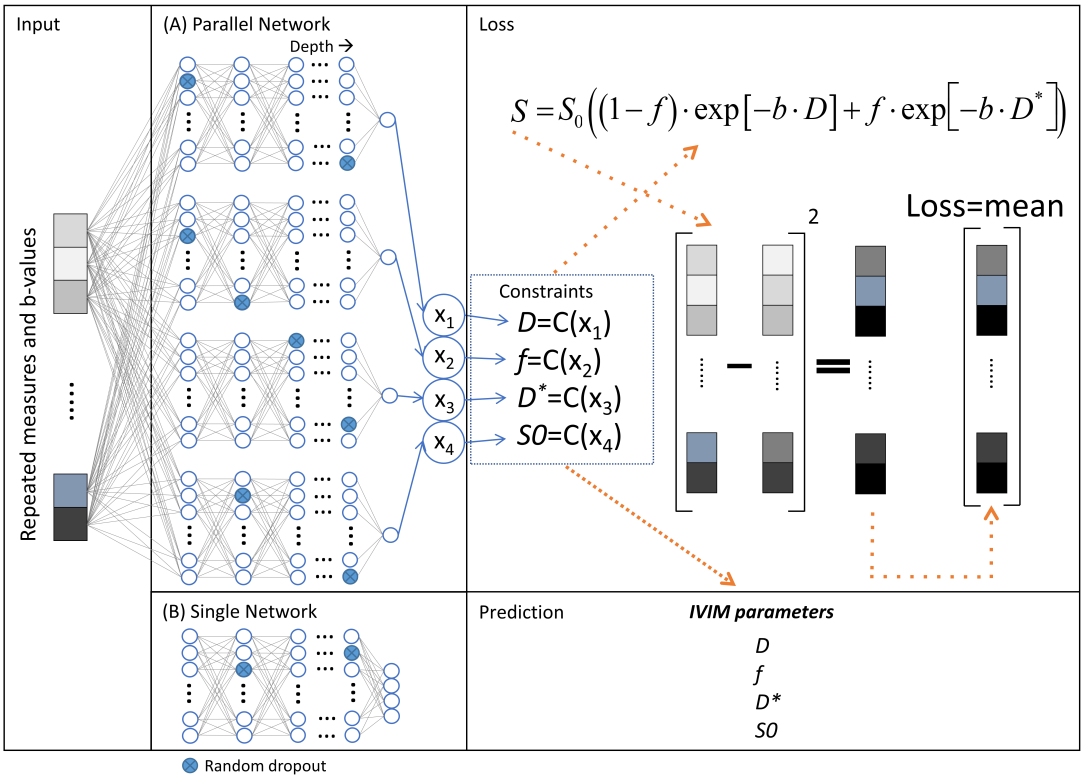


Figure 1: Representation of the IVIM-Net network design. (Kaandorp et al., 2021)

LSQ vs IVIM-Net

In order to evaluate the performance of the models using the different fitting algorithms, we investigate the residuals and inspect the predicted labels visually. An example of measured signal intensities with corresponding fitted IVIM curve for a given voxel is visualised in Figure 2. Since IVIM-Net is more robust to outliers than the conventional LSQ fitting, the distance between the curve and the outlier value will be enlarged and therefore, we expect IVIM-Net to exhibit a higher sensitivity to detecting outliers.

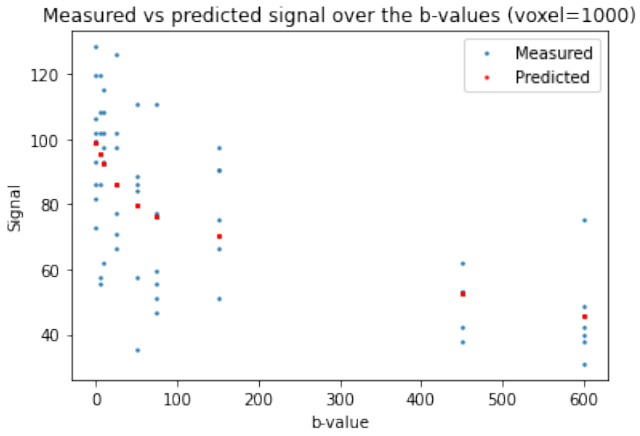


Figure 2: Measured signal intensities with corresponding fitted IVIM curve for a given voxel.

Implementation

Both fitting algorithms were already previously implemented by the research group, and only require some adaptions to the specific task of outlier detection. For instance, each measurement has to be considered independently, instead of aggregating measured signals for a given b-value. Furthermore, the residuals need to be determined per voxel per measurement, instead of solely considering the sum of squared residuals summed over all measurements and slices.

The LSQ algorithm is implemented by using the *scipy.optimize* library. The model to be fit is the bi-exponential IVIM model as described in Section 3.1.3. The loss function that is used for fitting the curve is the Mean Squared Error (MSE) loss, of which the formula is shown in Formula 4, where N is the voxel's number of measurements. While fitting the curve per voxel, the parameter estimates are determined by solving the bi-exponential IVIM model.

$$MSE = \frac{SSR}{N} \tag{4}$$

The IVIM-Net is implemented using the *PyTorch* library, which is proven to be computationally efficient for deep learning tasks. We use the IVIM-Net$_{optim}$ design to perform a voxel-wise fit on all valid voxels per patient. Besides the background and noise mask applied while preprocessing the data, voxels exhibiting non-IVIM-like behaviour are filtered, according to the implementation of Kaandorp et al. (2021). The additional criteria for valid voxels in this section are:

1. The 95th percentile of signals with $b < 50$ is smaller than 1.3.
2. The 95th percentile of signals with $b > 50$ is smaller than 1.2.
3. The 95th percentile of signals with $b > 150$ is smaller than 1.0

The loss function used is a physics-based loss function, shown in Formula 5, that computes the root MSE (RMSE) loss between the measured signal and the predicted IVIM signal. The predicted signal is obtained by inserting the predicted parameter estimates into the normalized IVIM model. The network has a depth of 2 and a width that equals the number of b-values. Furthermore, the model is trained during 1000 epochs using the *Adam* optimizer with a learning rate of 0.00003, a batch size of 128 and a maximum of 500 iterations per epoch. Moreover, there is an early stopping criterion after 10 bad epochs, a dropout rate of 0.1, batch normalisation and *Sigmoid* activation for the output.

$$loss = \sqrt{\frac{1}{N}\sum_{i=0}^{N}(\frac{y_i}{y_0} - \hat{y}_i)^2} \tag{5}$$

with

$$\hat{y}_i = (1-f)e^{-bD} + fe^{-bD^*} \tag{6}$$

Consequently, after obtaining parameter estimates using both models, the residuals per measurement are determined for each voxel by subtracting the predicted signal from

the measured signal.

4.1.2 Scaling methods

The distribution of the resulting residuals, of which an example is visualised in Figure 3, is anticipated to be dependent on the b-value. This assumption is primarily underpinned by the fact that the distribution of noise evolves as the b-value increases. At lower b-values, where the signal intensity is higher, the predominant source of noise is background noise, typically following a Gaussian distribution. As the b-value increases, the Signal-to-Noise Ratio (SNR) drops and the signal converges to a Rician noise floor. Moreover, for certain types of artifacts we expect that they occur more often at higher b-values, which increases the level of distortion. In the figure, we observe a decreasing variance over the range of b-values, except at the highest b-value.

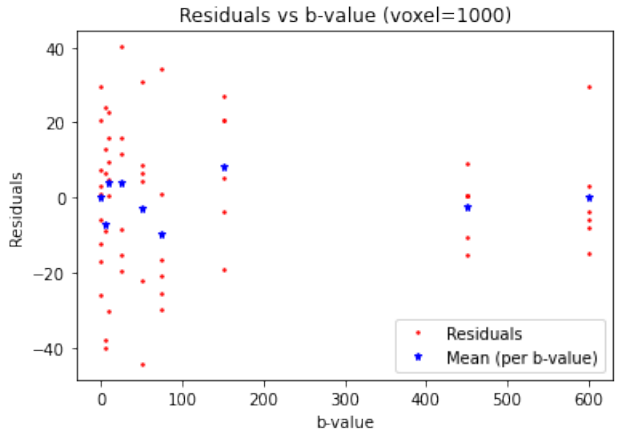


Figure 3: Residuals resulting from LSQ fitting a given voxel.

Secondly, the range of residuals is expected to vary across different Regions-Of-Interest (ROIs). A given voxel, which captures one or more specific tissue type(s), may be more or less susceptible to a particular artifact than another voxel. For instance, a voxel centered in the liver is generally more affected by respiratory motion compared to one in the spinal cord. Furthermore, voxels located near tissue borders tend to be fuzzier and more challenging to predict, which contributes to greater uncertainty in predictions, resulting in a higher variance of the residuals.

Therefore, we want to scale the residuals in a way that enables to quantitatively compare residual values across different b-values and within various ROIs, as both the b-value and voxel location correlate with the expected magnitudes of residuals. Hence, cancelling out the dependency on the b-value and the location is an essential step to facilitate an informed decision on which measurements are outliers. To achieve this, we examine multiple scaling methods, which are described in subsequent paragraphs.

Relative residuals

Firstly, we explore relative residuals, which are retrieved by dividing the residuals for each voxel for a certain b-value by the mean residual value of that voxel over all measurements for that b-value. This is done by using Formula 7, where $R_{x,vox}$ is the residual value for voxel vox at measurement timepoint x. T is the set of all measurement timepoints and b_x is the b-value at which measurement x was scanned. The resulting relative residuals are then independent of the expected residual magnitudes per b-value.

$$\begin{gathered} \text{relative } \mathrm{R}_{x,vox} = \frac{R_{x,vox}}{mean(R_{vox} \in T_B)} \qquad (\forall x \in T, vox \in V) \\ \text{where,} \quad R_{vox} \in T_B := R_{t,vox} \big| t \in T[b_t = b_x] \end{gathered} \tag{7}$$

A downside of this method is that if the mean residual value - the denominator - has a magnitude smaller than 1, the resulting relative residual value would be enlarged. Especially if the mean value approaches 0, it would lead to dramatically enlarged, or even exploding, relative residuals.

Min-max scaling

Hereafter, we explore the option of using a *MinMaxScaler* per group of measurements with equal b-value (b-group), in order to control the value ranges and calibrate the ranges between different b-groups. We want the minimum residual value to be 0 and the maximum residual value to be 1. Hence, these are the range limits used during the min-max scaling.

The scaled residuals are obtained using the equations previewed in Formula 8. First, the residuals are standardized per b-group. Then, the min-max scaling happens through

multiplying the standardized residual value by the predefined min-max range and adding its lower bound.

$$\begin{aligned} \text{standardized } \mathrm{R}_{x,vox} &= \frac{R_{x,vox} - min(R_{vox} \in T_B)}{max(R_{vox} \in T_B) - min(R_{vox} \in T_B)} \qquad (\forall x \in T, vox \in V) \\ \text{where,} \quad R_{vox} \in T_B &:= R_{t,vox} \big| t \in T[b_t = b_x] \\ \text{MinMax } \mathrm{R}_{x,vox} &= \text{standardized } \mathrm{R}_{x,vox} * (max - min) + min \end{aligned} \tag{8}$$

The forced range used in this method has the advantage of allowing for direct comparison between different b-groups. However, scaling the residuals to a predefined range can cause misleading results. Scaling the highest deviating residuals to (close to) 1 suggests that these residuals are most likely to be outliers. However, it does not tell us how deviating these measurements truly are, since this information from the original distribution is lost. Therefore, we need to be cautious with setting a predetermined threshold to which the min-max scaled residuals will be subjected.

Absolute min-max scaling

Because we are interested in the absolute deviation, originating from a sudden and unexpected signal dropout or increase, we test the absolute value of the *MinMaxScaler* as a scaling method. This is done by taking the absolute values of the residuals, as done in Formula 9, before applying the *MinMaxScaler*.

$$R_{x,vox} = \left| R_{x,vox} \right| \qquad (\forall x \in T, vox \in V) \tag{9}$$

We hypothesize that using the absolute residual values reduces the amount of information lost by scaling the residuals into a fixed range. Using the previous method, the smallest residual value (negative) will be scaled to 0, suggesting no deviation, which is not true and leads to misinterpretations. Although the original distribution of the residual data is preserved less compared to the previous method, we hypothesize that using this method will yield more insightful results.

Z-score

Using the z-score is a commonly used scaling technique. It indicates the number of

standard deviations that a certain measurement or value is above the mean. Therefore, z-scores ranging between -2 and 2 are considered normal, as they encompass approximately 95% of the data in a normal distribution. However, we consider the absolute z-score, since we are again solely interested in the absolute deviation to label a value as outlier.

In this research, the (absolute) z-score per voxel is computed in two different ways. The first is by determining the z-score for that voxel over all measurements, for which the formula is shown in Formula 10. The second is to compute it over all measurements with an equal b-value, previewed in Formula 11. By using the second approach, the resulting scores tell how deviating a certain value is within its b-group. More specifically, they tell how many standard deviations it is away from the expected value for that voxel at that b-value.

$$\text{abs } Z_{x,vox} = \left| \frac{R_{x,vox} - mean(R_{vox})}{std(R_{vox})} \right| \qquad (\forall x \in T, vox \in V) \tag{10}$$

$$\begin{aligned} \text{abs } Z_{x,vox} &= \left| \frac{R_{x,vox} - mean(R_{vox} \in T_B)}{std(R_{vox} \in T_B)} \right| \qquad (\forall x \in T, vox \in V) \\ \text{where,} \quad R_{vox} \in T_B &:= R_{t,vox} \big| t \in T[b_t = b_x] \end{aligned} \tag{11}$$

After determining a z-score per voxel per measurement, we compute the mean z-score per slice per measurement in order to quantify the quality of that slice for that specific measurement. If the mean z-score is above a certain threshold, the slice is labeled as 'poor', meaning that it is corrupted to such an extent that it is favorable to filter it out or re-scan it.

We hypothesize that high z-scores originating from the fitting algorithms will point us towards outliers, which on their turn will likely point us to regions with a high artifact potential. Hence, by finding a suitable scaling method for the residuals, the resulting model is expected to aid us in the process of labeling in-vivo data, in an automatic way.

Threshold

Before being able to label slice-measurement pairs as poor, we need to determine a suit-

able threshold. However, the choice of threshold is dependent on the eventual scaling method to be used. The first three scaling methods might call for a threshold ratio instead of a predetermined value. For the relative residuals, this is because the values are not necessarily within a fixed range and can fluctuate quite a lot. For the (absolute) min-max scaling, this is because the maximum value is equal for each b-group, although the level of deviation can be very different. For the absolute z-score, however, an equal value means an equal level of deviation. A voxel with a value above 2 means that that certain measurement deviates more than expected and, therefore, more than favorable. Considering the mean z-score value of a whole slice, we certainly do not want the slice to, on average, deviate more than one standard deviation from the expected residual value for that specific measurement. Therefore, we want the mean z-score of a slice-timepoint pair to range between 0 and 1 for all b-groups. Hence, the threshold for the mean z-score is 1 during this research.

In all cases, slice-measurement pairs with a mean scaled residual value greater than the determined threshold are outputted as outliers. By qualitatively evaluating the outputs, we try to validate this first model as a proof-of-concept that scaling residuals could lead us to outliers that could point at artifacts. In that way, the z-scores generated by the first model could function as pseudo-labels for the data to be used in the second model.

4.1.3 Local versus global

Moreover, in order to gain insights into the types or artifacts that are detected or labeled as outliers, we investigate how local or global the deviation is. This is done by determining how many voxels in a specific slice at a specific measurement exhibit a scaled residual value higher than the threshold, which leads to an integer value equalling the sum of poor voxels within that slice. Since this value is expected to correlate with the number of valid voxels present in that slice, the sum is divided by this number. Moreover, how many voxels are corrupted is expected to be somewhat correlated with how corrupted the slice is. Therefore, the sum is consequently divided by the corresponding mean scaled residual value of that slice. As a result, we obtain the ratio of corrupted voxels within a slice that is independent of how corrupted the slice is.

We hypothesize that the resulting ratio will differentiate local artifacts from global artifacts, which would provide insights into the type of artifact that is detected.

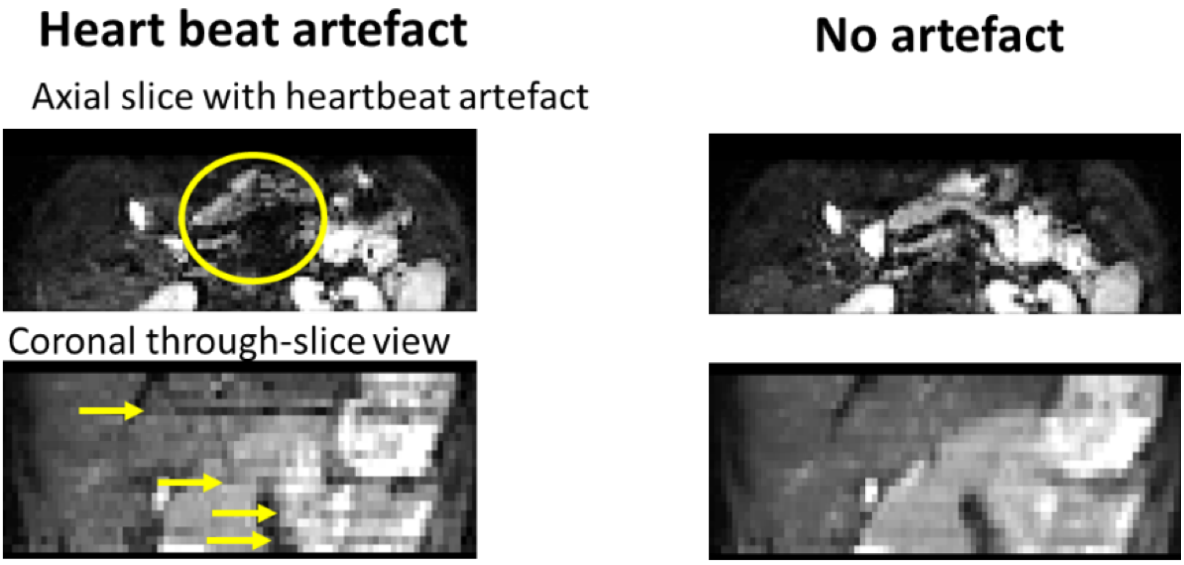
4.1.4 Evaluation

In order to evaluate the model's performance in terms of how accurately it is able to detect artifacts, a visual inspection is performed. This is done by manually labeling the slices across all measurements using the labels previewed in Table 3. These artifacts usually exhibit visually recognizable patterns in the scans, which are detailed in the table.

Table 3: The artifact labels used during the visual inspection.

Artifact label	**Visual pattern**
Interleaved Motion Artifact	Interleaved pattern in coronal view
Motion Artifact	Blurry or dislocated slice
Local Signal Dropout	Black region
Local Increased Signal	Bright region

Firstly, the data of a given patient is visualised using *MeVisLab*, which is a medical image analysis and visualisation framework. The *OrthoView2D* method allows us to visualise a slice from multiple views simultaneously. We are predominantly interested in the axial view, which is the dimension the slices are acquired in during scanning, since it provides the most valuable insights in locally deviating voxel regions. Moreover, it corresponds to exactly one measurement, since the measurements are acquired slice-wise. However, the coronal view is very useful for detecting Interleaved Motion Artifacts, since this type of artifact causes jagged edges in an interleaved pattern in the coronal dimension, representing the dislocated slices. An example of two slices with and without an artifact in both axial and coronal view is visualised in Figure 4. In this figure, the yellow arrows are pointing at slices containing artifacts, which can be recognized by the significantly darker regions.



(a) Scans with heartbeat artifact.

(b) Scans without artifact.

Figure 4: An example of two slices in axial and coronal view with (a) and without (b) artifact. (Gurney-Champion, KWF Proposal)

The process is executed by for each slice looping through all timepoints, which represent the measurements in chronological order. At each timepoint, the current slice is compared with (at least) its two neighboring slices as well as with the measurements of the current slice at timepoints with an equal b-value to the current one. In the coronal view, we check whether the current slice at the current timepoint is dislocated compared to neighboring slices. Besides that, we check whether it contains any unexpectedly dark or bright spots. Next, the slice-timepoint pairs are labeled using the labels from Table 3. An important note is that a pair can receive zero or more labels.

For the sake of time, we decided to evaluate, and therefore to manually label, the model performance on the first three patients. We start by determining for each timepoint whether it contains at least one poor slice and then comparing our verdict to the model's output. Subsequently, we visually inspect all slice-timepoint pairs that were labeled as outliers by the model and label them according to the four previously mentioned artifact labels.

Limitations to this approach of evaluation include the inevitable subjectivity, the labor-intensity and therefore, the lack of scalability. However, because of the lack of ground truth labels in our data, benchmarks or a golden standard solution, we for now consider

this evaluation as sufficient to be able to draw a conclusion on whether the first model could function as proof-of-concept. Eventually, we are seeking for similar patterns in the slices labeled as outliers by the model and the slices labeled as containing artifacts during the visual inspection. This would testify of a correlation between outliers detected by our model and the presence of artifacts.

4.2 Outlier detection

In order to resolve the limitations inherent to deploying the previously proposed data labeling model, such as the scalability, we aim to develop a deep learning approach that approximates the performance achieved by our current 'golden standard'. Besides striving for good performance, a primary criterion of deeming such a model successful is the scalability, since this is a crucial aspect for clinical adoption. The first model operates in essence on a per-patient per-voxel basis by running the fitting algorithms, as well as deploying the scaling methods, voxel-wise. Furthermore, the model relies on completeness of the data of a given patient to be able to scale the residuals in an informed way, thereby ensuring reliability, giving that scaling incomplete (residual) data could lead to biased results. Lastly, the first approach does not exploit all information captured in the spatial dimension since it operates voxel-wise, whereas we hypothesize that doing so could provide valuable insights.

Deep learning could address these limitations by providing a faster and more flexible alternative. Therefore, we aim to develop a suitable deep learning model for automatic 4D outlier detection. This model takes the 4D IVIM MRI data as input, exploiting both the spatial and the temporal dimensions. Subsequently, it tries to approximate the previously obtained scaled residuals from the first model by learning the relationship between signal intensities and scaled residuals.

4.2.1 Data preparation

Slices

First, the data needs some additional preparation on top of the preprocessing that has been performed in Section 3.2.2. For each patient, we start by padding and/or cropping the slices into a uniform shape of 144×144. If the original slice has a x- and/or

y-dimension smaller than 144, we pad it by adding rows or columns containing zeros to either side of the slice. If the slice is wider or higher than 144, we crop the slice by removing an equal number of rows or columns from each side of the slice. Each padding or cropping operation that is performed has to be passed on to the (valid) voxel arrays too through adding and/or removing the voxels in question.

b-schemes

Moreover, we want to standardize the b-value schemes across the patients to enhance the comparability. Therefore, we want to select a uniform b-value scheme of 9 unique b-values with 6 repeated measurements per b-value, resulting in a total of 54 measurements. This b-value scheme is chosen by balancing the greatest common denominator across the patients and retaining as much data as possible. The selection procedure starts by filtering all b-values that have at least 6 repetitions. If this results in 9 b-values, these are returned as final b-values. If it results in more than 9 b-values, the first 5 and last 4 b-values are selected, since this selection is deemed to cover the range of b-values well.

If both scenarios are not the case, we combine remaining b-values until we reach 9 (combined) b-values with at least 6 repeated measurements each. We iterate through the remaining b-values in an ascending order. As long as the (combined) number of repetitions of the currently selected b-values combined is less than 6, we add the next left-over b-value to the current selection. When the selection reaches 6 repetitions, the new, combined b-value is determined by taking the weighted mean of the selected b-values. For instance, if the selection contains 3 repeated measurements for $b = 100$ and 3 for $b = 150$, the newly combined b-value will be 125, with 6 repetitions. By combining b-values in an ascending order, we aim to minimize the impact of adapting the b-value scheme by ensuring that the combined b-values selections are as close to each other as possible.

For all selected b-values we then take the 6 repetitions and we remove all measurements acquired at other b-values or during other repetitions, which leaves us with exactly 54 measurements.

Multidimensional arrays

Thereafter, for each patient the data is reshaped into an array with shape (nr_measurements,

nr_slices, $x * y$). Moreover, the data is split per b-value and per b-value per slice, resulting in two arrays with shapes (nr_bvalues, nr_repetitions, nr_voxels) and (nr_bvalues, nr_repetitions, nr_slices, $x * y$) respectively. Furthermore, these arrays are reshaped in order to obtain the data split per repetition and per repetition per slice, resulting in arrays with shapes (nr_repetitions, nr_bvalues, nr_voxels) and (nr_repetitions, nr_bvalues, nr_slices, $x * y$).

Data labels

Regarding the data labels, we start by loading the scaled residual values that were outputted and stored by the first model. Loading theses values is done using a *csv.reader*. These values are stored in an array with shape (nr_measurements, nr_slices, x, y), where the number of measurements, x- and y-dimension are, at this point, the original and unedited dimensions. Hereafter, this array is adapted according to the b-scheme standardization, cropping and padding as previously described. Furthermore, the scaled residuals are split per b-value and the mean residual value per slice per b-value is determined. Moreover, the slice labels are determined by subjecting the mean residual value per slice to the predetermined threshold. These labels are stored in an array with shape (nr_measurements, nr_slices), where the number of measurements at this point is 54.

Lastly, all measurements, and hence all previously obtained arrays, are ordered on ascending b-value, mimicking the signal decay curve.

4.2.2 Class imbalance

An important characteristic of the data that we need to take into account is the severe class imbalance. Since we are interested in detecting artifacts, which is the positive class, we want to focus on the minority class. Therefore, we need to account for this imbalance by using classification metrics that have sufficient focus on the minority class. For instance, a model's accuracy can be very high even though only predicting negative labels, as long as the class imbalance is strong enough. Therefore, we take both the loss as the types of error made into account when evaluating the models presented in the next sections.

4.2.3 Baseline models

We start by implementing a baseline model in order to obtain a benchmark performance. We use the simplest case, which is a fully-connected Neural Network (NN) with one hidden layer that classifies slice-measurement pairs as outlier/no outlier. It uses one patient per batch and takes the flattened voxel array as input, represented as 1D tensor instead of a 2D slice. This has two major downsides. First, a significant proportion of the spatial information is lost. Second, the fully-connected network will have a large number of coefficients, which strongly affects the computational efficiency.

Furthermore, we hypothesize that this model is not complex enough to learn all the hierarchies needed. Nevertheless, it is used to determine a benchmark for the expected performance and to learn what architectural choices could improve model performance. Therefore, we implement the baseline model and two more variants of it to investigate how certain types of additional information affect the model performance.

Base classifier

The *Base model* is a fully-connected NN with one hidden layer that operates as a classifier. It uses *ReLU* non-linearity, *Sigmoid* activation, a *Binary Cross-Entropy* (BCE) loss function and *Stochastic Gradient Descent* (SGD) optimizer with a learning rate of 0.01. The *Sigmoid* activation forces the output values to be between 0 and 1, which is what we want for a classification task. Moreover, BCE loss is a commonly used loss function for binary classification problems in machine learning, which measures the performance of a classification model whose output is a probability value between 0 and 1. It calculates the difference between the true label (ground truth) and the predicted probability assigned by the model, as defined in Formula 12. Here, N is the number of samples, y_i is the actual label - either 0 or 1 - of item i and p_i is the predicted probability of item i being 1.

$$loss_{BCE} = -\frac{1}{N}\sum_{i=0}^{N}[y_i log(p_i) + (1 - y_i) log(1 - p_i)] \tag{12}$$

The model takes the signal decay per voxel as input and generates a binary label per slice per measurement as output. Furthermore, a 15-4 train-test split is performed on the pool of 19 patients. This results in a training array with shape (15, nr_measurements,

nr_slices, $x * y$) and testing array with shape (4, nr_measurements, nr_slices, $x * y$). The dimensions of the resulting model are:

- Input: (1, 54, 18, 20736)
- Hidden layer: (1, 54, 18, 20736) → (1, 54, 18, 20736)
- Output: (1, 54, 18, 20736) → (1, 54, 18, 1)

Two-class model

In a first attempt to compensate for the class imbalance, we implement the *Two-class model*. This model classifies each slice for every measurement into either the 'outlier' or 'no outlier' class. In order to do so, we make a few adaptions to the Base model. First, the *Sigmoid* activation is changed into a *Softmax* activation, such that for each instance the sum of the two output classes is exactly 1. Second, we use a *Weighted BCE* loss function, of which the formula is previewed in Formula 13, to account for the class imbalance. In this formula, w_0 and w_1 are the weights assigned to the positive and negative classes, respectively. This modification helps in focusing the training process more on the minority class, which is often underrepresented and thus typically less accurately predicted.

$$loss_{weightedBCE} = -\frac{1}{N}\sum_{i=0}^{N}[w_1 * y_i log(p_i) + w_0 * (1 - y_i) log(1 - p_i)] \tag{13}$$

The resulting dimensions of the Two-class model are:

- Input: (1, 54, 18, 20736)
- Hidden layer: (1, 54, 18, 20736) → (1, 54, 18, 20736)
- Output: (1, 54, 18, 20736) → (1, 54, 18, 2)

Grouped b model

In order to exploit the knowledge we have about the b-value at which a specific measurement is acquired, we implement the *Grouped b* model. This model has a separate input channel per b-value, which facilitates the model to learn features per channel. The data

has 9 unique b-values, which means that the model has 9 input channels. The number of repeated measurements per b-value is 6. Therefore, the resulting *Grouped b* model has the following dimensions:

- Input: (1, 9, 6, 18, 20736)
- Hidden layer: (1, 9, 6, 18, 20736) → (1, 9, 6, 18, 20736)
- Output: (1, 9, 6, 18, 20736) → (1, 9, 6, 18, 1)

Base regressor

Since we expect the Base model to be insufficiently complex to learn the relationship between signal intensities as input and a binary outlier label as output, we investigate whether learning a scaled residual values per voxel enhances performance. This is done by adapting the Base model by using a *Mean Squared Error* (MSE) loss function instead of the BCE loss function. Furthermore, for each voxel the *Base regressor* aims to approximate the scaled residual values over the measurements. The dimensions of the resulting Base regressor are:

- Input: (1, 54, 18, 20736)
- Hidden layer: (1, 54, 18, 20736) → (1, 54, 18, 20736)
- Output: (1, 54, 18, 20736) → (1, 54, 18, 20736)

Loss masking

In order to improve the Base regressor's efficiency, we use a loss mask when applying the MSE loss function. This mask filters irrelevant voxels, which enables the model to solely learn on valid voxels. This reduces the run time and minimizes the influence of irrelevant voxels on the learning process. The mask is generated by filtering the non-numeric, in this case *np.nan*, values in the data.

Training

The training is performed on a GPU. The models are trained using their corresponding loss function for various numbers of epochs, depending on their speed of convergence. The training algorithm we use is outlined in Algorithm 1. During every epoch, we

start by shuffling the training data and generating the training and validation batches. Patients 4, 5, 11 and 12 are always used for validation, the 15 remaining patients form the training set. The consistency in the train-validation split allows us to compare the performances of different runs or models.

During every epoch, we loop through all batches (patients in this case) in the training set and all batches in the validation set exactly once. For each training batch we take one training step, which starts by predicting its label for each slice-timepoint pair present in the batch. Using these predicted labels, we compute the loss and performance metrics, and store these. Then, we backpropagate the loss and update the weights of the model accordingly. After doing this for each batch, we determine the mean training loss over the batches and the mean training metrics, and we store these for the current epoch.

Algorithm 1 Training loop NN

```
for epoch in epochs do
    load data                                          ▷ training data is shuffled
    for batch in train_loader do
        predict values for current batch
        compute loss
        compute performance metrics
        store loss
        store metrics

        backpropagate loss
        update weights
    store training loss                                ▷ mean over training batches
    store training metrics                             ▷ mean over training batches
    for batch in val_loader do
        predict values for current batch
        compute loss
        compute performance metrics
        store loss
        store metrics
    store validation loss                              ▷ mean over validation batches
    store validation metrics                           ▷ mean over validation batches
return training results, validation results
```

Validation

Hereafter, we loop through the validation batches. Again, during each iteration we predict the labels for the current batch, we compute the corresponding loss and metrics and we store them. After this loop, we again determine the mean loss and metrics for the validation phase and we store them.

In the end, this training process aims to optimize the model parameters to approximate the labels obtained by the model presented in Section 4.1. Furthermore, the outputs of the models are qualitatively evaluated by visualizing the distributions of true and predicted labels.

4.2.4 CNN architectures

Among the various deep learning techniques, CNN is the most widely used as it is very similar to conventional NN. Unlike a typical NN, CNN processes an image as input and has three-dimensional arrangement of neurons that connect with a small region of the preceding layer rather than the entire layer, which makes the CNN computationally efficient. Therefore, CNNs are at the forefront of deep learning techniques for image processing, including outlier detection. Their ability to automatically and adaptively learn spatial hierarchies of image features from low- to high-level patterns makes them particularly suited for identifying atypical patterns in visual data.

The CNN consists of multiple layers including a convolutional layer, a non-linear activation layer such as a Rectified Linear Unit (ReLU) layer, a pooling layer or a fully connected layer. The convolutional layer performs a convolution operation between pixels of the input image and a filter, producing volumes of feature maps that capture the features extracted by the filter. The ReLU layer introduces non-linearity by applying the function $f(x) = max(0, x)$ to the input values, which increases non-linearity and accelerates training. The pooling layer reduces the spatial dimensions of the image by down-sampling the input values, which lowers computational costs and helps prevent overfitting while being translation invariant, as the operations depend on neighboring pixels. Typically, the last layer of a CNN is a fully connected layer, which resembles the hidden layers of a traditional NN in the sense that all neurons are connected to those in the preceding layer.

In this section, we present three different CNNs that each exploit the data along a different dimension. Moreover, we combine the outputs of the spatial and the temporal CNN in order to mimic spatio-temporal predictions. Each model takes the normalized signal intensities as input and aims to approximate the mean scaled residual value per slice as output.

Our model architecture is based on *LeNet* (LeCun et al., 1998). It is constituted by two subsequent blocks that each consist of two convolutional layers with *ReLU* activation followed by one *Max pooling* layer. After these two blocks, there are two fully connected layers, of which the first has a *ReLU* activation and *Dropout*. The exact dimensions of each layer are detailed per model separately.

CNN - channel per slice

The first CNN, of which the network architecture is visualised in Figure 5, processes the data by using a separate input channel per slice. Through this architecture, the CNN exploits the spatial information contained in each slice by performing 2D convolutions. The spatial information contained in the slice-dimension is captured, since each channel contributes to all nodes in the following layer.

The input data shape depends on the size and depth (channels) of our input images. Therefore, the shape of our input tensors is (1, 18, 144, 144). The output tensors have a shape of (1, 18, 1, 1). All intermediate dimensions can be found in the figure. This model allows a varying number of measurements as input, since each measurement is treated as separate tensor. Hence, in this case the full input tensor has a shape of (54, 18, 144, 144) per patient and the shape of the output tensor is (54, 18, 1, 1). The number of slices, however, has to be equal for all patients, since it determines the number of input channels of the model.

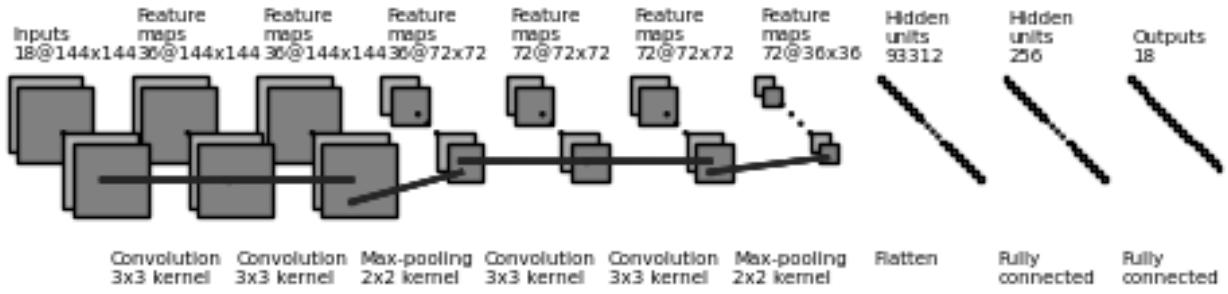


Figure 5: Representation of the CNN - channel per slice network design.

CNN - channel per measurement

The second CNN, of which the network architecture is visualised in Figure 6, processes the data by using a separate input channel per measurement. This CNN also exploits the spatial information within slices by performing 2D convolutions. However, this CNN captures the temporal information contained in the measurement-dimension instead of the spatial information from the slice-dimension.

The input data shape depends on the size of our input images and the number of measurements. Therefore, the shape of our input tensors is (1, 54, 144, 144). The output tensors have a shape of (1, 54, 1, 1). Again, all intermediate dimensions can be found in the figure. This model allows varying numbers of slices as input, but requires the number of slices to be equal for all patients. The full input and output tensors per patient have a shape of (18, 54, 144, 144) and (18, 54, 1, 1), respectively.

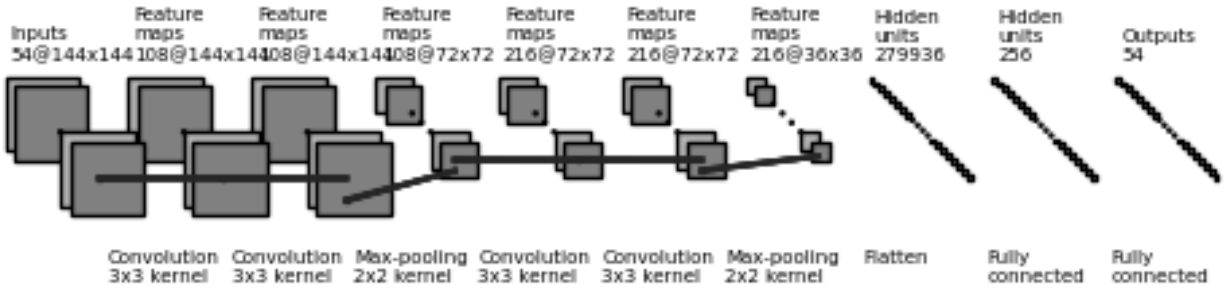


Figure 6: Representation of the CNN - channel per measurement network design.

CNN - channel per repetition

The third CNN, of which the network architecture is visualised in Figure 7, processes the data by using a separate input channel per repeated measurement per b-value, referred to as repetition. This CNN again exploits the spatial information within slices by performing 2D convolutions. However, this specific CNN captures the temporal information contained in the repetition-dimension.

The input data shape depends on the size of our input images and the number of repetitions per b-value. Therefore, the shape of our input tensors for this CNN is (1, 6, 144, 144). The output tensors have a shape of (1, 6, 1, 1). Again, all intermediate dimensions can be found in the figure. This model requires the b-schemes to be uniform for all patients. The full input and output tensors per batch have a shape of (batch_size, 6, 144, 144) and (batch_size, 6, 1, 1), respectively.

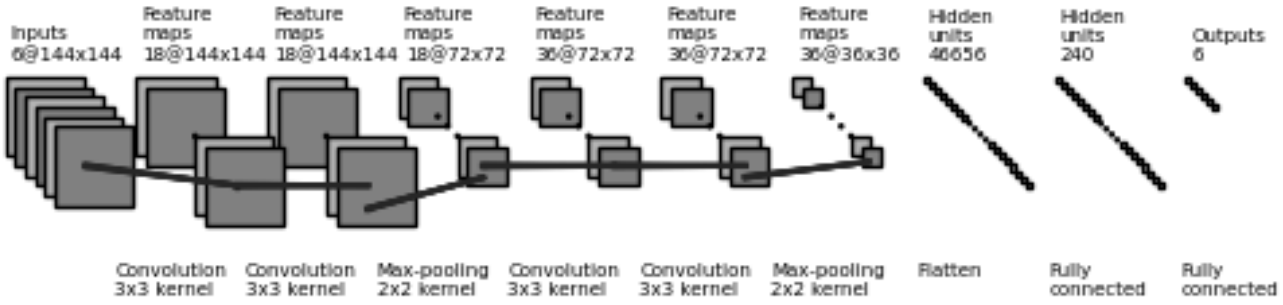


Figure 7: Representation of the CNN - channel per repetition network design.

Loss function

The CNNs are trained using the Mean Squared Error (MSE) loss function. The purpose of the training process is to minimize the MSE loss, thereby minimizing the (squared) difference between the actual scaled residuals and the predicted scaled residuals. The formula of the MSE can be found in Formula 14, where $\hat{y}_i$ are the predicted mean scaled residuals of batch element i, y_i are the actual mean scaled residuals and N is the number of elements in the current batch.

$$loss_{MSE} = \frac{1}{N}\sum_{i=0}^{N}(\hat{y}_i - y_i)^2 \tag{14}$$

Training

The training algorithm we use is the one previously outlined in Algorithm 1. Again, patients 4, 5, 11 and 12 are used for validation and the 15 remaining patients form the training set. For the CNN_{slices} (CNN - channel per slice) and the CNN_{meas} (CNN - channel per measurement) models holds that one batch equals one patient. For the CNN_{reps} (CNN - channel per repetition) models holds that one batch equals 64 independent tensors sampled from the data pool in order to increase regularization and enhance generalizability.

We expect that batch-wise training and shuffling the training data after each epoch will cause some regularization. This regularization arises from the fact that overfitting to a noisy batch will lead to decreased performance in other batches. Furthermore, as a form of L2 regularization, we use the *weight_decay* parameter of the *Adam* optimizer.

Adding weight decay can help prevent overfitting by ensuring that the regularization is adapted along with the learning rates. We compare the models with and without this form of regularization.

During every epoch, we loop through all batches in the training set and all batches in the validation set exactly once. For each training batch, we take one training step, which starts by predicting the mean scaled residual value for each slice-timepoint pair present in the batch. Using these predicted values, we compute the MSE loss and performance metrics, and store these. Then, we backpropagate the loss and update the weights of the model accordingly. After doing this for every batch, we determine the mean training loss and mean training metrics over the batches, and we store these for the current epoch.

Validation

Hereafter, we loop through the validation batches. Again, during each iteration we predict the values for the current batch, we compute the corresponding loss and metrics and we store them. After this loop, we again determine the mean loss and metrics for the validation phase and we store them.

In the end, this training process aims to optimize the model parameters to approximate the scaled residual values obtained through the 'golden standard' model. Furthermore, the outputs of the models are qualitatively evaluated by visualizing the distributions of the ground truth and the predicted values.

4.2.5 Evaluation

Performance metrics

The resulting models will be evaluated on how well they detect artifacts. The main objective is to approximate the 'golden standard' performance as achieved by the model presented in Section 4.1. As a baseline model, we use the *Base model* described in Section 4.2.3. In order to evaluate the performances of the models, we deploy several performance metrics, each assessing a different facet of the performance. All metrics are detailed in the next paragraphs.

Accuracy

Accuracy, as defined in Formula 15, represents the ratio of items that are classified correctly. In this formula, TP and TN are the number of True Positives and True Negatives, respectively. FP is the number of False Positives and FN is the number of False Negatives. It is the most basic performance metric, however it has the limitation to misrepresent performance in case of class imbalance. In such a case, a higher accuracy for the dominating class will overshadow the lower accuracy associated with the minority class thus providing biased results.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{15}$$

Sensitivity

Sensitivity is also known as the recall or the true positive rate, which is the probability of a positive label, conditioned on the class truly being positive. Therefore, it represents the proportion of poor slices in the ground truth that were correctly identified by the model and hence, it is computed using the formula shown in Formula 16. The sensitivity metric provides insights in the types of misclassification made by the model, since it is sensitive to under-classification as that results in low sensitivities.

$$Sensitivity = \frac{TP}{TP + FN} \tag{16}$$

Precision

Precision, as defined in Formula 17, represents the proportion of the poor slices as labeled by the model that match with the ground truth poor slices. Over-classification results in low precision scores and therefore, precision is a useful performance metric for our research.

$$Precision = \frac{TP}{TP + FP} \tag{17}$$

F1-score

Another performance measure that is used is the Dice score (F1-score). This is a com-

mon metric for medical image segmentation and classification tasks and measures the similarity between predicted labels and their ground truth. It determines the harmonic mean of the precision and sensitivity, as defined in Formula 18. High values for both measures means that the predicted classification labels match with the ground truth both in terms of detecting which slices are poor and the correctness of these slices' positive labels.

$$F1 = 2 * \frac{Precision * Sensitivity}{Precision + Sensitivity} \tag{18}$$

Weighted F1-score

The weighted F1-score is used to evaluate the performance of a classification model where there are imbalanced classes. It is particularly useful in scenarios where each class contributes unequally to the overall model accuracy, and it is important to consider the size (weight) of each class when calculating the metric. For a model dealing with multiple classes, the weighted F1-score is calculated by taking the F1-scores of each class and averaging them, weighting each by the proportion of true instances of each class, as defined in Formula 19. Here, C represents the number of classes, which is two in our case, the negative and positive classes. n_i is the number of items in class i, N is the total number of items over all classes and $F1_i$ is the F1-score of class i. The weighted F1-score provides a more accurate measure of model performance across all classes by considering both the abundance of each class and the individual class performance, thus offering a more nuanced view than simple unweighted F1-score.

$$WeightedF1 = \sum_{i=0}^{C} (\frac{n_i}{N} * F1_i) \tag{19}$$

Criteria

All performance metrics range between 0 and 1, and for all holds the higher, the better. What are considered good values for these metrics highly depend on the specific task, the nature and quality of the data, model architecture et cetera. Therefore, there is no specific benchmark for these values. Also, the type of task determines performance priorities. For some tasks, avoiding a false negative might be just as important as

achieving a high accuracy. In this case, the main goal is achieving a high F1-score and sensitivity, since a misclassification is not disastrous or life-threatening in our case, but we do aim to detect as many (true) artifacts as possible.

Eventually, the model that achieves the best performance is preferred. Moreover, we perform a thorough analysis that evaluates the models on how well they could solve the problem of detecting outliers. Finally, potential shortcomings of the model in terms of computational complexity and scalability, which are essential for potential clinical adoption, will be taken into account. For example, a long execution time would be disastrous for clinical adoption. After finishing this research, it should be clear whether these types of model architectures could provide a solution to the problem of automated artifact detection in 4D IVIM MRI, such that the research group can decide on continuing this research or seeking for any other solution.

5 Results

In this chapter we discuss the results obtained by the previously introduced models. We elaborate on the results of each model and highlight the key outcomes.

This chapter is divided into five different sections. First of all, we highlight the results of fitting the IVIM model and scaling the resulting residuals of the data labeling part described in Section 4.1. Hereafter, we investigate the outlier-artifact correlation by considering relevant results from the same section. Subsequently, we highlight the results from the deep learning models, obtained by the base models, as described in Section 4.2.3, and the CNNs, as in Section 4.2.4. Besides that, we discuss the potential benefits of using deep learning. Next, we try to assess the influence of incorporating spatio-temporal information into the models through considering the insights obtained from both parts of this research - the data labeling and the outlier detection. Finally, we discuss two primary criteria regarding the real-time applicability of the models.

The results of each model are evaluated both quantitatively and qualitatively using the criteria described in Sections 4.1.4 and 4.2.5.

5.1 Data labeling

5.1.1 Fitting the curve

Influence of fitting algorithm

In this section, we highlight the results of the fitting algorithms applied to patient 1. Both fitting algorithms - LSQ and IVIM-Net - are deployed on the patient's valid voxels, of which the mask is previewed in Figure 8.

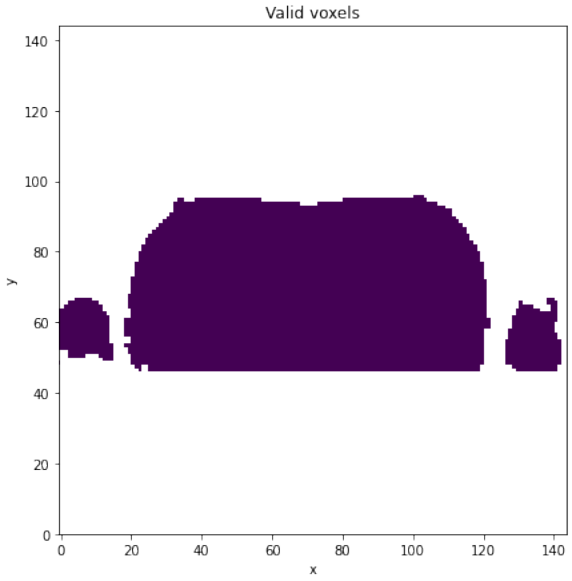
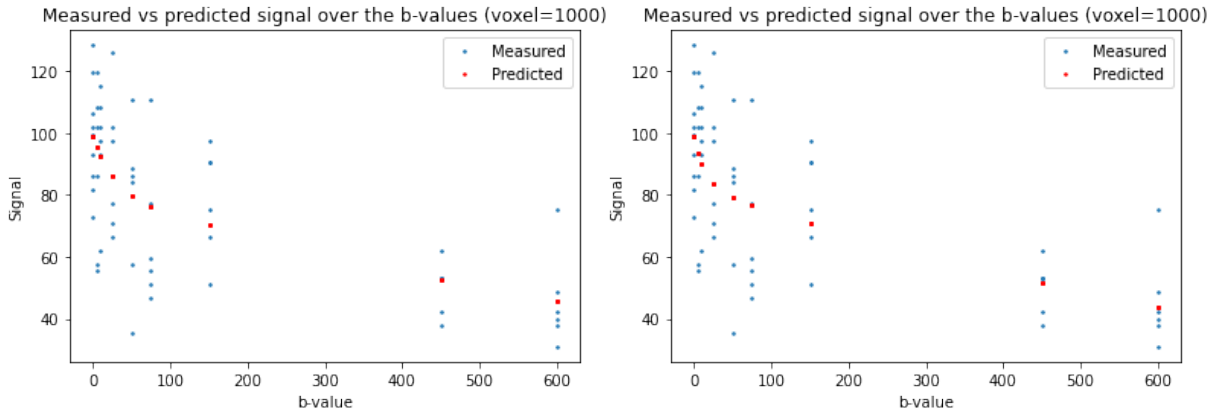


Figure 8: Valid voxel mask for patient 1.

After fitting the measured signal decay for each valid voxel, their resulting IVIM curves and corresponding IVIM parameters are obtained. In Figure 9, for both fitting algorithms both the measured signal and the predicted signal are visualised for voxel number 1000. What can be concluded from these figures is that at first sight the two curves look very similar. By analysing the two curves for multiple voxels and patients, we could observe minor differences between the shapes of the LSQ curve and IVIM-Net curve at the lowest b-values. These predicted values seemed less affected by the most extreme signal intensities measured at their corresponding b-values. At higher b-values the difference between the two curves, however, were very minimal. Summarising, we observe negligible differences between the two fitting algorithms in terms of the shape of the resulting curves.



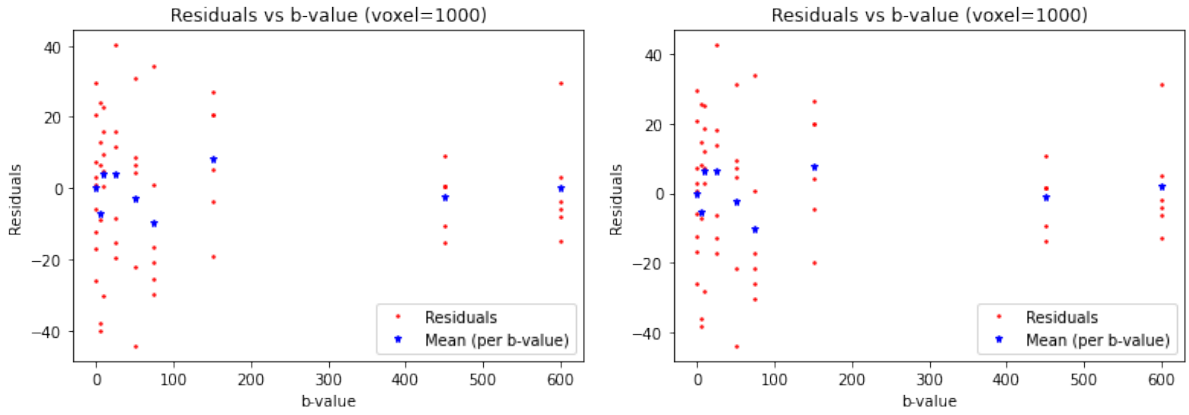
(a) LSQ fit

(b) IVIM-Net fit

Figure 9: True values and predicted IVIM curve for voxel nr. 1000 of patient 1 - using LSQ (a) and IVIM-Net (b) fitting

Residuals

If we consider the residuals and especially the mean residual values per b-value, which are visualised for one given voxel in Figure 10, we observe the same negligible differences between the two fitting algorithms as before. Specifically, the mean residual values of the first few b-values show a minimally deviating pattern.



(a) Residuals from the LSQ fit

(b) Residuals from the IVIM-Net fit.

Figure 10: Residuals resulting from the fit of voxel nr. 1000 of patient 1 - using LSQ (a) and IVIM-Net (b) fitting

An important note is that the training process of the IVIM-Net is computationally more

exhaustive compared to one of the LSQ fitting algorithm. Besides that, the differences are insignificant. Moreover, we are not specifically interested in absolute parameter or residual values but primarily in the way a measured signal intensity relates to the other signal intensities at that b-value, which is not expressed as an independent value but as a relationship or proportion. We observed during the experiments that the differences between the two algorithms vanished when applying the scaling methods and combining this with the computational demands of the algorithms, we decided to continue with the LSQ fitting algorithm. Therefore, from here on all the results in Sections 4.1 and 5.2 are obtained by using the LSQ fitting algorithm.

5.1.2 Scaling the residuals

As expected, by visually inspecting the sum of squared residuals for various slices and patients, we observed that the sum of squared residuals, as visualised for a given slice and patient in Figure 11, correlated with the measured signal intensities. This is due to the fact that if the signal in a certain tissue type is stronger and the percentage deviation remains indifferent, the resulting absolute residuals will be greater. Evidently, the sum of squared residuals will be too then. This correlation clouds our judgement on which measurement is an outlier an which is not, as it will direct us towards high-signal regions instead of highly deviating regions. Therefore, we explore different methods for scaling the residuals, which are detailed in the next paragraphs.

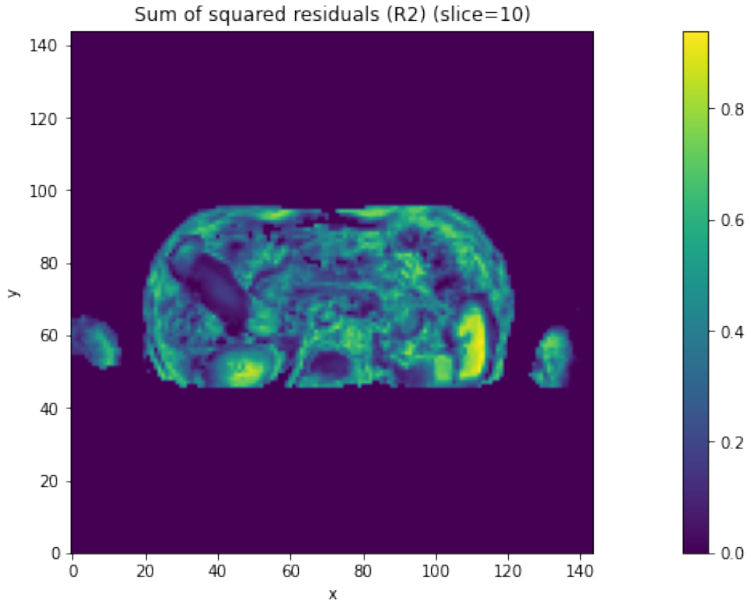


Figure 11: Sum of squared residuals over the measurements of slice 10 for patient 1.

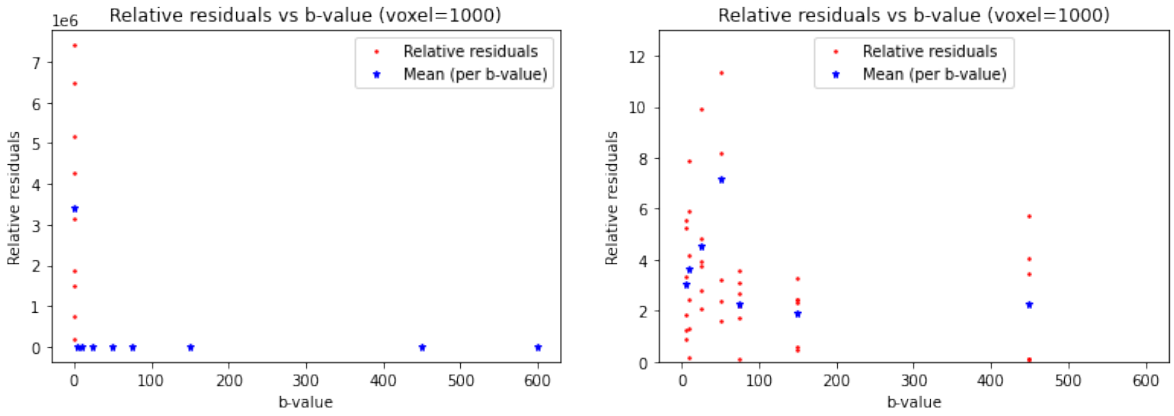
As a final remark, we observed that the magnitude of the b-value, and hence the signal intensity, is anti-correlated with the variance of the residuals. For instance, when looking at the plots in Figure 10 again, this can be noticed by the decreasing spread as the b-value increases. This can be explained by the fact if the signal weakens, and the percentage error, constituted by for instance background noise, remains stable, the absolute deviation will be smaller.

Relative residuals

The first scaling method that we discuss is the relative residuals. As hypothesized, this scaling method leads to exploding values at b-values where the mean residual value is close to 0. This is, for instance, the case for voxel number 1000 of patient 1 at $b = 0$, as can be observed in Figure 12a, where the highest relative residual value is around $7.5{*}10^6$. This makes sense, since the mean residual value in Figure 10b is approximately 0. This issue was observed multiple times when visually inspecting various voxels and patients.

If we consider the relative residual values for the other b-values in some more detail, which are visualised in Figure 12b, we can conclude that the method proves quite effective outside those problematic cases. The ranges of the residuals at different b-values

are already significantly more comparable. Originally, the residual values ranged from around -45 to 40, this range has been limited to approximately 0 to 12. Furthermore, the mean relative residual values lie within approximately 2 and 7 instead of the -10 and 10 from the original range. Moreover, the distribution of the magnitude of the spread has shifted, which is especially visible at $b = 75$ and $b = 150$.



(a) Relative residuals

(b) Relative residuals (with y-limit)

Figure 12: Relative residuals of voxel nr. 1000 of patient 1 - without y-limit (a) and with y-limit (b) for plotting

MinMax residuals

When considering the residuals scaled by using the MinMaxScaler, as visualised in Figure 13 for a given voxel and patient, the first thing that stands out are the boundaries at $y = 0$ and $y = 1$. These are the forced limits of the scaled residuals that we earlier mentioned as a potential limitation of this method. We see that for every b-value the smallest value is located at 0 and the largest at 1. This suggests that each scaled residual value of 1 at a certain b-value is equally deviating, which is not true, as we learned in the previous paragraphs. This issue recurred in nearly all the voxels and patients that we visually inspected. Therefore, this method is rather misleading and the results can be misinterpreted very easily.

Evidently, a strength of this method is the fixed range, and therefore the comparable magnitudes. However, the level of corruption of the original distributions and hence the misleading magnitudes outweigh the benefit of having comparable magnitudes.

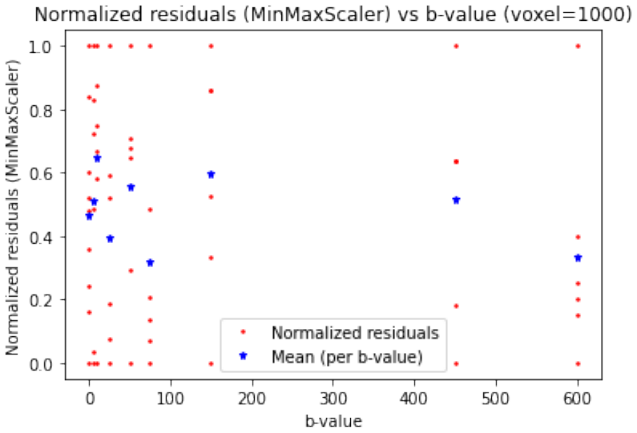


Figure 13: MinMax scaled residuals patient 1.

Absolute MinMax residuals

The absolute MinMax residuals are slightly less misleading, since a value close to 0 represents the smallest absolute residual value, which is actually close to 0, opposed to the regular MinMax residuals where a value close to 0 represents the smallest residual value, which is around -40 for some b-values. However, the insinuated equality in deviation between scaled residuals with an equal value remains unchanged and is therefore still a major limitation to this method.

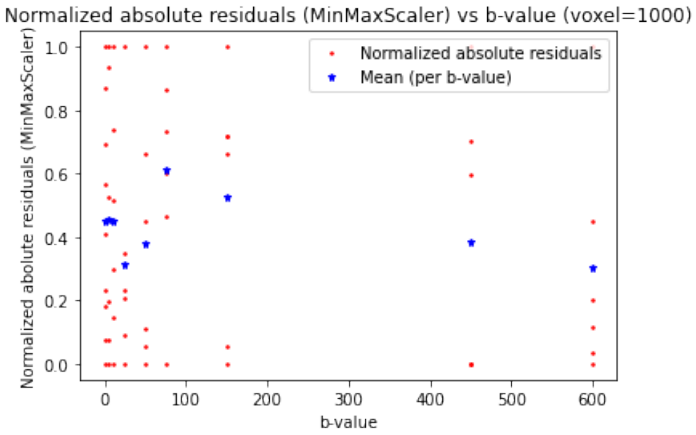


Figure 14: Absolute MinMax scaled residuals patient 1.

Absolute z-score

We explore two variants of the absolute z-score. The first is determined by normalizing

the residuals over all measurements for that voxel. Through analysing the resulting residuals, as visualised for voxel number 1000 in Figure 15, we observed that they are still correlated with the b-value. This can be observed by the recognizable pattern in spread over the b-values. This is not surprising as we normalize over all measurements, hence over all signal intensities, and do not account for the b-value at which a certain measurement is obtained.

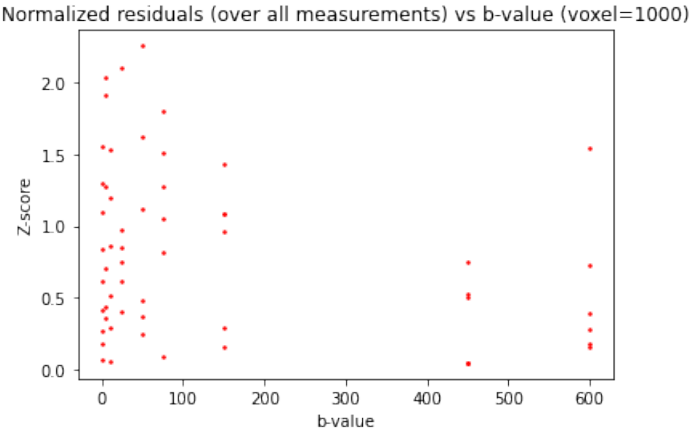


Figure 15: Z-score (normalized over all measurements) patient 1.

The second z-score variant is obtained by normalizing the residuals per b-value. Its outcomes for voxel number 1000 can be found in Figure 16. We observed by inspecting various voxels and patients that the previous correlation between the z-scores and the b-values has seemingly disappeared. This observation is underpinned by the fact that the mean z-scores per b-value are approximately aligned. Furthermore, the spread is quite evenly distributed compared to before. Therefore, this scaling method seems to be a suitable approach for our research.

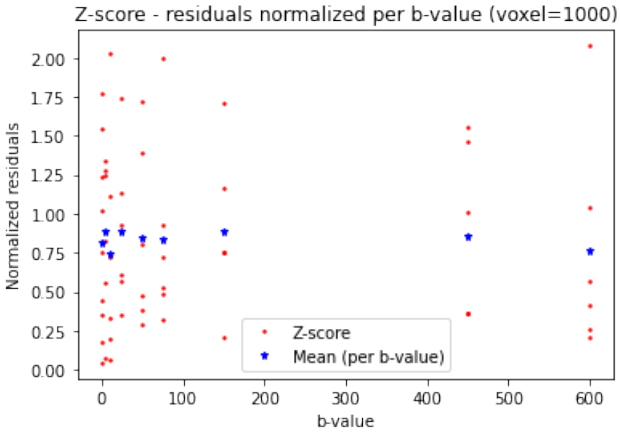


Figure 16: Z-score (normalized per b-value) patient 1.

By deploying latter scaling method, we are able to successfully eliminate the dependency on the b-value, which allows us to make a fair comparison between the residuals of different b-groups, which was the objective of scaling the residuals.

A downside of using the z-score is that it limits the number of 'bad measurements' that we filter. It functions as an indication on how deviating a given measurement is from the other measurements at that b-value, but that does not tell how poor the measurements at that b-value actually are. If the standard deviation of that b-group is extremely high and the spread of the measured signal intensities is therefore very large, the z-score will not necessarily capture this and the majority of these measurements will have a z-score below one. Namely, a high z-score is only obtained when a certain measurements deviates much more than the others within that group. Therefore, a z-score will not always tell how good or poor a measurement is, it just indicates how it compares to the others within that b-group.

5.2 Investigating the outlier-artifact correlation

Since we aim to find slice-timepoint pairs that are corrupted by artifacts, we consider the mean z-score per slice as a quality measure instead of a z-score per voxel. The obtained values for patient 1 are visualised in Figure 17. Note that these measurements are not chronologically sorted, but on ascending b-value. On the x-axis, we see the b-value for that certain measurement and which repeated measurement it is. On the

y-axis, we see the slices. Hence, each square equals one unique measurement for a certain slice.

The brighter and yellower the square is, the higher the mean z-score, which ranges from approximately 0.6 to 1.3. A column containing multiple bright squares therefore indicates that that measurement is poor compared to the other measurements at that b-value.

Patient 1

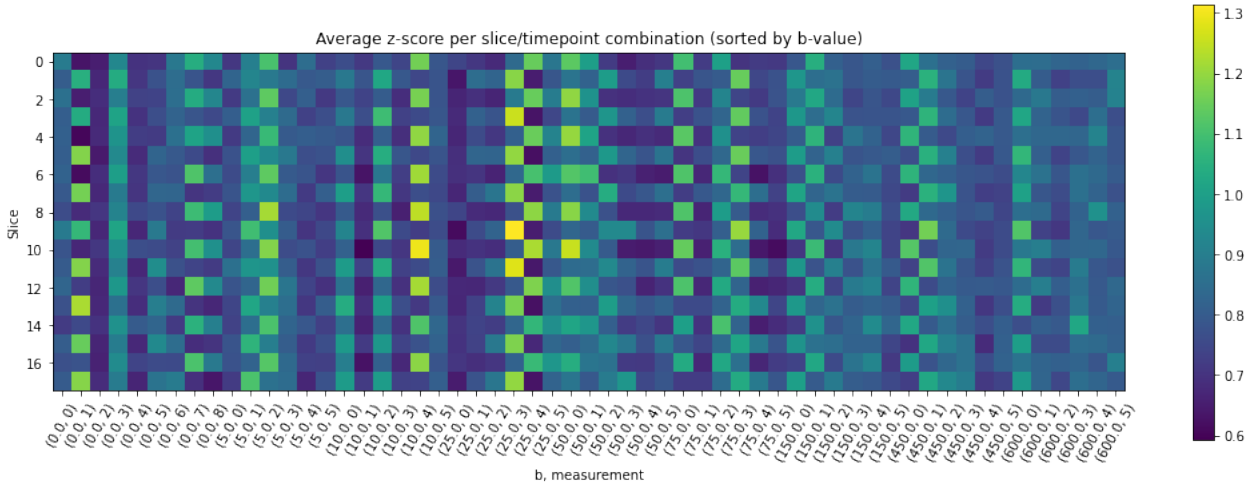
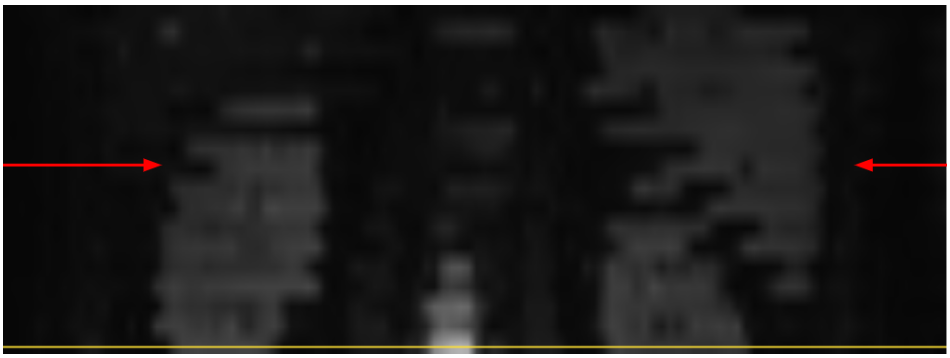


Figure 17: Mean z-score per slice/timepoint pair for patient 1.

In this figure we can distinguish an interleaved pattern. This pattern arises because of the acquisition protocol. Typically, during such protocol odd-numbered slices are for instance scanned firstly and even-numbered slices are scanned afterwards. If there is patient motion (e.g. respiratory motion) in the two phases of scanning, the slices will be dislocated when stacking them alternately. This dislocation results in jagged edges in the coronal orientation, as visualised in Figure 18a, and an interleaved pattern as present in the figure.



(a) An interleaved acquisition pattern from patient 1.

(b) A regular acquisition pattern from patient 1.

Figure 18: Two examples of an interleaved (a) and regular (b) acquisition pattern from patient 1.

If we filter the pairs with a mean z-score higher than 1 - our pre-determined threshold, we obtain the pattern displayed in Figure 19, which represents the pairs that are deemed outliers by the model. This leaves us with the question whether the outlier pairs lead us towards the presence of artifacts in that slice. This question can be answered by performing a manual inspection on the slice-timepoint pairs, of which the results are discussed in Section 5.2.2.

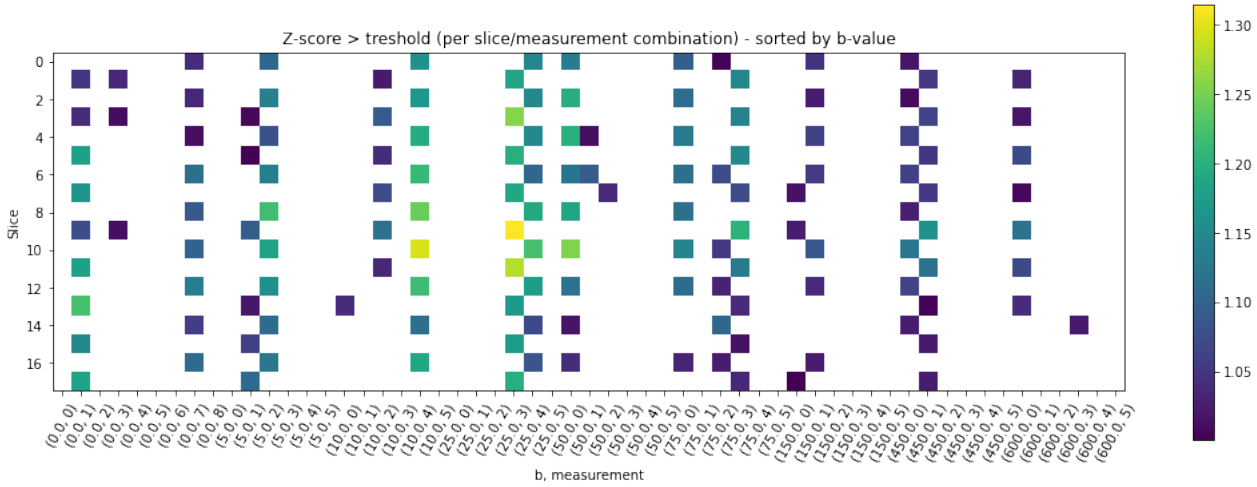


Figure 19: Slice/timepoint pairs with mean z-score > threshold for patient 1.

Patient 2

If we consider the mean z-scores obtained for patient 2, which can be found in Figure 20, we can again recognize the interleaved pattern. However, it is less visible than for patient 1. Furthermore, although most pairs have a slightly lower score than the ones of patient 1, the range is enlarged and ranges to approximately 1.9. On a side note, the first column is the only column that contains scores above 1.6.

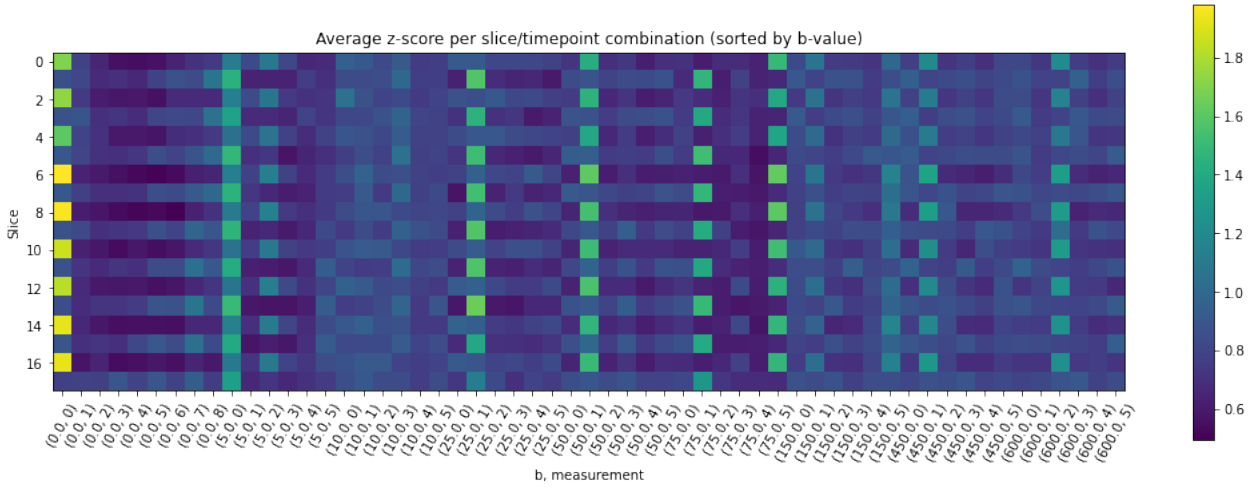


Figure 20: Mean z-score per slice/timepoint pair for patient 2.

If we filter the outlier pairs, we obtain the figure shown in Figure 21. Considering these

measurements in chronological order, as shown in Figure 22, we see that the interleaved pattern disappears in the last 22 measurements. A reason for this could be that the patient adhered to the instructions more closely than during the preceding series of measurements. Such instructions could for instance be breath-holding instead of free breathing.

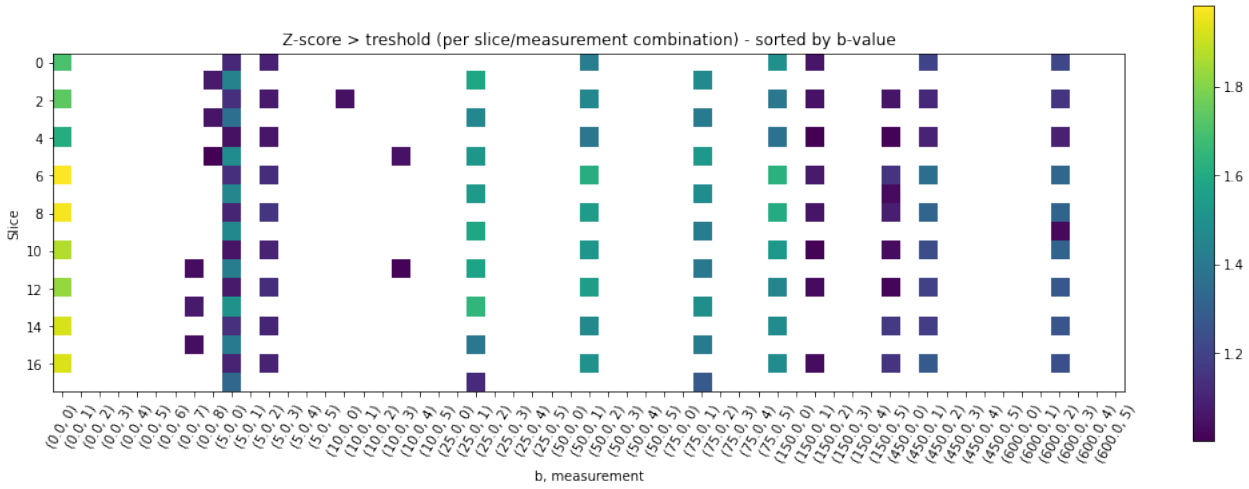


Figure 21: Slice/timepoint pairs with mean z-score > threshold for patient 2.

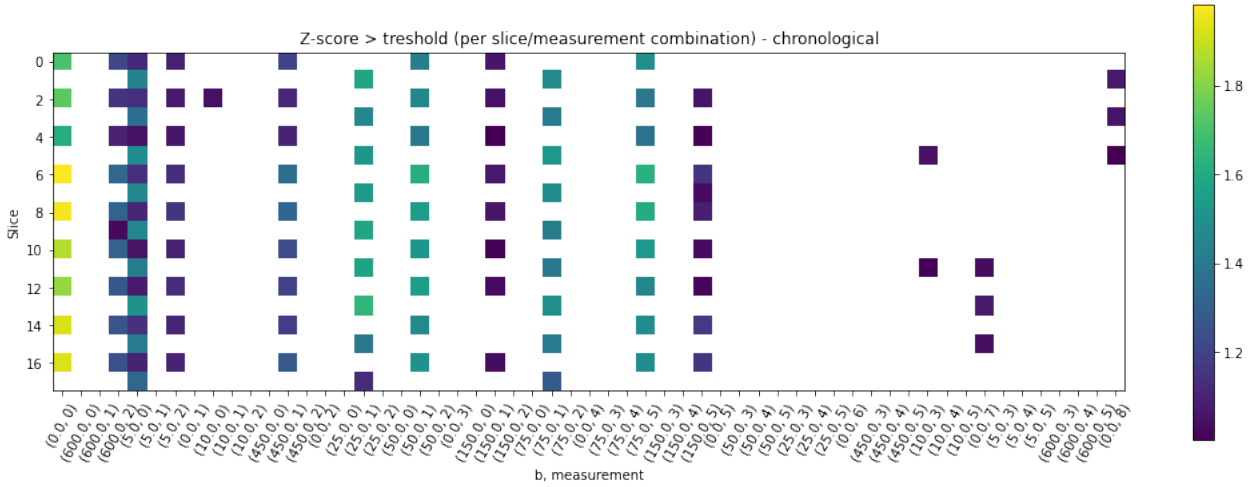


Figure 22: Slice/timepoint pairs with mean z-score > threshold (in chronological order) for patient 2.

Patient 3

For patient 3, for which the mean z-scores are visualised in Figure 23, the range is

slightly tighter. The highest score is about 1.2, which is just above the threshold. Furthermore, we again observe an interleaved pattern.

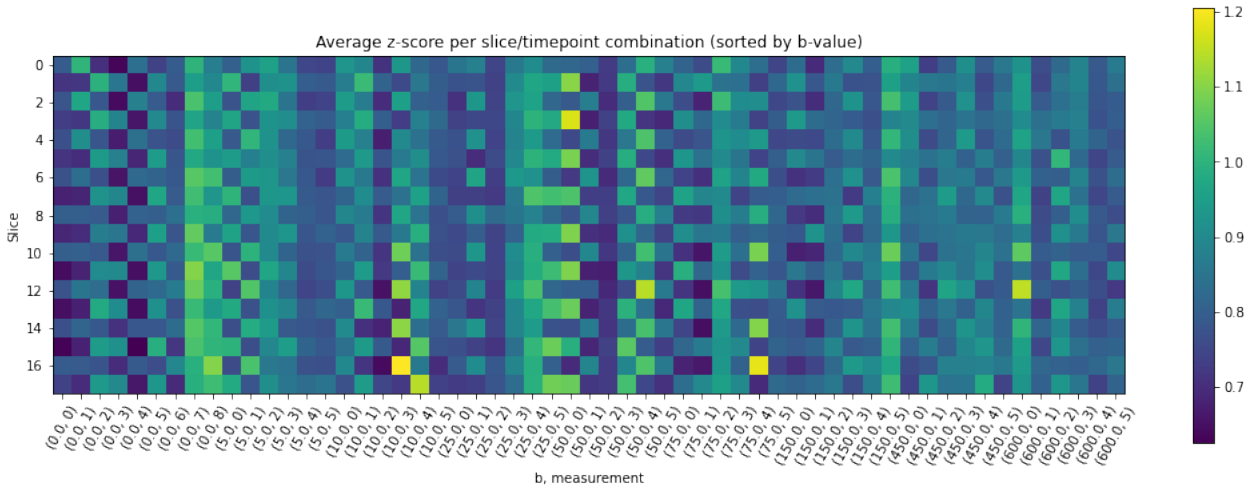


Figure 23: Mean z-score per slice/timepoint pair for patient 3.

If we consider the outliers, as shown in Figure 24, the interleaved pattern slightly disappears as the outliers show quite some randomness.

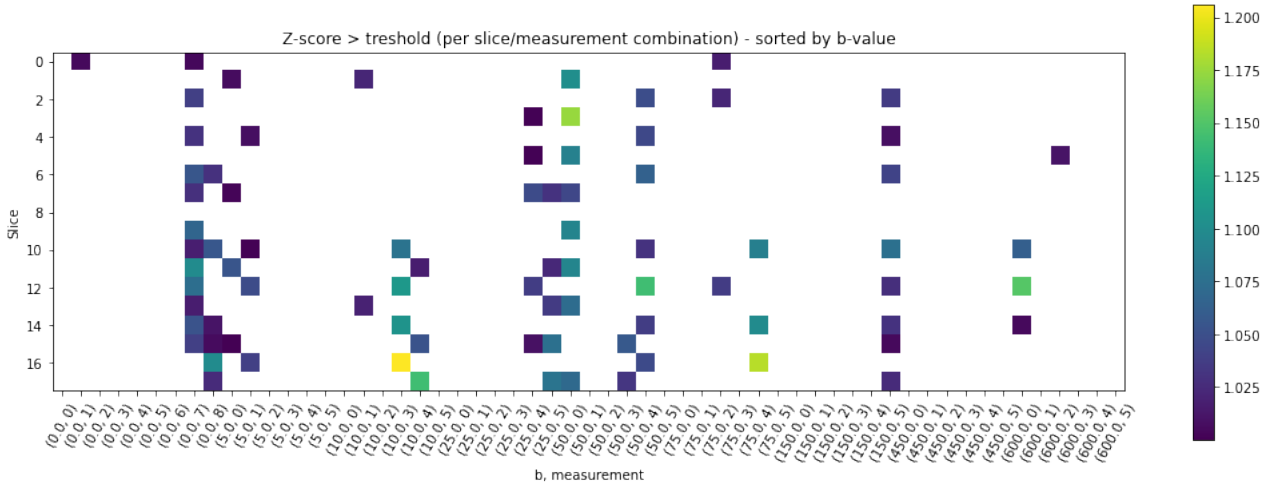


Figure 24: Slice/timepoint pairs with mean z-score > threshold for patient 3.

Patient 4

Patient 4 has an even tighter range of mean z-scores, as can be seen in Figure 25. There

are very few scores greater than the threshold, as is underpinned by Figure 26 and the fact that the highest score is approximately 1.15. If we compare these outliers to the ones from patient 1, we see both a different number of outliers and a different pattern in which they appear. Furthermore, except for the second to last column that contains outliers, all other columns do not contain outliers across the full range of slices, which is the case for, for instance, interleaved motion artifacts. However, the columns mostly contain singular outlier slices, which hints at a different type of artifacts.

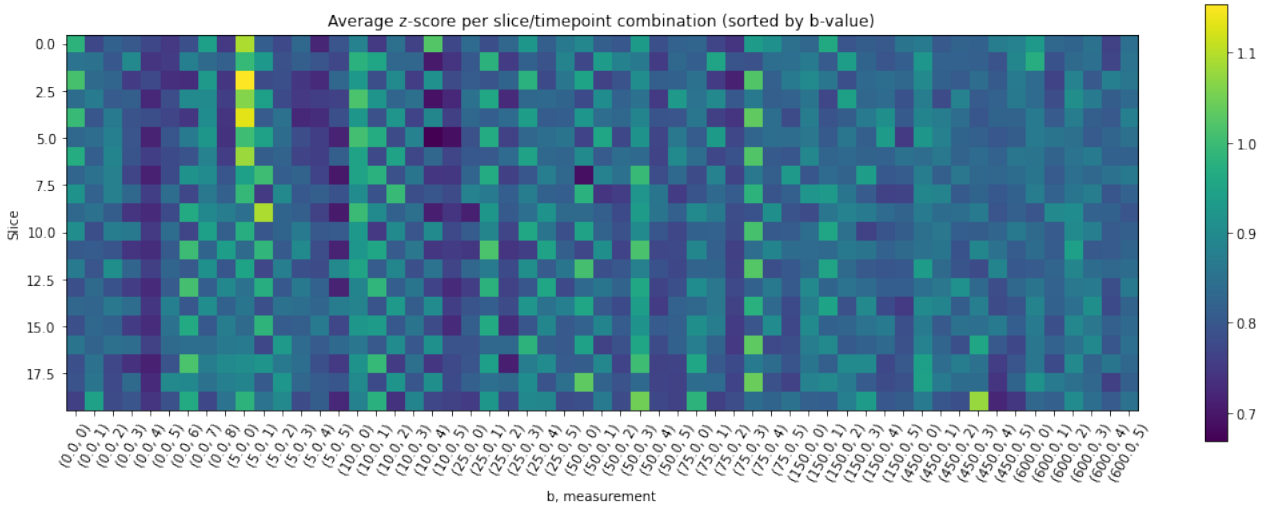


Figure 25: Mean z-score per slice/timepoint pair for patient 4.

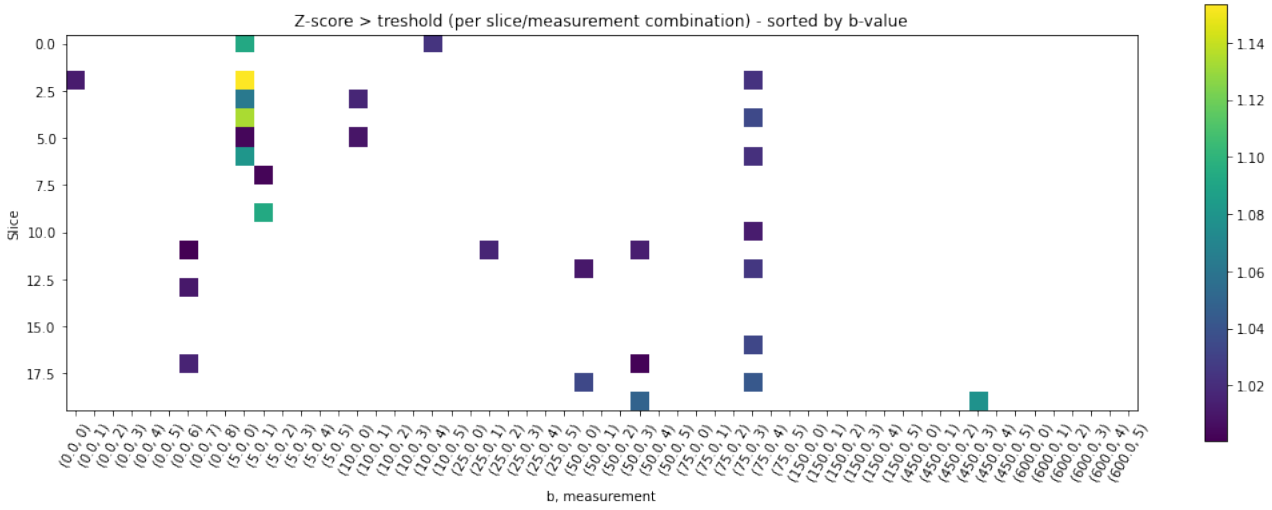


Figure 26: Slice/timepoint pairs with mean z-score > threshold for patient 4.

5.2.1 Local versus global

In order to gain insights in which types of artifact are detected in an outlier slice, we explore how locally or globally the artifact is caused. To assess this, we deploy the method described in Section 4.1.3. While doing so, we firstly visualise the results obtained by computing the sum of voxels with a z-score greater than our threshold per slice and dividing it by the mean z-score of that slice. We refer to this as the normalized sum. This sum is determined for each slice-timepoint pair that was labeled as outlier.

Patient 1

In Figure 27, the resulting normalized sum per outlier pair is visualised for patient 1. If we analyse the values in this figure, we suspect a correlation between the number of valid voxels in a certain slice and the normalized sum obtained, because of the trend of lower values in the top part of the figure compared to the higher values for the slices in the middle. Furthermore, by taking the sum of voxels without accounting for the number of valid voxels in the slice, we obtain absolute values of which we are not able to tell how these values relate to each other.

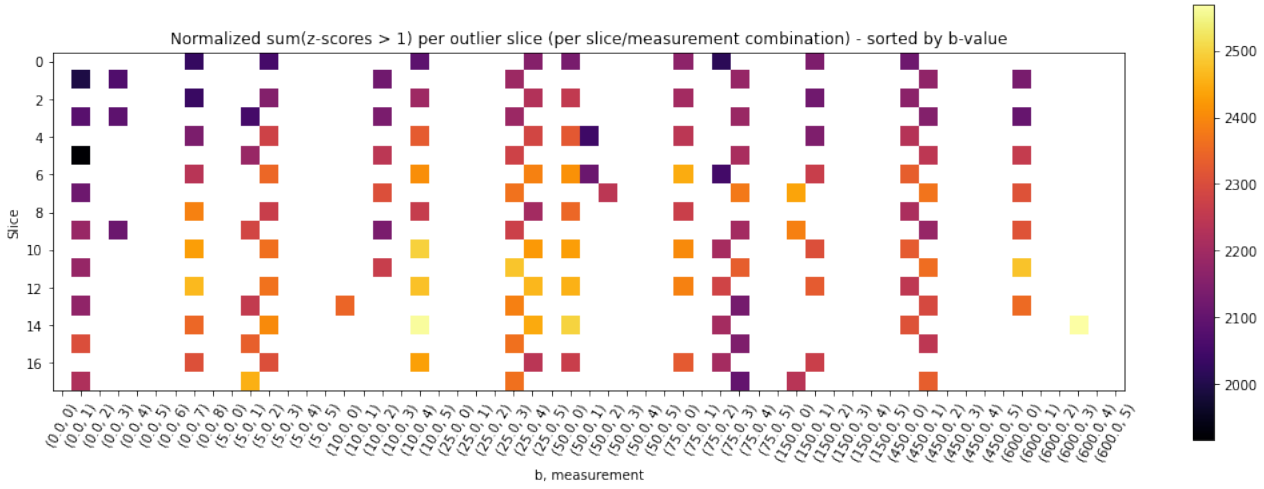


Figure 27: Normalized sum of voxels with a z-score > threshold per outlier slice for patient 1.

Therefore, we decided to divide this value by the number of valid voxels per slice, which results in a ratio of outlier voxels, as visualised in Figure 28. In theory, this value now ranges between 0 and 1. However, for patient 1 the values range from approximately

0.41 to just above 0.52. The trend that we noticed before has disappeared, which means that the resulting values now show us the proportion of the slice that causes the slice to be an outlier.

The far right column contains one of the highest ratios, which means that this outlier is caused more globally than for instance the one for slice 5 in the first column. Moreover, apart from some fluctuations, the values remain comparable and besides that, there is no clear trend in the rest of the columns. Although, in Figure 19 we observed lower values for the columns in the right part of the figure compared to the ones in the middle. Hence, we can state that the level of distortion decreases over these columns, but the ratio of outlier voxels remains roughly the same. During the manual inspection, of which the results are discussed in Section 5.2.2, we will explore the images corresponding to some of these outliers in order to investigate the differences.

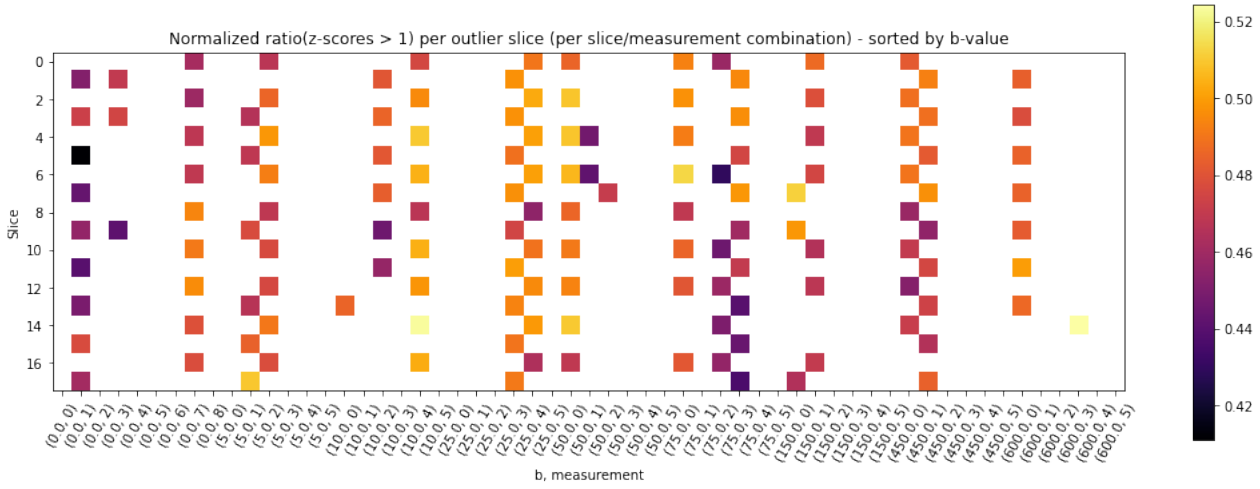


Figure 28: Ratio of voxels with a z-score > threshold per outlier slice for patient 1.

Patient 2

Considering patient 2, of which the ratios are visualised in Figure 29, we see that the range is wider than for patient 1, ranging from around 0.39 to 0.60. Furthermore, we observe significantly lower values in the first column than in the rest of the columns. However, in Figure 21 we observed the highest values in the first column, which suggest that the lowest ratio of voxels causes most distortion, with mean z-scores up to almost 2. Hence, the outliers at this timepoint are caused rather locally by highly deviating

regions. Furthermore, we observe similar behaviour to patient 1 in terms of a decreasing mean z-score as we move to the right of the figure coupled with a ratio that remains stable.

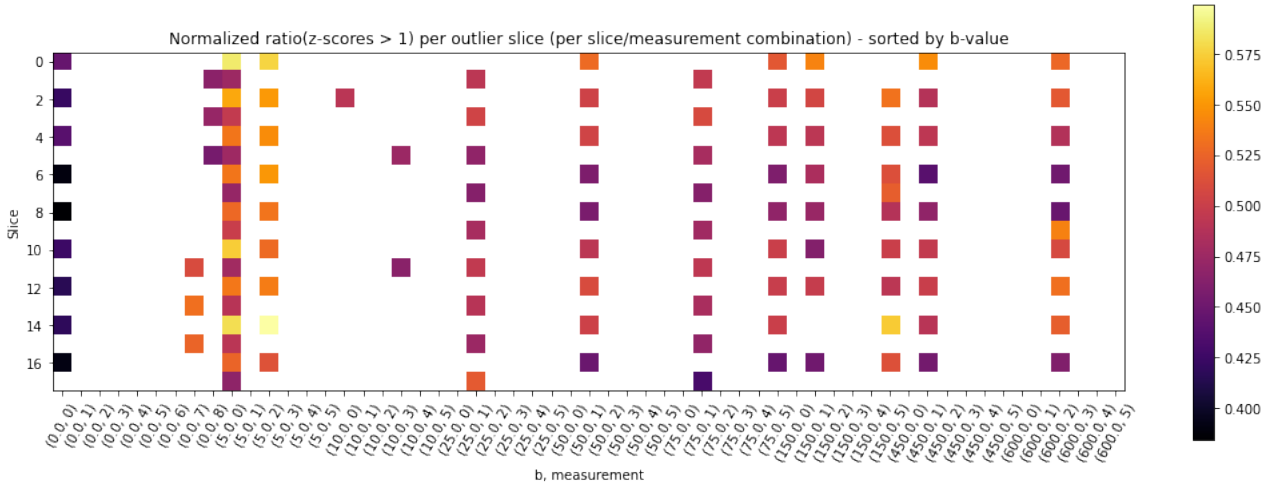


Figure 29: Ratio of voxels with a z-score > threshold per outlier slice for patient 2.

Patient 3

If we look at the ratios for patient 3, as visualised in Figure 30, we observe more randomness and therefore, fewer trends. The only clear pattern that we can distinguish is the center column (50.0, 0) which contains bright colors, and therefore high values, for almost every odd-numbered slice. This suggests an interleaved motion artifact. Furthermore, the values in this figure range from around 0.41 to almost 0.54, which is similar to patient 1.

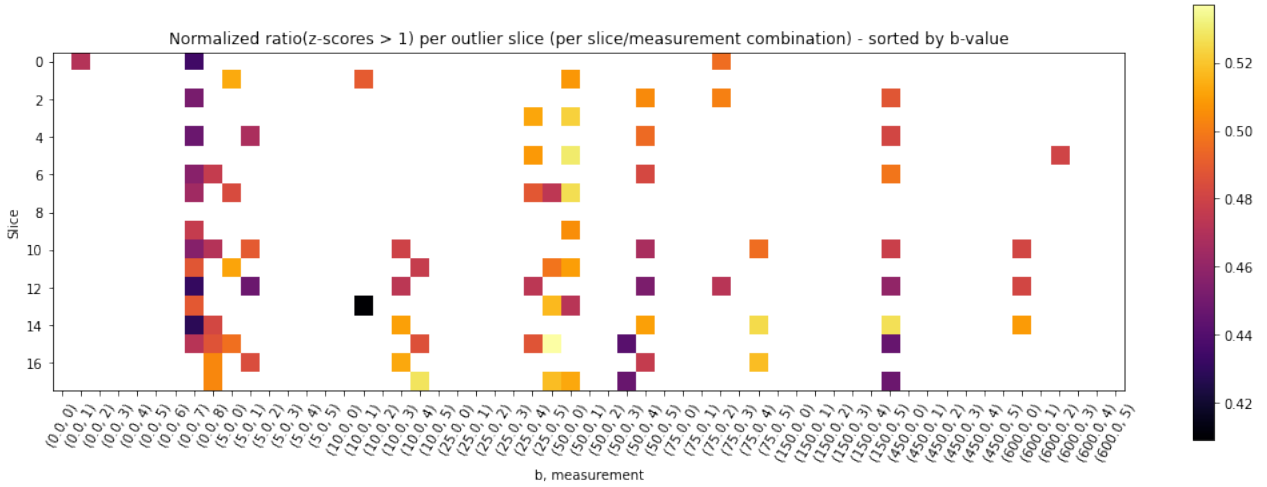


Figure 30: Ratio of voxels with a z-score > threshold per outlier slice for patient 3.

Patient 4

For patient 4, the ratios are visualised in Figure 31. If we compare these results to the ones in Figure 25, we see that the highest values in both figures are centered around the columns of the first and second measurement for $b = 5$, i.e. (5.0, 0) and (5.0, 1), and the single outlier at $b = 450$. Furthermore, we again observe an increasing ratio as the b-value increases paired with a stable mean z-score, which means that the outlier becomes more global. Lastly, the range is narrow, ranging from approximately 0.455 to 0.52.

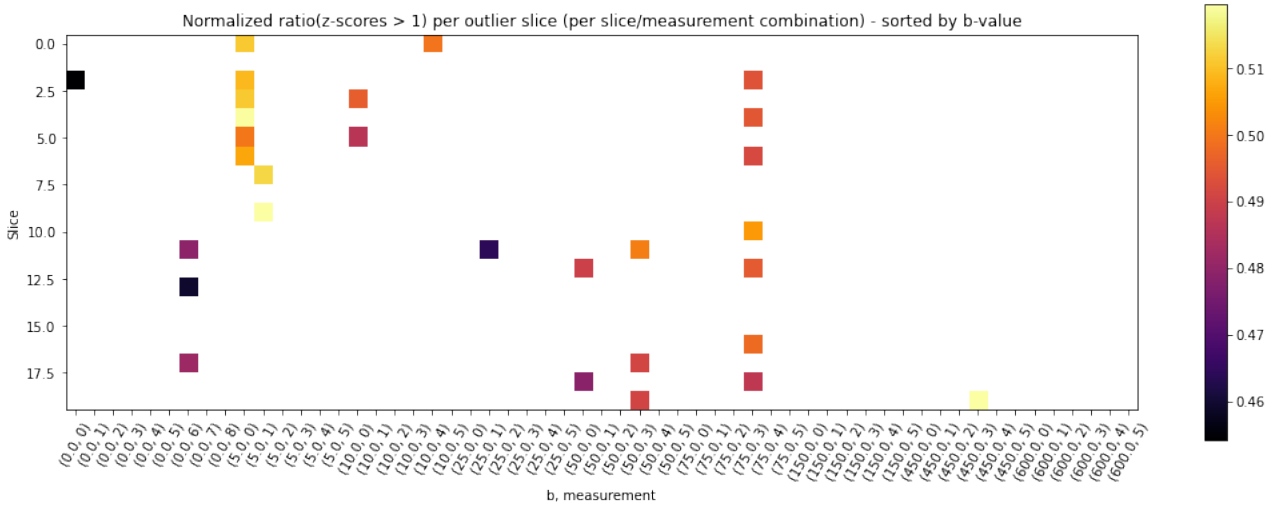


Figure 31: Ratio of voxels with a z-score > threshold per outlier slice for patient 4.

5.2.2 Manual inspection

In order to evaluate the performance of this first model, we perform a manual inspection. The procedure we follow is described in Section 4.1.4. The first part of this inspection focuses on assessing the model performance in terms of how well it detects outliers and happens through visual inspection of the measurements. In the second part, we consider a selection of measurements that either have a very high or low mean z-score or are very globally or locally caused. This part provides us with insights into which types of artifact are detected by the model.

Evaluation

For this part, the results can be found in Appendix A. The artifact labels that we use are the ones detailed in Table 3. Furthermore, Table 6 shows the labels for patient 1 resulting from the visual inspection for the 28 slice-timepoint pairs with the highest mean z-scores. Conversely, Table 7 shows the results for the 34 pairs with a mean z-score of just above the threshold, which makes them minimal outliers. Furthermore, Table 8 shows for all timepoints whether they contain at least one outlier slice according to the visual inspection (second column) and the model (third column).

By analysing the first two tables, we can conclude that there is a high correlation between the outliers as labeled by the model and the artifacts as labeled during the inspection, since each outlier, also the minimal ones, contains at least one artifact. If we look at the third table, the first thing we notice is that the patterns look quite similar. If we observe them in a bit more detail, we do see some differences, however. Especially the labels for timepoints 41 until 47 differ. In summary, 4 of the 26 'poor' timepoints are not labeled as poor (false negatives) and 1 of the 23 timepoints is falsely labeled as poor (false positives). Overall, around 85% of the poor timepoints is detected and less than 5% has a false 'poor' label.

Local vs global

In this part, we start by visually inspecting three measurements of patient 1 with an equal b-value, of which the third has a high mean z-score and was therefore labeled as outlier. These measurements are visualised in Figure 32, in which the left plots showcase the z-score per voxel for slice 9 at b-value 25. The right images show the measured signal intensities during these measurements. Since the measurements have an equal

b-value, we would expect them to be very similar, which is not the case per se. We can see from the bottom left figure, which is the outlier measurement, that the outlier is caused rather globally, since the yellow areas (high values) are visible over the whole slice. In the upper figure, we see a local deviation in the center of the bottom part of the plot. However, because the rest of the plot has rather low z-scores, the resulting mean z-score of the slice was not high enough to be labeled as outlier.

If we look at the scans on the right side of the figure, we can clearly recognize the deviating regions indicated by the z-score plots on the left. For the upper scan, we clearly distinguish the brighter region in the center of the bottom part. In the bottom scan, we are able to distinguish the patterns too. However, in this case we recognize both signal increase as signal dropout. The bright yellow region just left from the middle of the plot can be recognized in the scan by the brighter region. The bright yellow half circle below, however, can only be noticed in the scan by comparing the bright region to the circles in the other two scans, which are bigger. Hence, this is a decrease in signal. The same holds for the center part of the scan, which is darker for the bottom scan than for the other two. Lastly, the two separate bright yellow regions on either side of the abdomen, which are the arms, are not clearly visible in the scans on the right. However, if we look close enough, we can see that in the bottom scan the arms are even darker than in the two upper scans, which explains the higher z-scores.

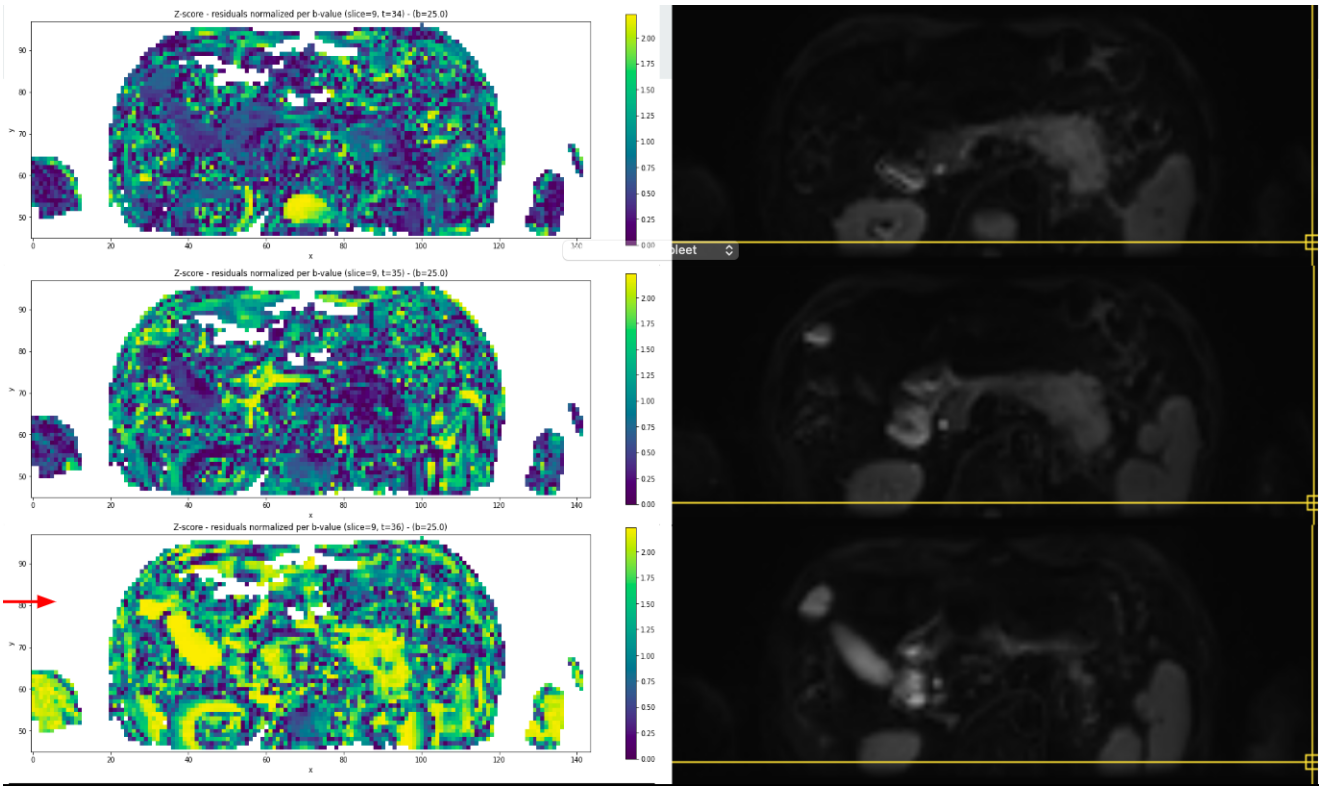


Figure 32: A global outlier example from patient 1 for b=25.

Next, we explore three measurements for slice 9 of patient 1 at a b-value of 50, which are visualised in Figure 33. Evidently, we observe a locally deviating region in the middle plot of the left side of the figure, showcasing a z-score of more than 2. However, since this deviation is very local and multiple parts of the slice contain values below 1, the resulting mean z-score is just below 1 - our threshold. Therefore, this measurement was not labeled as outlier, nor were the other two measurements.

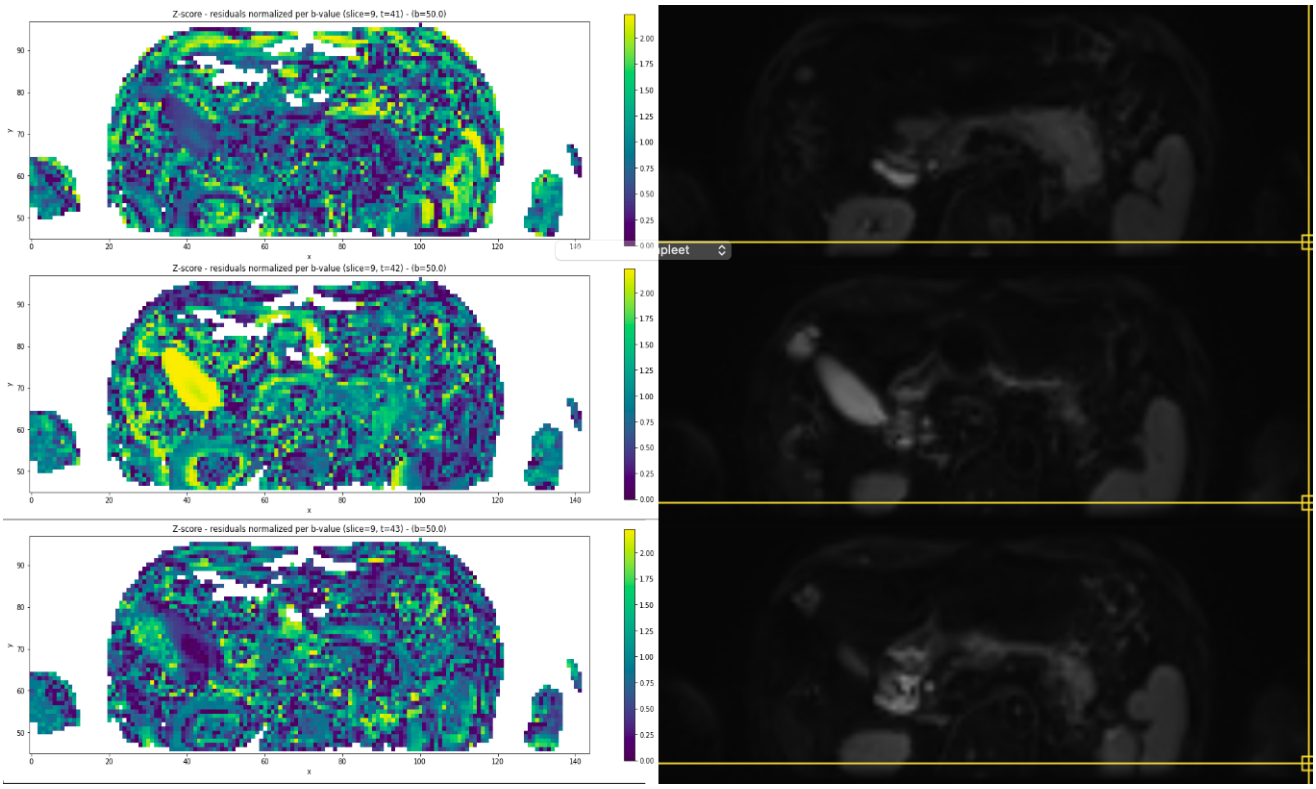
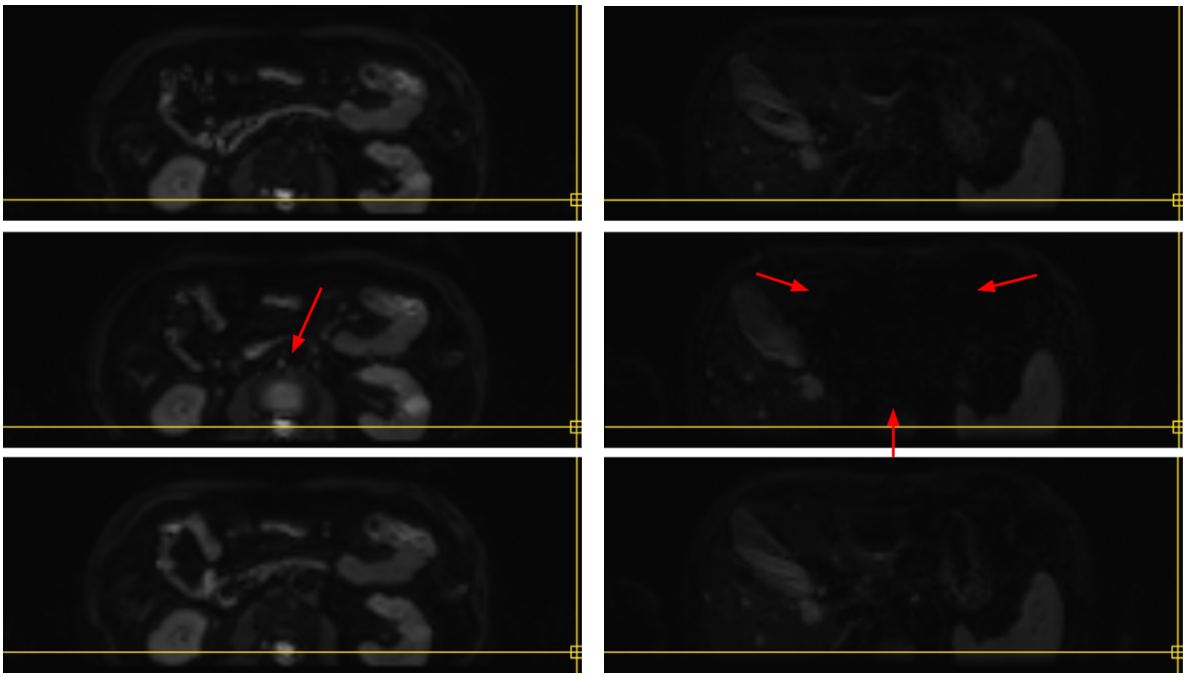


Figure 33: An example of a local high z-score region from patient 1 for b=50 (no outlier slice.

In Figure 34, we see a local and a global artifact from patient 1. Figure 28 showed the highest ratio for the global artifact visualised in Figure 34b and a rather low ratio for the local artifact previewed in Figure 34a. The scans on the left are scanned at a b-value of 75, the ones on the right at a value of 600. The artifacts occur in the middle row of the figure and are indicated by the red arrows. For the local artifact we observe a locally increased signal intensity in the bottom part of the scan. The global artifact can be recognized by the large dark region in the center of the scan, caused by signal dropout.



(a) A local artifact from patient 1.

(b) A global artifact from patient 1.

Figure 34: Two examples of a local (a) and global (b) artifact from patient 1.

Summarising the results obtained by performing the visual inspection, we can conclude that there is a high correlation between the mean z-scores and the artifacts. Moreover, the interleaved motion artifacts are detected very well. However, the model is able to capture local artifacts too. Since the aim of this model is to function as proof-of-concept and provide pseudo-labels for the second part of the research, these results are sufficient for now to advance to the second part of the research.

5.3 Deep learning to the rescue?

In order to develop a more advanced and more efficient alternative, which does not necessarily rely on completeness of the patient's data, we try to answer the question whether deep learning can provide a reliable solution to automated outlier detection in 4D IVIM MRI. Therefore, in Section 5.3.1 we start by analysing the performances of the base models described in Section 4.2.3. The training and validation performance curves of these models can be found in Appendix B. Subsequently, in Section 5.3.2 we analyse the performances of the models detailed in Section 4.2.4. In Appendix C, the

performance curves and outputs of the CNN's are detailed.

The base models are trained according to the procedure described in Section 4.2.3. For the CNNs, the training process is detailed in Section . All models are trained on a GPU.

5.3.1 Base models

Firstly, we consider the performances and outputs of the base models in order to determine a baseline performance. The baseline model is the Base classifier, which is highlighted in the next section. Hereafter, we analyse two variants on this model, the Two-class model and the Grouped b model. Lastly, we consider the Base regressor.

We expect the base models to not be efficient enough, yet complex enough to yield both a good training and validation performance. However, they provide us with insights in a baseline performance and therefore, allow us to assess the differences in performance when adjusting the model architecture.

Base classifier

The results of the Base classifier are visualised in Appendix B.1. If we consider the training and validation loss, visualised in Figure 49, we notice that the validation loss starts increasing after around only 10 epochs. This means that the model starts overfitting from that point and therefore, it becomes worse. But, if we take a look at the performance metrics visualised in Figures 50 until 54, we can conclude that what the model has learned during the first 10 epochs is negligible. It is likely that the model is at that point still in its local minimum where all predicted labels are 0.

If we continue the training process, and thus continue overfitting, we do see that for instance the validation sensitivity and F1-score keep improving. Therefore, the model improves at predicting a positive label - outlier in our case, but the labels are not necessarily correct. So, it predicts more positive labels, yet it also becomes less certain in predicting the labels. Furthermore, the overfit results in a large discrepancy between the training and validation performance, as exemplified by the sensitivity curves, which show a similar pattern but approach values of, respectively, 1 and 0.3.

Lastly, we inspect the predicted values versus the true labels for the four validation patients, as visualised in Figure 56, where the blue dots represent the true outlier labels and the red dots are the values predicted by the model. Here, we observe that for the first two patients the model's predicted values are in general too low. The third patient's values look decent. The ones of the fourth patient do so too, however, there are many values in the center part of the plot. The more values there are in this region, the more values are close around 0.5. Predictions around 0.5 are the most uncertain ones, since a minor change in value could result in an opposite class label. Therefore, the predictions for validation patient 4 look rather uncertain.

Two-class model

For the Two-class model, the results are detailed in Appendix B.2. By analysing the performance curves, and especially the loss curves, we can conclude that this model exhibits similar behaviour to the base classifier. However, for this model the overfitting starts after around 40 epochs. Furthermore, the training loss is much higher compared to the previous model. The validation loss is initially much higher too, but converges to a comparable value. It is important to note that the loss for the two models is based on a different loss function, namely BCE and weighted BCE respectively. This hinders direct comparison between the loss of the models.

Two other performance metrics that stand out are the extremely low values in the validation F1-score plot (Figure 61b) and validation sensitivity plot (Figure 61b). If we compare them with their training counterparts, this is accentuated even more. Although both metrics keep improving over the epochs, the performance on these metrics is very poor. Furthermore, the validation precision curve, shown in Figure 62b, is interesting since it fluctuates quite a bit and does not improve overall.

If we compare the predicted values with the true values, as visualised in Figure 64a, we observe that the model performs poorly on all validation patients. For patient 3, it stands out that the model solely predicts values close to 0. Although this patient seems to have very few positive labels in its true data, this still is undesired behaviour. The predicted values for the first and fourth patient are in general too low. Moreover, the predictions for patients 1, 2 and 4 seem very uncertain, since many of them are in the center part of the plot.

Grouped b model

Regarding the results of the Grouped b model, as detailed in Appendix B.3, we can conclude that the model again suffers from overfitting. By observing the loss, as visualised in Figure 65, we see that the overfitting starts after approximately 70 epochs, which is seven times slower than the base classifier. Furthermore, the overfit is smaller, as can be seen by the final value of the validation loss which is around 0.38, versus 0.62 from the base model. Moreover, their minimum is almost equal, but is reached during a different epoch.

In this case, the validation F1-score does not keep improving while overfitting, nor does the precision, which both show a very similar pattern. As for the first two models, the validation F1-score for the grouped b model is again very poor. The validation sensitivity is as low as for the two-class model, and therefore much lower than for the base classifier. Furthermore, the validation precision is much worse than for both previous models.

In line with the validation curves that were worse than the ones from the previous models, the predicted values versus true values shown in Figure 72 exhibit a poor performance. The predicted values for the first two patients are (almost) all 0 or close to 0. For the third and fourth patient, the model predicts values over the whole range from 0 to 1, which is good. However, by looking at the distribution of these values, we can conclude that the model is very uncertain.

Base regressor

Because these first three models show a strong overfit and yield a rather poor performance, we wanted to test whether using a regressor to predict a z-score per voxel instead of classifying a whole slice at once would enhance performance through aiding the model by providing an intermediate step. However, after implementing the base regressor, we noticed that training the model takes approximately one hour per iteration, which makes the model infeasible for clinical adoption in real-time. Therefore, we decided to not continue with this model.

5.3.2 CNNs

In order to facilitate multi-dimensional inputs instead of the flattened voxel tensors used in the base models and thereby enhancing efficiency, we test the three CNN's described in Section 4.2.4. These models aim to approximate the mean z-scores as closely as possible by minimizing the MSE loss. The classification metrics are determined by subjecting the predicted values to our threshold and subsequently computing the metric in question. Firstly, we analyse the results of the three separate models. Then, we compare the performances of the three models. For each model, we consider the performances both with and without L2-regularization.

CNN - channel per slice

First, we consider the CNN_{slices} model, of which the performance curves and outputs can be found in Appendix C.1. If we analyse the training and validation loss for the model without (Figure 73) and with (Figure 74) L2-regularization, we observe an increased training loss for the regularized model. The validation loss, however, yields an equal score, although the shapes of the curves are different. The regularized model has a more unstable validation curve that seems to start increasing in the last 50 epochs. The model without regularization seems to stabilize after 250 epochs. Furthermore, the training and validation losses converge to very similar values, which implies that the level of overfitting has clearly decreased compared to the base models.

Moreover, the other metrics exhibit similar behaviour. If we consider for instance the accuracy, as shown in Figure 75 and 76, we observe a decreased performance for the model with regularization. Furthermore, as the training process progresses, the model's predictions become slightly more certain. As expected, the regularized model is slightly more uncertain during training. All four curves yield a high accuracy, but the instability of the curves is undesirable.

The differences between training and validation performance become clear when looking at the F1-score, sensitivity and precision, which are visualised in Figures 77, 78 and 81 until 84. In these figures, we observe high discrepancies between the training and validation performance for the model without regularization. For the regularized model, these differences are much smaller, which means that the regularization is effective for preventing overfitting.

Overall, the L2-regularization does not seem to enhance the performance of the CNN_{slices} model in terms of boosting the performance metrics. However, it does reduce the overfit, which is useful too.

If we inspect the validation outputs, as visualized in Figure 87 until 96, this claim is confirmed. When observing the true versus predicted plots shown in the first two figures, we see that the spread in these plots is smaller for the regularized model. However, ideally we want the datapoints to lie on the diagonal line starting in the origin and ending at the top right part of the graph. This is not the case, since the points in the plots are located on a horizontal line, if we distinguish a line at all. This indicates that the range of predicted values is much tighter than the range of true values. Therefore, the more extreme a true value is, the higher the deviation is between the true and predicted value.

Next, we consider the predicted, true and difference in mean z-scores per slice-timepoint pair. We cannot distinguish clear similarities between the patterns in the predicted and true plots per patient, for both the model with and without regularization. If we compare the two variants to each other, we do notice a difference in predicted patterns. The regularized model seems to predict similar patterns for validation patients 1, 3 and 4, although their true values do not match. Therefore, we suspect that the regularized model suffers from too much regularization, which causes the model to make similar predictions for different inputs. Either way, the regularized model does not adapt sufficiently to new data.

CNN - channel per measurement

The second model to be discussed is the CNN_{meas} model, of which the results are highlighted in Appendix C.2. We start by considering the loss curves, which are shown in Figure 97 and 98. When analysing these curves, we again observe a decrease in training performance for the regularized model. Furthermore, although both validation losses exhibit similar values, their trends differ. The model without regularization has a decreasing loss curve, while the regularized model leads to an increasing validation loss. On the other hand, the difference in values between training and validation for the model without regularization is much larger than for the model with regularization, which suggests that adding regularization to the model will lead to a smaller overfit.

If we look at the accuracy, as visualised in Figure 99 and 100, we observe similar behaviour. The performance on the training accuracy is worse for the regularized model. Also, the validation accuracy increases over the epochs for the model without regularization and decreases for the regularized model. However, the training and validation values are more comparable for the model with than without regularization.

For the F1-scores, visualised in Figure 101 and 102, it is clear that the regularization worsens the model's uncertainty. This can be seen by the increase in variance. Furthermore, the training scores are much lower for the regularized model. However, the overfit is smaller and the F1-score is increasing over the epochs, as opposed to the F1-score of the model without regularization.

When considering the weighted F1-score (Figure 103 and 104) and the sensitivity (Figure 105 and 106), we notice that the regularized model becomes more uncertain as the epochs increase, while the model without regularization becomes more certain, which can be seen by the evolving variance. However, for the sensitivity the regularized model does show a more promising trend compared to the model without regularization.

Moreover, the precision, as detailed in Figure 107 and 108, does show a significantly worse performance for both the training and validation precision of the regularized model. The only improvement is that the regularized model exhibits a lower variance.

When analysing the validation outputs, as visualized in Figure 111 until 120, the first thing that stands out in the true versus predicted plots is the fact that the spread increases when adding the regularization. Furthermore, the points do not follow a horizontal trend anymore.

If we consider the plots per slice-timepoint pair, we immediately notice that the patterns show much more resemblance. Apart from the fact that the predicted values often have a narrower range, we can clearly recognise parts of the true value pattern in the predicted values. The model without regularization shows greater overlap between these patterns than when adding regularization.

CNN - channel per repetition

Next, we analyse the CNN_{reps} model, of which the performance curves and outputs are

highlighted in Appendix C.3. This model is not directly comparable to the previous two CNN's because of the difference in sampling strategy. The batches that are used for training this model do not consist of the full set of tensors of one given patient, but are constituted by sampling 64 tensors from a pool of tensors of all training patients, slices and b-values together. Each tensor thus represents the six repeated measurements for that specific patient, slice and b-value. Furthermore, it is important to note that the regularized model is trained for 1500 epochs instead of the 500 epochs for the model without regularization.

The first thing that stands out when inspecting the performance curves is that especially the training curves are much stabler and show less uncertainty compared to the previous two CNN's. This can be explained by the fact that the tensors within a batch differ more than the tensors within a patient. Therefore, the model is immediately forced to learn to generalize, which makes that its performance does not rely on overfitting on one patient, or one fixed batch of tensors, before fitting the next patient.

If we consider the loss curves, which are visualized in Figure 121 and 122, we notice very smooth curves. However, we do notice that the validation curves do not improve over the epochs. Moreover, the training loss for the regularized model is slightly higher than for the model without regularization, but the validation losses are almost identical. Hence, the regularization brings minimal improvement to the overfit, but does increase the uncertainty.

Considering the accuracy, highlighted in Figure 123 and 124, we again observe a worse training performance after regularizing the model. Furthermore, the regularization clearly increases the variance. However, although being much more uncertain, the validation accuracy for the regularized model is higher. Therefore, the discrepancy between training and validation performance for the regularized model is decreased.

Regarding the F1-score, which is visualized in Figure 125 and 126, we again observe a decrease in training performance when regularizing the model. Furthermore, the validation score and the variance worsen too. For the remaining performance metrics, we observe the same behaviour.

If we inspect the validation outputs, which are visualized in Figure 135 until 144, we can

conclude from the true versus predicted plots that the positioning of the points is significantly better, especially for the model without regularization. The datapoints show a diagonal trend, where we aim for. However, they do show quite some spread, which testifies of uncertainty while predicting. The spread slightly decreases when adding regularization, although the resulting predictions are often a bit too low.

We conclude that the regularization affects the speed of convergence, since after 500 epochs the regularized curves have not converged as much as the curves from the model without regularization have after the same number of epochs. Therefore, it also delays the overfitting. However, this unfortunately does not result in the model achieving better validation performances in the meantime when comparing it to the ones of the model without regularization. Hence, we can conclude that the regularization just challenges the model by adding extra randomness, and fails in forcing the model to learn to generalize to new data.

However, because of the delayed overfitting the model can be trained for more epochs, which results in predicted patterns that look more similar to the actual patterns in the data. This can be noticed when comparing Figure 137b until 140c to Figure 141b until 144c. Apart from the fact that the predicted values are often not in the exact same range as the actual values, we can visually recognize similar patterns between the predicted and true values.

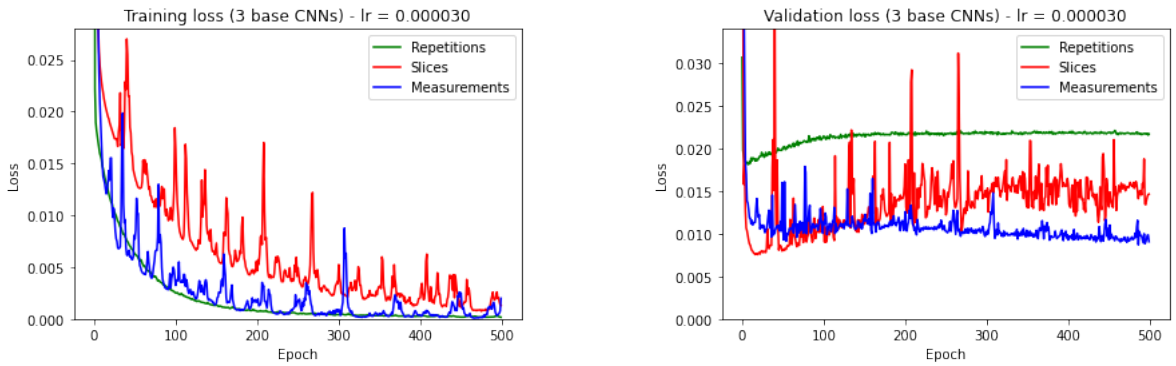
3 CNNs

In this section we compare the performances and outcomes of the three previously discussed CNN's - both without and with regularization - in order to pass judgement on which model performs best. Besides the aforementioned difference in sampling strategy, the models are trained using identical settings to allow for a fair comparison.

Starting with the loss, which is visualised in Figure 35 and 36, we immediately notice the difference in smoothness of the curves between the models. As mentioned before, the CNN_{reps} model exhibits a much smoother learning process, which involves less uncertainty. Without regularization this model outperforms the CNN_{slices} model in terms of training loss, while showing an almost equal performance to CNN_{meas} except for the level of uncertainty.

When adding the regularization, CNN_{meas} suddenly shows the worst training loss. CNN_{reps} clearly outperforms the other two models if we consider the regularized versions and CNN_{slices} scores precisely in between the other two models.

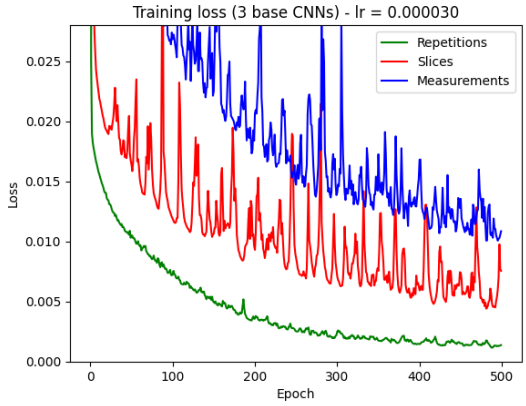
If we consider the validation losses, CNN_{reps} evidently shows the worst loss for the models without regularization, yielding a value almost twice as high as the CNN_{meas} validation loss. This time the CNN_{slices} starts off worst, but begins to outperform CNN_{slices} after around 150 epochs. The latter shows a slightly increasing trend, which hints at overfitting. Moreover, the regularized versions of these models exhibit similar behaviour for the validation loss. However, in this case both CNN_{slices} and CNN_{meas} have increasing curves and CNN_{reps} has a curve that starts decreasing slightly after the first 150 epochs.



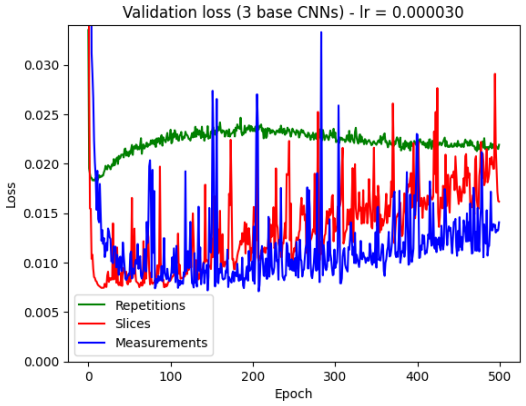
(a) Training loss

(b) Validation loss

Figure 35: Training and validation loss of the 3 CNN variants (MSE loss)



(a) Training loss - with regularization

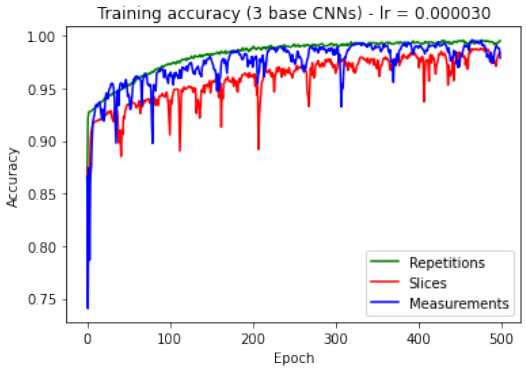


(b) Validation loss - with regularization

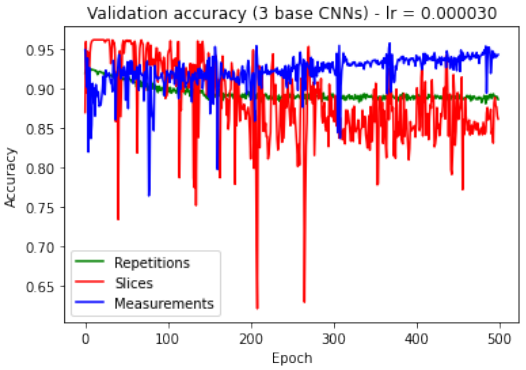
Figure 36: Training and validation loss of the 3 CNN variants - with regularization (MSE loss)

If we consider the accuracy, which is visualized in Figure 37 and 38, we observe very similar behaviour for the training curves as for the loss training curves.

Regarding the validation accuracy, CNN_{meas} again outperforms the other models. For the models without regularization, it is the only model with an increasing accuracy curve. Moreover, CNN_{slices} has a much larger variance than the other models, especially with regularization.

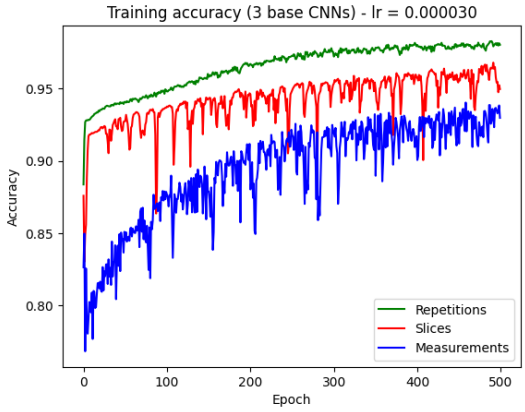


(a) Training accuracy

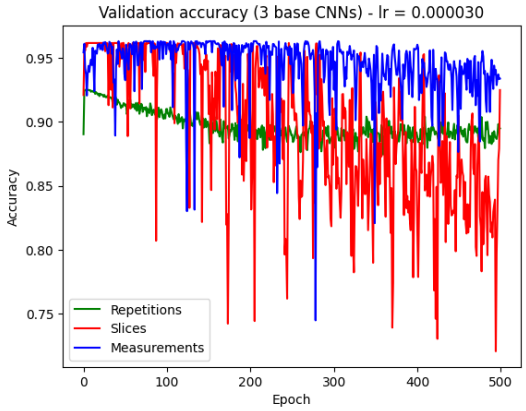


(b) Validation accuracy

Figure 37: Training and validation accuracy of the 3 CNN variants (MSE loss)



(a) Training accuracy - with regularization



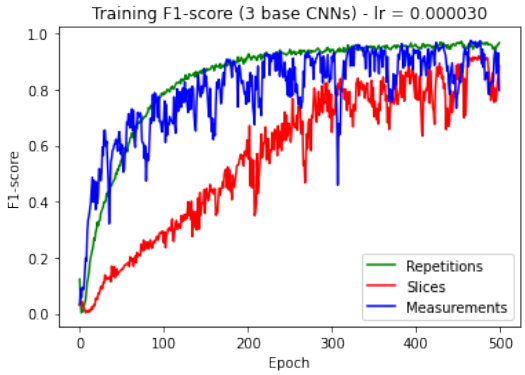
(b) Validation accuracy - with regularization

Figure 38: Training and validation accuracy of the 3 CNN variants - with regularization (MSE loss)

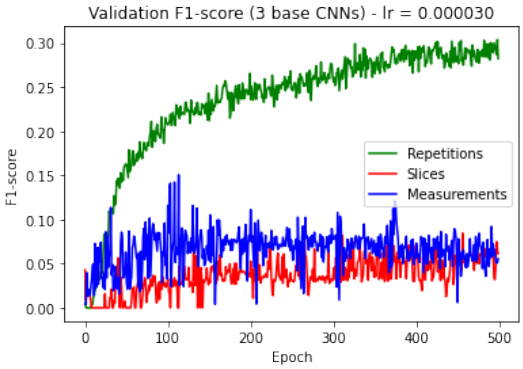
For the F1-score, which is previewed in Figure 39 and 40, we observe different patterns. The training curves for the model without regularization do have the same order as for the accuracy, but the CNN_{slices} curve has a deviating shape during the first half of the epochs. For the regularized curves, the CNN_{meas} model performs best at the start. However, as the epochs pass, the CNN_{reps} strongly outperforms the other two models. Furthermore, the CNN_{slices} surpasses the CNN_{meas} after around 300 epochs.

If we consider the validation performance on the F1-score, we can obviously conclude that the CNN_{reps} model's performance, both without and with regularization, highly exceeds the performances of the other two models. Without regularization the CNN_{meas} model firstly outperforms the CNN_{slices} model, but they converge towards each other eventually. With regularization, CNN_{slices} outperforms CNN_{meas}. However, during the last 100 epochs CNN_{meas} starts increasing slightly and therefore approximates the CNN_{slices} curve.

The validation F1-score of CNN_{meas} decreases when regularizing the model. For CNN_{slices}, it increases and for CNN_{reps} there is almost no difference.

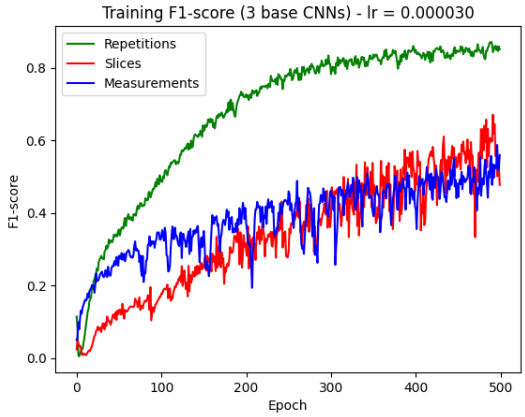


(a) Training F1-score

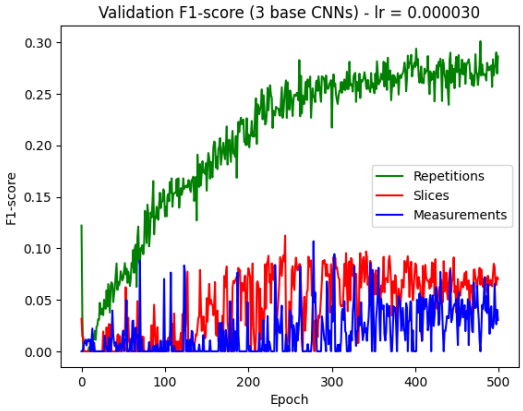


(b) Validation F1-score

Figure 39: Training and validation F1-score of the 3 CNN variants (MSE loss)



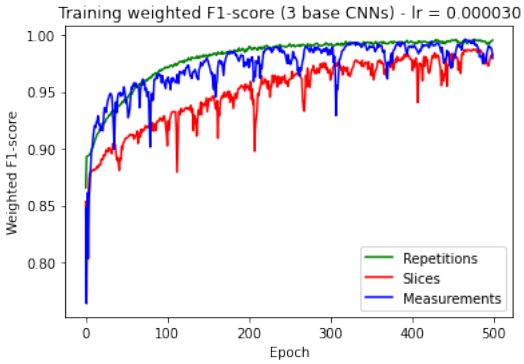
(a) Training F1-score - with regularization



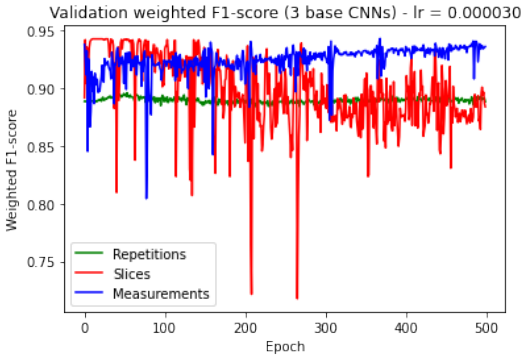
(b) Validation F1-score - with regularization

Figure 40: Training and validation F1-score of the 3 CNN variants - with regularization (MSE loss)

The weighted F1-score, as previewed in Figure 41 and 42, exhibits nearly identical patterns to the accuracy, and therefore the previously discussed observations and conclusions in that paragraph hold for this metric too.

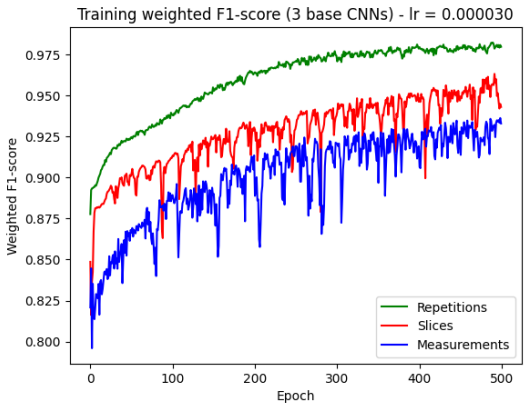


(a) Training weighted F1-score

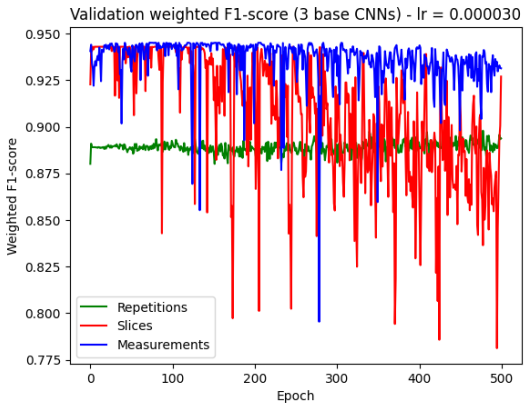


(b) Validation weighted F1-score

Figure 41: Training and validation weighted F1-score of the 3 CNN variants (MSE loss)



(a) Training weighted F1-score - with regularization

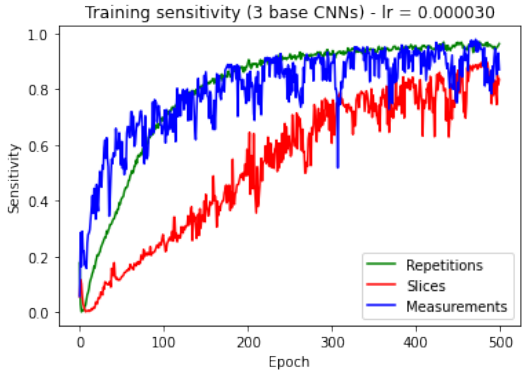


(b) Validation weighted F1-score - with regularization

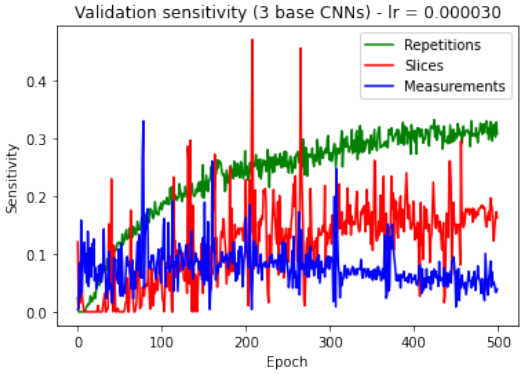
Figure 42: Training and validation weighted F1-score of the 3 CNN variants - with regularization (MSE loss)

By observing the sensitivity, highlighted in Figure 43 and 44, we can conclude that regularizing the models badly influences the training sensitivity. The validation sensitivity remains rather stable, although the trends of the curves do change. Where CNN_{slices} and CNN_{meas} without regularization do not necessarily have increasing validation curves, their regularized counterparts do show an increase over the epochs. However, the level of uncertainty in the validation curves of these two models is very high. This means that although the model becomes better at predicting positive labels, it does not necessarily become more certain in predicting these labels. We expect this

to be reflected in the precision curves.

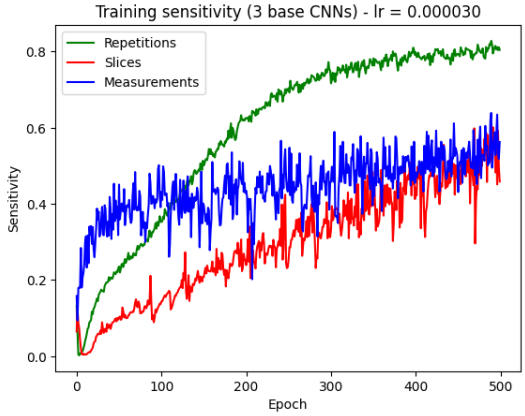


(a) Training sensitivity

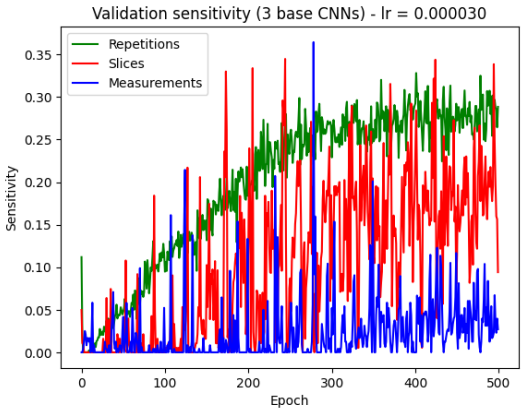


(b) Validation sensitivity

Figure 43: Training and validation sensitivity of the 3 CNN variants (MSE loss)



(a) Training sensitivity - with regularization



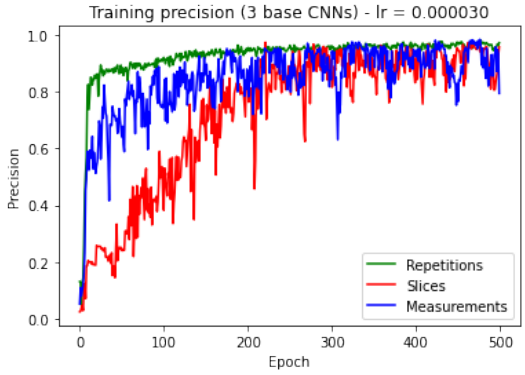
(b) Validation sensitivity - with regularization

Figure 44: Training and validation sensitivity of the 3 CNN variants - with regularization (MSE loss)

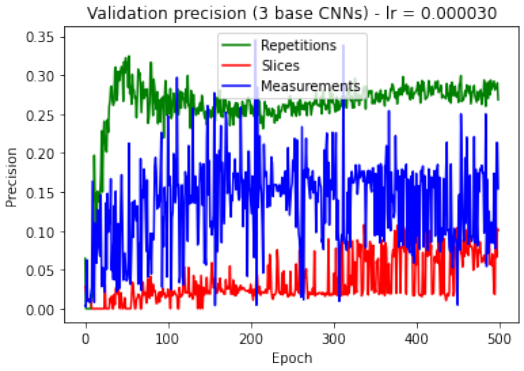
If we consider the precision, which can be found in Figure 45 and 46, we see that especially CNN_{slices}'s precision is affected by the regularization. Both the training and validation precision nearly halve when adding regularization to the model. In this case, the regularization leads to a smaller variance, which is contrary to most of the earlier-mentioned performances. Furthermore, the CNN_{reps} model outperforms the

other models in both the training and validation phase.

Moreover, as we expected the validation precision remains stable over the epochs. Except for a slightly increasing validation curve for the CNN_{reps} model with regularization, all other curves do not exhibit an increasing trend. This means that the models' predictions do not become more certain and therefore, the models have difficulties with learning to adapt to new data.

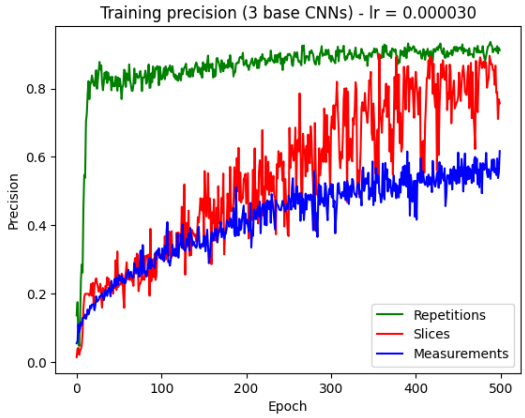


(a) Training precision

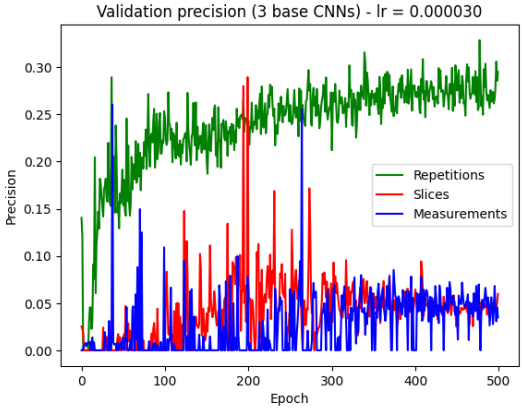


(b) Validation precision

Figure 45: Training and validation precision of the 3 CNN variants (MSE loss)



(a) Training precision - with regularization

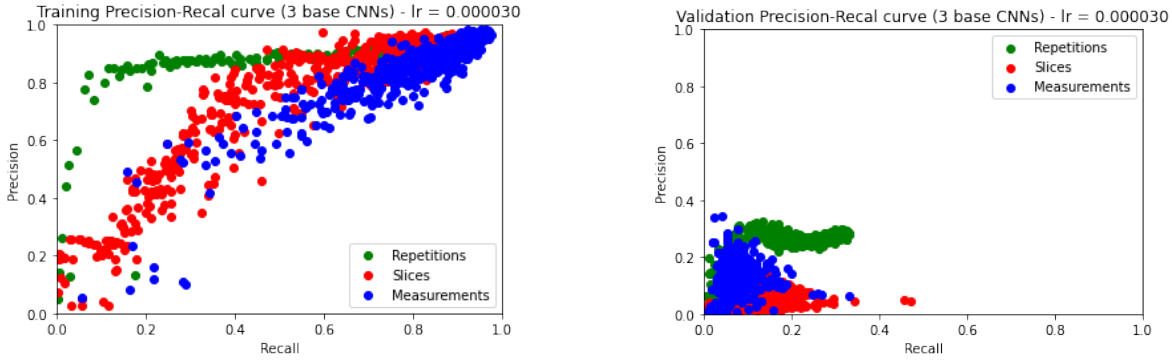


(b) Validation precision - with regularization

Figure 46: Training and validation precision of the 3 CNN variants - with regularization (MSE loss)

Lastly, we consider the precision versus sensitivity plots previewed in Figure 47 and 48. Ideally, we want to reach the top right corner of the plot, which embodies a model that is very good at correctly detecting outliers. Therefore, the training curves for the models without regularization testify of a good performance. If we consider the validation counterpart, however, we observe a poor performance. Especially since the differences between the training and validation curves are huge.

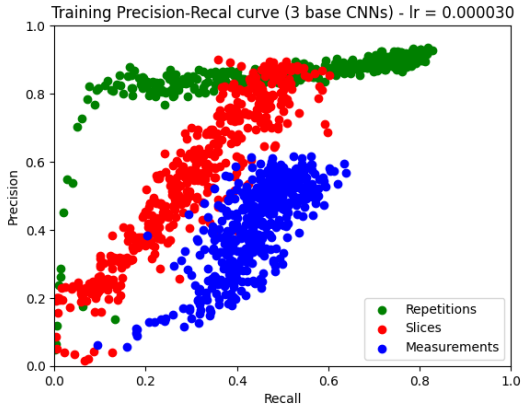
If we consider the regularized versions of the models, we observe a similar training pattern for CNN_{reps} compared to the model without regularization. The validation patterns are quite similar too, although the regularized version achieves a slightly lower precision and slightly higher variance. For CNN_{meas}, the training performance significantly dropped when adding the regularization, yielding scores that are half as good as without the regularization. Although the variance decreases for the validation phase of the regularized version of this model, the achieved scores do too. Lastly, we consider the CNN_{slices} model, for which the training performance is badly influenced by the regularization too. Furthermore, the regularization does not seem to affect the validation performance much.



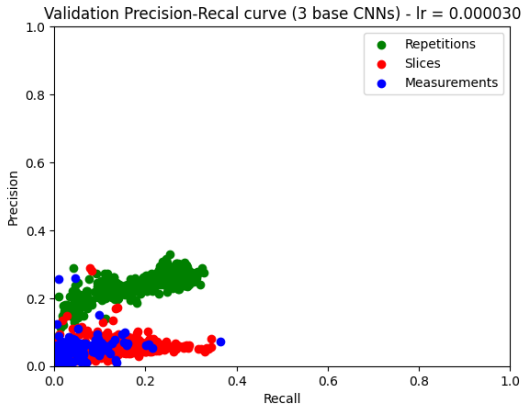
(a) Training precision vs sensitivity

(b) Validation precision vs sensitivity

Figure 47: Training and validation precision vs sensitivity of the 3 CNN variants (MSE loss)



(a) Training precision vs sensitivity - with regularization



(b) Validation precision vs sensitivity - with regularization

Figure 48: Training and validation precision vs sensitivity of the 3 CNN variants - with regularization (MSE loss)

Finally, we discuss which model is preferred. Having considered all the metrics and outputs, and taking into account our previously determined success criteria (Section 4.2.5), we prefer the CNN_{reps} model. This choice is motivated by a few arguments.

First of all, our aim was to develop a model that excels at correctly detecting outliers. Therefore, we mainly focus on the precision and sensitivity. Since the F1-score is a harmonic mean of these two metrics, we also strive for a high F1-score. Having considered these metrics in the foregoing paragraphs, we conclude that CNN_{reps} outperforms the other models in terms of these metrics. Additionally, CNN_{reps} demonstrates significantly more stable performance curves, which means a smoother training and validation process, and consequently, less uncertainty.

Secondly, CNN_{reps} suffers less from overfitting. As a result, the difference between training and validation performance is smaller, which is important since we are interested in the model's performance on new data. Because of the reduced overfitting, its validation curves keep improving over the epochs, which is not the case for the other two models. Therefore, it pays to continue training the model.

Moreover, CNN_{reps} offers more flexibility both in terms of input data and model tuning. Since the model uses a data pool of all patients, slices and b-values together, it

does not rely on completeness of a patient's data. The only requirement for the tensors in the batch is to have six repeated measurements and equal slice shapes. Besides that, it does not matter how many patients there are and how many slices or b-values they contain, which makes the model more flexible compared to the other two CNNs. This also reduces the amount of lost data. Furthermore, the model can be tuned by varying the batch size or the sampling strategy. Additionally, the model can be optimised by tuning the model hyperparameters, as is the case for the other two models too.

Lastly, adding regularization to the model reduces the overfit. However, the regularized model seems to exhibit slightly more uncertainty. Furthermore, the regularization does not necessarily enhance the model performance, since the metrics are quite similar. On the other hand, by the end of the 500 epochs, the validation curves for the regularized model demonstrate a more apparent increase than those of the model without regularization. Accordingly, the regularized model has greater potential for improved performance with further training.

After the two CNN_{reps} variants, the CNN_{meas} without regularization would be our model of preference. Overall, the model's prediction range is in general somewhat too low and narrow, resulting in an excess of negative labels and strongly affecting the performance metrics. However, if we consider the predicted values per slice-timepoint pair and compare them to their true counterparts, we can conclude that the model does well at learning and predicting the patterns in the data. During validation, the predicted patterns show significant similarities to the actual data, which suggests that the model is suitable for the task at hand.

5.4 The influence of incorporating spatio-temporal information

To examine the effects of the treatment of spatial, temporal and spatio-temporal information on the models' sensitivity to detecting artifacts, we discuss the relevant outcomes of the scaling methods, local versus global part and the neural networks, which each treat the data's dimensions differently.

5.4.1 Scaling methods

By fitting the IVIM model and scaling the residuals voxel-wise, we solely exploit the temporal information. Using the knowledge we have about the b-value at which a certain slice is measured by grouping the timepoints with an equal b-value, allows us to scale the residuals to facilitate a fair comparison between measurements of different b-groups. Furthermore, this fair comparison enhances the model's sensitivity to detecting outliers, since the level of expected distortion depends on the b-value. For instance, at a b-value with a low level of distortion, medium deviations are not labeled as unexpectedly large without accounting for the expected level of distortion.

The spatial information is only incorporated by considering the mean z-score per slice for each measurement, instead of per voxel. This is a rather simplified representation, since all information contained in the slice is aggregated. However, in this case the simplification is not necessarily problematic, since we are interested in filtering the most distorted slices, in order to remeasure them. Therefore, a mean z-score suffices for this purpose, since it quantifies how distorted the slice as a whole is. This means that a very poor local region or a slightly distorted global region could result in an equal mean z-score, which is not a problem because we want to eliminate both phenomena.

5.4.2 Local versus global

Taking into account the spatial information about how many voxels in the slice are outliers certainly provides additional and useful information, especially when considering the normalized ratio of outlier voxels. However, this information primarily provides us with knowledge on which type of artifact is detected and does not necessarily enhance the model's performance in terms of improving the sensitivity to detecting artifacts.

If we would switch to a multi-class classification model in order to label the main type of artifact present in the slice, we hypothesize that including the normalized ratio as additional input could enhance the model's sensitivity to detect certain types of artifacts, namely, the local ones.

5.4.3 Convolutions

When comparing the base models which take flattened voxel tensors as input to the CNNs that take 2D slices as input and combining all results, we can conclude that the CNNs outperform the base models. The base models are much more prone to overfitting than the CNNs are. They seem to have difficulties with learning how to generalise to new data, which hints at failing to recognise patterns in the data. This might be due to the loss of information about the slice structure when flattening the voxel tensor.

An important remark to make is that the base models function as classifiers and the CNNs as regressors. However, indirectly they have the same objective of minimising the difference between predicted z-labels/mean z-scores and true z-labels/mean z-scores. Since the performance metrics for both types of models are computed by using the z-labels, which are either directly predicted or obtained by subjecting the mean z-scores to the threshold, we are able to compare the performances of all models rather directly. Therefore, the differences in model architectures seem rather small and a significant proportion of the aforementioned performance improvement seems to be due to using convolutions in order to retain the slice structure.

Purely in terms of enhancing the sensitivity, the effect of using 2D slices is not that clearly visible. The base models converge to sensitivities of around, respectively, 0.3, 0.1 and 0.12. The CNNs yield sensitivities of 0.2 (versus 0.2 with regularization), 0.08 (versus 0.1) and 0.33 (versus 0.25). Overall, the CNNs obtain slightly higher sensitivity scores, but the differences are small. However, paired with these sensitivities, the CNNs do exhibit significantly higher precisions. This is relevant to the aim of correctly detecting as many artifacts as possible.

Considering the influence of which dimension is handled through the separate input channels, we discuss the assumptions made for the three CNN architectures and their effects. Starting with CNN_{slices}, which assumes independence between the measurements. Each input channel is responsible for one of the slices and learns its features by considering a batch of measurements for that slice. Subsequently, the learned features are passed on to all entries in the next layer of the network. Therefore, the slice dimension is incorporated in the next layers. Since the convolutions happen within the slices, the network exploits all three spatial dimension.

Conversely, the CNN_{meas} has an input channel per measurement and assumes independence between the slices. Similarly, it exploits the two dimensions present in a slice, moreover it exploits the temporal information present in the measurements. The CNN_{reps} has a comparable architecture, since it uses an input channel per repeated measurement of a certain b-value and the repeated measurements constitute a temporal dimension too. However, now the temporal information is grouped and split per b-value. Therefore, it assumes independence between the slices as well as the different b-groups.

Comparing the different ways of treating the spatial and temporal dimensions, we can state that incorporating the information we have about the b-values by splitting the timepoints on their b-value enhances the sensitivity significantly. We cannot assign all improvement to this additional source of information, because of the difference in batching strategies between the models. However, it does seem beneficial to treat the repetitions through input channels and therefore, assume independence between the measurements of different b-values and the slices. By using this approach, we exploit two spatial dimensions, the slice structure, and one temporal dimension, the repetitions.

Furthermore, CNN_{meas} outperforms CNN_{slices} at capturing the patterns in the data, although the latter achieves a higher sensitivity. This can be explained by the fact that CNN_{meas} assumes independence between the slices, where CNN_{slices} does not. Since the aforementioned interleaved pattern highly distorts the relationship between two consecutive slices, we hypothesize that it is highly confusing for the model to learn this relationship. We we have to learn this relationship, if we do not assume independence between these slices. Between the measurements there is much less distortion, which makes it easier to learn this relationship.

5.5 Real-time applicability

An important aspect of the model to discuss is the real-time applicability. Therefore, we consider the computational efficiency and the reliability of the models.

5.5.1 Computational efficiency

Data labeling

The first step in the data labeling part of the research is fitting the patients to retrieve the IVIM parameters and corresponding residuals. As detailed earlier, we tested two different fitting algorithms - LSQ and IVIM-Net, and discovered that IVIM-Net was more computationally exhaustive while not significantly improving the model's ability to detect outliers. In Table 4, we can see the increased runtime for IVIM-Net compared to LSQ when fitting patient 1, which is computationally-wise a relatively non-exhaustive patient for its relatively low number of valid voxels to be fit. As a reference, patient 1 has about 85.000 valid voxels, while patient 11, for instance, has 720.000 valid voxels to be fit, which highly enlarges the difference in runtime between the two fitting algorithms. This difference in efficiency underpins our choice to continue with LSQ fitting.

Table 4: Computational demands of fitting patient 1 by using LSQ/IVIM-Net as fitting algorithm.

Fitting algorithm	**Runtime**
LSQ	3 min 17 sec
IVIM-Net	3 min 55 sec

After deploying the LSQ algorithm, which takes some time to fit the patients initially, we still have to preprocess the data, scale the residuals and visualise the outcomes. All these steps evidently increase the computational demands. Moreover, this approach relies on completeness of the data in order to scale the residuals well-foundedly, which implies that we would have to perform a full scan before this algorithm could be deployed. Subsequently, we would fit the data and generate the z-scores in order to decide on which slice-timepoint pair(s) need to be re-measured. Therefore, this process has to be as efficient as possible to avoid delaying the total scanning procedure exorbitantly.

Furthermore, this algorithm operates on one patients at a time, the current patient, which means that the data always has to be fit and processed in real-time starting from zero. Conversely, the neural networks from the second part could be trained on historical data and applied to the data of the current patient, subsequently.

Outlier detection

This part starts by fitting all patients that form the training and validation data once using the data labeling algorithm and storing the resulting z-scores. Hence, the start-up phase is computationally exhaustive. However, after the z-scores have been written to the dedicated output files, we can easily access them at any time. From there on, the idea is to train the model on the data of these patients - the historical data, and subsequently predict the z-scores for the current patient in real-time. Thus, it is not necessary to start from zero every time, which highly increases the efficiency and hence, decreases the runtime.

In Table 5, the runtime in minutes for the full training and validation phase are detailed for all seven neural networks. The models are trained during 500 epochs. The first six models iterate over the 19 patients, while the CNN_{reps} iterates over the 42 batches that we sampled. Furthermore, the number of tensors per iteration differs and so does the tensor shape. As mentioned before, the Base regressor takes about an hour per epoch, which makes it infeasible for the purpose of our research.

Considering the remaining six models, we can conclude that the two-class model and grouped b model have the shortest runtime and CNN_{reps} the longest. However, CNN_{reps} executes more than twice as many iterations as the other models. Furthermore, CNN_{reps} is the most memory-efficient and CNN_{meas} the least, because of its large number of input channels.

Table 5: Computational demands of the base models and CNNs when trained on a 40GB GPU.

Model	**Epochs**	**Iterations**	**Batch size**	**Parameters**	**Runtime (min)**
Base regressor	500	19	(54, 18)	860.0M	1h/epoch
Base classifier	500	19	(54, 18)	430.0M	12
Two-class model	500	19	(54, 18)	430.0M	7.5
Grouped b model	500	19	(54, 18)	430.0M	7.75
CNN_{slices}	500	19	54	24.0M	10
CNN_{meas}	500	19	18	72.5M	12
CNN_{reps}	500	42	64	11.2M	16

As long as the models suffer from overfitting, on the other hand, we cannot draw any final conclusions on how the computational demands relate to real-time application. One of the potential resolutions for the overfitting is to use more training data. Evidently, this would increase the computational demands and hence, the runtime. However, this could also accelerate the convergence towards a good validation performance, which might lead to a smaller number of required epochs.

5.5.2 Reliability

Since we have not been able to resolve the overfitting of the models during the time span of this research, we were not able to investigate the reliability of the predictions in depth. Therefore, we cannot draw informed conclusions on what information is essential in order to facilitate reliable detection. However, in the next paragraphs we discuss a few relevant insights we obtained throughout the process.

As previously noted, having complete patient data is crucial for ensuring reliability when scaling residuals in the data labeling part. Incomplete data can lead to biased z-scores, which undermines the reliability of the resulting outliers. Therefore, it is essential to consider all measurements when determining the z-scores.

For the outlier detection part of this research, we state that the more similar the patients' dimensions and b-schemes are, the more reliable the predictions will be. Varying dimensions lead to, for instance, padded slices or the loss of information. Furthermore, differences in b-schemes lead to higher variability in the data, which complicate the model's learning process, enhance uncertainty and hence, weaken reliability. This can be explained by the fact that the b-schemes are decisive for which information is captured during the measurements. Furthermore, the CNN_{reps} model for example requires uniform b-schemes to ensure six repeated measurements per b-value. The more deviating a given b-scheme is, the more it will be corrupted by our previously described method for uniformity in the b-schemes. Logically, increased corruption leads to decreased reliability.

Considering the essential information for the base models, we can state that the only requirement is that the shape of the slices is $144x144$, leading to a flattened voxel tensor

of length 20.736. The number of measurements and number of slices are not restricted, since these dimensions are treated determine the batch size.

Moreover, we consider the essential information for the CNNs. First of all, all three CNNs require the patients to have slices of shape $144x144$. For CNN_{slices}, it is essential to have exactly 18 slices per patient, since the network consists of 18 separate input channels. Likewise, CNN_{meas} needs exactly 54 measurements for the input channels. Conversely, CNN_{reps} requires exactly 6 repeated measurements, which is ensured by making the b-schemes uniform. Besides that, this model treats the data batch-wise instead of per patient, which enhances the flexibility of the model regarding the required completeness of the data.

6 Conclusion and recommendations

6.1 Conclusions

This project has focused on answering the question of how deep learning techniques can be used to accurately detect artifacts in 4D IVIM MRI. This was done by answering the four subquestions, of which we summarize the main conclusions in the next paragraphs.

1. Can the residuals of an IVIM fitting algorithm direct us towards the presence of an artifact?

We can conclude that the residuals of an IVIM fitting algorithm, LSQ in this case, can direct us towards the presence of an artifact. We discovered that scaling the residuals of different b-groups by using the absolute z-score is a suitable approach to enable this. Additionally, this approach demonstrates promising potential to operate as an automated data labeling model, thereby alleviating the shortage of labeled data in the medical field. Moreover, it could provide pseudo-labels for our unlabeled data, twhich would facilitate the development of deep learning alternatives.

2. Can deep learning techniques provide a reliable solution for automated outlier detection in IVIM MRI?

We believe that deep learning techniques can provide a reliable solution for automated outlier detection in IVIM MRI. However, we have not succeeded in developing a deep learning alternative that approaches the performance of our data labeling model yet. The main reason for this is the significant overfit of the models, which we have not been able to resolve during the time span of this project. However, the models are able to achieve a perfect training performance for all deployed metrics, which testifies of their potential to provide a reliable solution for this task. Furthermore, the deep learning models show potential to operate in real-time by providing a faster alternative that is promising in terms of scalability and generalizability. Moreover, they are able to handle higher-dimensional data, which minimizes the loss of spatial and temporal information. However, further research is necessary to be able to provide a definite answer on the reliability of the deep learning models in this research.

Summarizing, the deep learning models provide an alternative that outperforms the data labeling model regarding their suitability to clinical adoption. Incorporating spatial context could potentially improve the models' performance in detecting outliers. However, due to the unresolved overfitting, the models cannot yet serve as the promising alternative they could be, as they are currently unreliable.

3. Can incorporating spatio-temporal information of the data enhance our models' sensitivity to detecting artifacts?

Upon reviewing the influence of the integration of spatial, temporal, and spatio-temporal information on a model's sensitivity to detecting artifacts, we discovered that preserving the 2D structure (x- and y-dimension) of the slices enhances the model's performance. Furthermore, the predictions of the models that treated the slices as independent, for instance CNN_{meas} and CNN_{reps}, exhibited patterns significantly more similar to those in the actual data. This implies that the slice-dimension (z-dimension) is less essential to incorporate in the deep learning models than the other dimensions. Furthermore, we observed that including the knowledge we have on the b-values improves the model's performance. However, it would be interesting to further investigate the inclusion of spatio-temporal information, for instance by examining the effects of using 3D convolutions.

4. What is considered essential information to facilitate reliable detection?

As previously stated, we are not able to draw definite conclusions on what is considered as essential information to facilitate reliable detection because of the overfitting issue. Beyond the architectural requirements of each model for essential information, resolving the overfitting issue is necessary before we can further explore for instance the required number of measurements for reliability. Investigating this topic in depth would provide us with valuable insights on the applicability of the models in real-time.

In conclusion, we developed a model that is able to automatically label our data and filter the artifacts, which was a primary goal of this research. In existing literature, the lack of a golden standard solution to this problem is addressed as a key bottleneck in the research area of automated outlier detection in medical data. The solution that we proposed - the absolute z-score model - is still absent in existing literature and can

therefore be seen as a contribution.

Our attempt to develop a more efficient deep learning alternative to our proposed golden standard model that is able to achieve a comparable performance did not succeed during the time span of this project. Nevertheless, the developed deep learning models exhibit potential to effectively function as automated outlier detection model for IVIM MRI data. In order to answer the question whether these models could indeed provide a reliable solution to this task, further research is needed. This research should focus on resolving the overfitting issue first to facilitate a fair assessment of the performance of the models.

Ultimately, the deep learning models provided us with valuable insights regarding suitability of the models to clinical practice. Moreover, these models are more efficient and have an enhanced scalability and generalizability. Furthermore, they do not rely on completeness of the patient's data, which increases their flexibility and enhances their clinical applicability.

6.2 Limitations

There are several limitations inherent in this research. First of all, the nature of this research is exploratory. Therefore, we focused on exploring new approaches to automated outlier detection in 4D IVIM MRI. The lack of existing literature on this specific topic testifies of the need and the complexity of this task. Due to the limited amount of time, our aim was to implement the intended approaches and investigate and asses their potential and suitability to this task. As a consequence, the validation of the models' performances is limited.

Given the limitations associated with specific architectural choices of the models, we begin with the absolute z-score. As previously detailed, using the z-score limits the number of poor labels predicted. The z-score functions as an indication of how deviating a given measurement is from the other measurements in that b-group, but does not necessarily tell how poor these measurements actually are. Therefore, only the worst few measurements in that b-group can and will have a z-score greater than one. Hence, eventually we filter the worst few measurements per b-value, although some of them

might not be among the worst few measurements overall. Conversely, we might not filter some measurements of a b-group that should have been filtered when considering their quality globally.

A primary limitation of the deep learning models is the required uniformity in slice shapes and b-schemes, and for some models the number of slices or measurements. This limitation reduces the flexibility in input data and hence, the generalizability. Furthermore, ensuring dimensional uniformity between the patients leads to reduced data quality due to loss of information and data corruption.

Another limitation at this point is the runtime. Most of the proposed deep learning models require between 10 and 16 minutes for the training and validation process. Combined with a comparable scan time in real-time, this runtime is too long for clinical adoption. Moreover, adding more training data to the models, which is one of the proposed resolutions to the overfitting issue, would increase the computational demands, but might also decrease the number of epochs needed for convergence. Therefore, we cannot draw a definite conclusion on the length of the runtime yet.

Lastly, the overfitting issue of the deep learning models is the major limitation. All deep learning models strongly overfit on the training data. The discrepancy between training and validation performance is large and the validation performance is noisy and unstable. Therefore, all deep learning models are currently unreliable, which hinders us in assessing their performance, feasibility and potential.

One of the reasons that the models overfit is probably the class imbalance in the training data. The class imbalance, originating from the fact that the majority of mean z-scores is lower than one, leads to the models overfitting the majority class, which leads to a poor generalization to unseen data - the validation data. As a result, the models struggle to predict high mean z-scores, resulting in predicted scores that are often too low in general. This observation aligns with our analysis of the models' results.

6.3 Future research

Future work on improving the proposed deep learning models should prioritize resolving the overfitting issue. An obvious first step would be to assemble more training data. Alternatively, we could try learning the z-score for each voxel and then calculate the mean z-score, rather than learning the mean z-score directly. While this approach would increase computational demands, it would also expand the amount of training data. Moreover, it could be more difficult for the models to learn features from the aggregated and simplified mean z-scores than from the original voxel-specific z-scores.

If the previously suggested changes do not adequately resolve the overfitting, we propose exploring options to address the class imbalance in the data to reduce overfitting. Options to consider include data augmentation techniques or, for instance, oversampling the minority class. Otherwise, we could consider using alternative loss functions, such as focal dice loss, which is particularly suited for imbalanced data.

For optimizing the models, tuning the hyperparameters can enhance their performance. Among other parameters, the learning rate, dropout, and regularization rate can be tuned using, for instance, a coarse-to-fine optimization approach. Moreover, optimizing the efficiency of the model architectures through additional research could further reduce the runtime.

Furthermore, in order to treat the spatial information in the slices in more detail, it might also be interesting to test a patch-wise learning approach instead of predicting a value per slice. Because of the limited time available, we could not test this approach. However, we hypothesize that treating the spatial information in the slice patch-wise, rather than reducing it to a single value, could primarily enhance the model's ability to detect local artifacts. By narrowing the region for which the z-score is aggregated, the resulting mean z-scores will become more precise. Imagine a very small artifactual region within a slice that predominantly contains low z-scores. Depending on the distortion level in the artifact, this will likely lead to a mean z-score below our threshold, causing the local artifact to go undetected.

An interesting extension of our approach would be to consider multiple orientations of the data when determining the outliers. In this way, we could leverage the 3D in-

formation from the neighborhood. This can be done, for instance, by training three separate 2D CNNs using 2D slices from three different orientations: axial, coronal and sagittal. Subsequently, the final predictions can be determined by taking the average of the probabilities across the three orientations. In our case, especially incorporating the coronal orientation might be interesting, as the interleaved motion artifacts are most distinctly visible in the coronal view. Therefore, this extension could enhance the sensitivity to detecting this type of artifact.

Additionally, it would be valuable to investigate extensions that facilitate flexible input dimensions, such as Neural Controlled Differential Equations (NCDE). This would remove the restriction of requiring an equal number of units in the dimension processed by the input channels for each tensor. For instance, it would no longer be necessary to require uniform b-schemes to ensure each input tensor has an equal number of repeated measurements to match the input channels. This extension would therefore highly enhance the models' flexibility and generalizability, which means improved real-time applicability.

A final suggestion for a possible extension of the models is extending them from 2D to 3D CNNs as three-dimensional convolutions are particularly suited to capture spatio-temporal information. A downside of using 3D CNNs is the increased computational demands, sometimes without a significant improvement in performance. An efficient and promising alternative is to investigate 2D+1 CNNs, which split the convolutions into a spatial 2D convolution and a temporal 1D convolution.

In conclusion, future research on improving the proposed neural networks should prioritize resolving the overfitting issue. Besides this, the models could be improved by addressing class imbalance, tuning hyperparameters and optimizing efficiency. Moreover, there are several interesting and promising potential extensions that could enhance various aspects of the models' performance. Improving the models and thereby obtaining reliable predictions would allow for an in-depth investigation into their real-time applicability.

References

Ahmad, A., et al. (2023). 3D-QCNet: A pipeline for automated artifact detection in diffusion MRI images. *Computerized Medical Imaging and Graphics, 103*, 102151.

Akkus, Z., et al. (2017). Deep learning for brain MRI segmentation: state of the art and future directions. *J. Digit. Imaging, 30*(4), 449-459. doi: https://doi.org/10.1007/s10278-017-9983-4.

Alfaro-Almagro, F., et al. (2018). Image processing and quality control for the first 10,000 brain imaging datasets from UK Biobank. *Neuroimage, 166*, 400–424. 10.1016/j.neuroimage.2017.10.034.

An, J., et al. (2015). Variational Autoencoder based Anomaly Detection using Reconstruction Probability. *Special Lecture on IE, 2*(1), 1–18.

Andersson, J., et al. (2003). How to correct susceptibility distortions in spin-echo echo-planar images: application to diffusion tensor imaging. *Neuroimage, 20*, 870–888. 10.1016/S1053-8119(03)00336-7.

André, E., et al. (2014). Influence of Noise Correction on Intra- and Inter-Subject Variability of Quantitative Metrics in Diffusion Kurtosis Imaging. *PLoS ONE, 9*(4), e94531. doi: https://doi.org/10.1371/journal.pone.0094531

Asadi-Aghbolaghi, M., et al. (2017). A survey on deep learning based approaches for action and gesture recognition in image sequences. *International Conference on Automatic Face and Gesture Recognition, IEEE*, 476–483.

Assaf, Y., and Pasternak, O. (2008). Diffusion Tensor Imaging (DTI)-based White Matter Mapping in brain research: a review. *Journal of molecular neuroscience: MN, 34*(1), 51–61. https://doi.org/10.1007/s12031-007-0029-0

Bagher-Ebadian, H., et al. (2011). Predicting final extent of ischemic infarction using artificial neural network analysis of multi-parametric MRI in patients with stroke. *PLoS One, 6*(8), e22626. doi: https://doi.org/10.1371/journal.pone.0022626.

Baliyan, V., et al. (2016). Diffusion weighted imaging: Technique and applications. *World J. Radiol, 8*(9), 785-798. doi: 10.4329/wjr.v8.i9.785.

Bammer, R., et al. (2003). Analysis and generalized correction of the effect of spatial gradient field distortions in diffusion-weighted imaging. *Magn. Reson. Med., 50*, 560–569. doi: 10.1002/mrm.10545

Barbieri, S., et al. (2020). Deep learning how to fit an intravoxel incoherent motion model to diffusion-weighted mri. *Magnetic Resonance in Medicine, 83*(1), 312–321. doi: https:// doi.org/10.1002/mrm.27910.

Baselice, F., et al. (2017). Bayesian MRI denoising in complex domain. *Magn. Reson. Imaging., 38*, 112-122.

Basser, P., and Jones, D. (2002). Diffusion-tensor MRI: theory, experimental design and data analysis - a technical review. *NMR in biomedicine, 15*(7-8), 456–467. https://doi.org/10.1002/nbm.783

Bastiani, M., et al. (2019). Automated Quality Control for Within and Between Studies Diffusion MRI Data Using A Non-Parametric Framework for Movement and Distortion Correction. *Neuroimage, 184*, 801–812. doi: 10.1016/j.neuroimage.2018.09.073

Baur, C., et al. (2019). Deep Autoencoding Models for Unsupervised Anomaly Segmentation in Brain MR Images. *BrainLes 2018*, 11383, 161–169. doi: https://doi.org/10.1007/978-3-030-11723-8_16.

Bellon, E., et al. (1986). MR artifacts: a review. *AJR: Am. J. Roentgenol., 147*(6), 1271–1281. https://doi.org/10.2214/ajr.147.6.1271

Bengs, M., et al. (2019). 4d spatio-temporal deep learning with 4d fmri data for autism spectrum disorder classification. *International Conference on Medical Imaging with Deep Learning*. arXiv:2004.10165. doi:

Bernal, J., et al. (2019). Deep convolutional neural networks for brain image analysis on magnetic resonance imaging: a review. *Artif. Intell. Med., 95*, 64-81. doi: https://doi.org/10.1016/j.artmed.2018.08.008.

Bernstein, M., et al. (2006). Imaging artifacts at 3.0T. *J. Magn. Reson. Imaging, 24*(4), 735–746. https://doi.org/10.1002/jmri.20698

Callaghan, P. (1991). Principles of nuclear magnetic resonance microscopy. *Oxford University Press*.

Chandola, V., et al. (2009). Anomaly Detection: A Survey. *ACM Computing Surveys, 41*(3), 1-58. doi: 10.1145/1541880.1541882.

Cheng, H., et al. (2021a). Modular interactive video object segmentation: interaction-to-mask, propagation and difference-aware fusion. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5559–5568.

Cheng, H., et al. (2021b). Rethinking space-time networks with improved memory coverage for efficient video object segmentation. *Adv. Neural Inf. Process. Syst., 34*, 11781–11794.

Cheng, J., et al. (2012). Nonrigid motion correction in 3D using autofocusing with localized linear translations. *Magn. Reson. Med., 68*(6), 1785-97. doi: 10.1002/mrm.24189.

Cho, G., et al. (2017). Intravoxel incoherent motion (IVIM) histogram biomarkers for prediction of neoadjuvant treatment response in breast cancer patients. *Eur. J. Radiol. Open, 4*, 101-107.

Choi, S., et al. (2011). DTI at 7 and 3 T: systematic comparison of SNR and its influence on quantitative metrics. *Magn. Reson. Imaging, 29*, 739–751.

Choy, C., et al. (2019). 4d spatio-temporal convnets: Minkowski convolutional neural networks. *Conference on Computer Vision and Pattern Recognition*, 3075–3084.

Ciritses, A., et al. (2018). Automated pixel-wise brain tissue segmentation of diffusion-weighted images via machine learning. *NMR Biomed., 31*(7), e3931. doi: 10.1002/nbm.3931.

Clark, D., and Badea, C. (2019). Convolutional regularization methods for 4d, x-ray ct reconstruction. *Medical Imaging 2019: Physics of Medical Imaging, International Society for Optics and Photonics*, 109482A. doi: https://doi.org/10.1117/12.2512816

Detre, J., et al. (1992). Perfusion imaging. *Magnetic Resonance in Medicine, 23*(1), 37-45.

Donahue, J., et al. (2015). Long-term recurrent convolutional networks for visual recognition and description. *Conference on Computer Vision and Pattern Recognition*, 2625–2634.

Duffy, B., et al. (2021). Retrospective motion artifact correction of structural MRI images using deep learning improves the quality of cortical surface reconstructions. *NeuroImage, 230*. doi: https://doi.org/10.1016/j.neuroimage.2021.117756.

Dyvorne, H., et al. (2013). Diffusion-weighted imaging of the liver with multiple b values: effect of diffusion gradient polarity and breathing acquisition on image quality and intravoxel incoherent motion parameters–a pilot study. *Radiology, 266*, 920-929. doi: https://doi.org/10.1148/radiol.12120686.

El Sallab, A., et al. (2018). Yolo4d: A spatio-temporal approach for real-time multi-object detection and classification from lidar point clouds. *Conference on Neural Information Processing Systems*.

Ettehadi N, et al. (2022) Automated Multiclass Artifact Detection in Diffusion MRI Volumes via 3D Residual Squeeze-and-Excitation Convolutional Neural Networks. *Front. Hum. Neurosci., 16*, 877326. doi: 10.3389/fnhum.2022.877326

Ettehadi, N., et al. (2021). Automatic volumetric quality assessment of diffusion MR images via convolutional neural network classifiers. *Proceedings of the 2021 43rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), IEEE*, 2756–2760. 10.1109/EMBC46164.2021.9630834.

Fantini, I., et al. (2018). Automatic detection of motion artifacts on MRI using Deep CNN. *Proceedings of the 2018 International Workshop on Pattern Recognition in Neuroimaging (PRNI), IEEE*, 1–4.

Farrell, J., et al. (2007). Effects of signal-to-noise ratio on the accuracy and reproducibility of diffusion tensor imaging-derived fractional anisotropy, mean diffusivity, and principal eigenvector measurements at 1.5 T. *J. Magn. Reson. Imaging, 26*, 756–767.

Feichtenhofer, C., et al. (2016). Convolutional two-stream network fusion for video action recognition. *Conference on Computer Vision and Pattern Recognition*, 1933–1941.

Funke, I., et al. (2019). Using 3d convolutional neural networks to learn spatiotemporal features for automatic surgical gesture recognition in video. *International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer*, 467–475.

Gao, C., et al. (2018). Learning to see forces: Surgical force prediction with rgb-point cloud temporal convolutional networks. *OR 2.0 Context-Aware Operating Theaters, Computer Assisted Robotic Endoscopy, Clinical Image-Based Procedures, and Skin Image Analysis*, 118–127.

Garcia-Garcia, A., et al. (2017). A Review on Deep Learning Techniques Applied to Semantic Segmentation. *TPAMI*. doi: https://doi.org/10.48550/arXiv.1704.06857.

Gessert, N., et al. (2020). Deep learning with 4D spatio-temporal data representations for OCT-based force estimation. *Medical Image Analysis*. doi: https://doi.org/10.1016/j.media.2020.101730.

Golan, I., et al. (2018). Deep anomaly detection using geometric transformations. *Adv. Neural Inf. Process. Syst., 31*.

Golkov, V., et al. (2016). Q-space deep learning: twelve-fold shorter and model-free diffusion MRI scans. *IEEE Trans. Med. Imaging, 35*(5), 1344-1351. doi: 10.1109/TMI.2016.2551324.

Gong, D., et al. (2019). Memorizing normality to detect anomaly: memory-augmented deep autoencoder for unsupervised anomaly detection. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1705–1714.

Graham, M., et al. (2018). A supervised learning approach for diffusion MRI quality control with minimal training data. *NeuroImage, 178*, 668-676. doi: https://doi.org/10.1016/j.neuroimage.2018.05.077.

Griswold, M., et al. (2002). Generalized autocalibrating partially parallel acquisitions (GRAPPA). *Magn. Reson. Med., 47*(6), 1202–1210. doi: https://doi.org/10.1002/mrm.10171

Gudbjartsson, H., et al. (1995). The rician distribution of noisy mri data. *Magn. Reson. Med., 34*(6), 910-914. doi: https://doi.org/10.1002/mrm.1910340618.

Gudovskiy, D., et al. (2022). CFLOW-AD: Real-Time Unsupervised Anomaly Detection With Localization via Conditional Normalizing Flows. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 98-107.

Guo, S., et al. (2019). Visual Anomaly Detection in Event Sequence Data. *2019 IEEE International Conference on Big Data (Big Data)*, 1125-1130. doi: 10.1109/BigData47090.2019.9005687.

Haddad, S., et al. (2019). Comparison of quality control methods for automated diffusion tensor imaging analysis pipelines. *PLoS One, 14*, e0226715. 10.1371/journal.pone.0226715.

Haselgrove, J., et al. (1996). Correction for distortion of echo-planar images used to calculate the apparent diffusion coefficient. *Magn. Reson. Med., 36*, 960-964. doi: 10.1002/mrm.1910360620.

Hastie, T., et al. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. *Springer Science and Business Media*. doi: https://doi.org/10.1007/978-0-387-21606-5.

Havaei, M., et al. (2017). Brain tumor segmentation with deep neural networks. *Medical image analysis, 35*, 18–31.

He, K., et al, (2016b). Identity mappings in deep residual networks. *European Conference on Computer Vision*, 630–645.

Heiland, S. (2008). From A as in Aliasing to Z as in Zipper: Artifacts in MRI. *Clinical neuroradiology, 18*(1), 25-36.

Heo, Y., et al. (2021). Guided interactive video object segmentation using reliability-based attention maps. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7322–7330.

Hochreiter, S., and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*, 1735–1780.

Ibtehaz, N., et al. (2020). MultiResUNet: Rethinking the U-Net Architecture for Multimodal Biomedical Image Segmentation. *Computer Vision and Pattern Recognition, 121*, 74-87. doi: https://doi.org/10.1016/j.neunet.2019.08.025.

Iglesias, J., et al. (2017). Retrospective head motion estimation in structural brain MRI with 3D CNNs. *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*, 314–322. 10.1007/978-3-319-66185-8_36.

Isensee, F., et al. (2018). nnU-net: Self-adapting framework for u-net-based medical image segmentation. *arXiv preprint*, arXiv:1809.10486.

Jenkinson, M., et al. (2001). A global optimisation method for robust affine registration of brain images. *Med. Image Anal., 5*, 143-156.

Jezzard, P., et al. (1995). Correction for geometric distortion in echo planar images from B0 field variations. *Magn Reson Med., 34*(1), 65-73. doi: 10.1002/mrm.1910340111.

Ji, S., et al. (2013). 3D convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 35*, 221–231.

Jiang, H., et al. (2006). DtiStudio: resource program for diffusion tensor computation and fiber bundle tracking. *Comput. Methods Programs Biomed., 81*, 106–116.

Jin, Y., et al. (2019). Multi-task recurrent convolutional network with correlation loss for surgical video analysis. *Medical image analysis*, 101572.

Jones, D., et al. (2004a). Squashing peanuts and smashing pumpkins: How noise distorts diffusion-weighted MR data. *Magn. Reson. Med., 52*(5), 979-993. doi: https://doi.org/10.1002/mrm.20283.

Jones, D., et al. (2004b). The effect of gradient sampling schemes on measures derived from diffusion tensor MRI: A Monte Carlo study. *NeuroImage*. doi: 10.1002/mrm.20033.

Kaandorp, M., et al. (2021). Improved unsupervised physics-informed deep learning for intravoxel incoherent motion modeling and evaluation in pancreatic cancer patients. *Magn. Reson. Med., 86*, 2250–2265. https://doi.org/10.1002/mrm.28852

Kamnitsas, K., et al. (2017). Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation. *Medical image analysis, 36*, 61–78.

Kelly, C., et al. (2017). Transfer learning and convolutional neural net fusion for motion artefact detection. *Proceedings of the Annual Meeting of the International Society for Magnetic Resonance in Medicine, 3523*.

Kingma, D., and Welling, M. (2013). Auto-Encoding Variational Bayes. *arXiv preprint*, arXiv:1312.6114. doi: https://doi.org/10.48550/arXiv.1312.6114.

Kiran, B., et al. (2018). An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. *J. Imaging, 4*(2), 36.

Klaassen, R., et al. (2020). Pathological validation and prognostic potential of quantitative MRI in the characterization of pancreas cancer: preliminary experience. *Mol. Oncol., 14*, 2176-2189.

Krizhevsky, A., et al. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 1097-1105. doi: https://doi.org/10.1145/3065386.

Krupa, K., et al. (2015). Artifacts in magnetic resonance imaging. *Pol. J. Radiol., 80*, 93-106. doi: 10.12659/PJR.892628.

Küstner, T., et al. (2020). CINENet: deep learning-based 3D cardiac CINE MRI reconstruction with multi-coil complex-valued 4D spatio-temporal convolutions. *Sci. Rep., 10*(1), 13710–13710. doi: 10.1038/s41598-020-70551-8.

Le Bihan, D., and Breton, E. (1985). Imagerie de diffusion in vivo par résonance magnétique nucléaire. *C. R. Acad. Sci., 301*, 1109–1112.

Le Bihan, D. (2003). Looking into the functional architecture of the brain with diffusion MRI. *Nature Reviews Neuroscience, 4*, 469–480. doi: https://doi.org/10.1038/nrn1119

Le Bihan, D. (2008). Intravoxel incoherent motion perfusion MR imaging: a wake-up call. *Radiology, 249*(3), 748–752. doi: https://doi.org/10.1148/radiol.2493081301

Le Bihan, D., et al. (1986). MR imaging of intravoxel incoherent motions: application to diffusion and perfusion in neurologic disorders. *Radiology, 161*(2), 401-407.

Le Bihan, D., et al. (1988). Separation of diffusion and perfusion in intravoxel incoherent motion MR imaging. *Radiology, 168*, 497-505.

Le Bihan, D., et al. (1992). The capillary network: a link between IVIM and classical perfusion. *Magnetic Resonance in Medicine, 27*(1), 171-178.

Le Bihan, D., et al. (2001). Diffusion tensor imaging: concepts and applications. *J. Magn. Reson. Imaging, 13*, 534–546.

LeCun, Y., et al. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE, 86*(11), 2278-2324. doi: 10.1109/5.726791.

Lee, Y., et al. (2015). Intravoxel incoherent motion diffusion-weighted MR imaging of the liver: effect of triggering methods on regional variability and measurement repeatability of quantitative parameters. *Radiology, 274*(2), 405-415.

Lemke, A., et al. (2010). An in vivo verification of the intravoxel incoherent motion effect in diffusion-weighted imaging of the abdomen. *Magn. Reson. Med., 64*(6), 1580-1585. doi: https://doi.org/10.1002/mrm.22565.

Li, Y., et al. (2013). Image Corruption Detection in Diffusion Tensor Imaging for Post-Processing and Real-Time Monitoring. *PLOS ONE, 8*(10), e49764. https://doi.org/10.1371/journal.pone.0049764

Liu, B., et al. (2015). Comparison of quality control software tools for diffusion tensor imaging. *Magn. Reson. Imaging, 33*, 276–285. 10.1016/j.mri.2014.10.011.

Liu, Z., et al. (2016). 3D-based deep convolutional neural network for action recognition with depth sequences. *Image and Vision Computing, 55*, 93–100.

Lundervold, A., et al. (2019). An overview of deep learning in medical imaging focusing on MRI. *Zeitschrift für Medizinische Physik, 29*(2), 102-127. doi: 10.1016/j.zemedi.2018.11.002.

Lysaker, M., et al. (2003). Noise removal using fourth-order partial differential equation with applications to medical magnetic resonance images in space and time. *IEEE Trans. Image. Process., 12*, 1579-1590.

Ma, W., et al. (2018). Quantitative parameters of intravoxel incoherent motion diffusion weighted imaging (IVIM-DWI): potential application in predicting pathological grades of pancreatic ductal adenocarcinoma. *Quant. Imaging Med. Surg., 8*, 301-310.

McVeigh, E., et al. (1985). Noise and filtration in magnetic resonance imaging. *Med. Phys., 12*, 586-591.

Merboldt, K., et al. (1985). Self-diffusion NMR imaging using stimulated echoes. *J. Magn. Reson., 64*, 479–486.

Mesri, H., et al. (2020). The adverse effect of gradient nonlinearities on diffusion MRI: From voxels to group studies. *Neuroimage, 205*, 116127. doi: 10.1016/j.neuroimage.2019.116127

Mirowitz, S. (1999). MR imaging artifacts. Challenges and solutions. *Magn. Reson. Imaging Clin. N. Am., 7*(4), 717–732.

Moratal, D., et al. (2008). k-Space tutorial: an MRI educational tool for a better understanding of kspace. *Biomedical imaging and intervention journal, 4*(1).

Morelli, J., et al. (2011). An image-based approach to understanding the physics of MR artifacts. *Radiographics, 31*(3), 849–866. https://doi.org/10.1148/rg.313105115

Mori, S., and Zhang, J. (2006). Principles of diffusion tensor imaging and its applications to basic neuroscience research. *Neuron, 51*, 527–539.

Morrill, J., et al. (2021). Neural Controlled Differential Equations for Online Prediction Tasks. *ArXiv, abs/2106.11028*. doi:10.48550/arXiv.2106.11028.

Myronenko, A., et al. (2020). 4d cnn for semantic segmentation of cardiac volumetric sequences. *Statistical Atlases and Computational Models of the Heart. Multi-Sequence CMR Segmentation, CRT-EPiggy and LV Full Quantification Challenges, Springer International Publishing*, 72–80. doi: https://doi.org/10.48550/arXiv.1906.07295

Nie, D., et al. (2018). 3-D fully convolutional networks for multimodal isointense infant brain image segmentation. *IEEE Trans. Cybern., 49*(3), 1123-1136. doi: 10.1109/TCYB.2018.2797905.

Oguz, I., et al. (2014). DTIPrep: quality control of diffusion-weighted images. *Front. neuroinformatics, 8*.

Oh, S., et al. (2019). Video object segmentation using space-time memory networks. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 9226–9235.

Ordóñez, F., and Roggen, D. (2016). Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors, 16*, 115.

Parker, C., et al. (2023). Rician likelihood loss for quantitative MRI using self-supervised deep learning. *arXiv, 2307.07072*. doi: https://doi.org/10.48550/arXiv.2307.07072

Pereira, S., et al. (2016). Brain tumor segmentation using convolutional neural networks in mri images. *IEEE transactions on medical imaging, 35*(5), 1240–1251.

Phophalia, A., et al. (2017). 3d MR image denoising using rough set and kernel PCA method. *Magn. Reson. Imaging, 36*, 135-145.

Pierpaoli, C., et al. (2010). TORTOISE: an integrated software package for processing of diffusion MRI data. *Proceedings of the ISMRM 18th annual meeting, 1597*.

Pigou, L., et al. (2018). Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. *International Journal of Computer Vision, 126*, 430–439.

Polders, D., et al. (2011). Signal to noise ratio and uncertainty in diffusion tensor imaging at 1.5, 3.0, and 7.0 Tesla. *J. Magn. Reson. Imaging, 33*, 1456–1463.

Qiu, Z., et al. (2017). Learning spatio-temporal representation with pseudo-3D residual networks. *International Conference on Computer Vision*, 5534–5542.

Ratai, E., et al. (2016). Chapter 5 - Clinical magnetic resonance spectroscopy of the central nervous system. *Handbook of Clinical Neurology, 135*, 93-116. doi: https://doi.org/10.1016/B978-0-444-53485-9.00005-2.

Reese, T., et al. (2003). Reduction of eddy-current-induced distortion in diffusion MRI using a twice-refocused spin echo. *Magn. Reson. Med., 49*(1), 177-182.

Reuter, M., et al. (2015). Head motion during MRI acquisition reduces gray matter volume and thickness estimates. *Neuroimage, 107*, 107–115. doi: 10.1016/j.neuroimage.2014.12.006

Rezende, D., et al. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models. *Proceedings of the 31st International Conference on Machine Learning, PMLR 32*(2), 1278-1286. doi: https://doi.org/10.48550/arXiv.1401.4082.

Rizwan-i-Haque, I., et al. (2020). Deep learning approaches to biomedical image segmentation. *Informatics in Medicine Unlocked., 18*, 100297. doi: 10.1016/j.imu.2020.100297.

Roalf, D., et al. (2016). The impact of quality assurance assessment on diffusion tensor imaging outcomes in a large-scale population-based cohort. *Neuroimage, 125*, 903–919. 10.1016/j.neuroimage.2015.10.068.

Ronneberger, O., et al. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *MICCAI 2015*, 234–241. doi: 10.1007/978-3-319-24574-4_28.

Sakurada, M., et al. (2014). Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction. *MLSDA'14, Association for Computing Machinery*, 4–11. doi: https://doi.org/10.1145/2689746.2689747

Salimi-Khorshidi, G., et al. (2014). Automatic denoising of functional MRI data: combining independent component analysis and hierarchical fusion of classifiers. *Neuroimage, 90*, 449-468.

Samani, Z., et al. (2020). QC-Automator: deep learning-based automated quality control for diffusion mr images. *Front. Neurosci., 13*, 1456. doi: 10.3389/fnins.2019.01456

Schenck, J. (1996). The role of magnetic susceptibility in magnetic resonance imaging: MRI magnetic compatibility of the first and second kinds. *Medical physics, 23*(6), 815-850. doi: 10.1118/1.597854.

Schlegl, T., et al. (2017). Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery. *International conference on information processing in medical imaging*, 146-157. Cham: Springer International Publishing.

Schnell, S., et al. (2009). Fully automated classification of HARDI in vivo data using a support vector machine. *NeuroImage, 46*(3), 642-651. doi: https://doi.org/10.1016/j.neuroimage.2009.03.003.

Shen, H., et al. (2017). Boundary-aware fully convolutional network for brain tumor segmentation. *International Conference on Medical Image Computing and Computer-Assisted Intervention*, 433–441. Springer.

Sijbers, J., et al. (1998). Maximum-likelihood estimation of Rician distribution parameters. *Medical Imaging, IEEE Transactions, 17*, 357-361. doi: 10.1109/42.712125.

Simmons, A., et al. (1994). Sources of intensity nonuniformity in spin echo images at 1.5 T. *Magnetic resonance in medicine, 32*(1), 121-128.

Simonyan, K., and Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. *Advances in Neural Information Processing Systems*, 568–576.

Smith, R., et al. (1991). Chemical shift artifact: dependence on shape and orientation of the lipid-water interface. *Radiology, 181*(1), 225-9. doi: 10.1148/radiology.181.1.1887036.

Smith, T., et al. (2010). MRI artifacts and correction strategies. *Imaging Med., 2*(4), 445–457. doi: 10.2217/IIM.10.33.

Soares, J.M., et al. (2013) A Hitchhiker's Guide to Diffusion Tensor Imaging. *Frontiers in Neuroscience, 7*. doi: https://doi.org/10.3389/fnins.2013.00031

Stadler, A., et al. (2007). Artifacts in body MR imaging: their appearance and how to eliminate them. *Eur. Radiol., 17*(5), 1242–1255. https://doi.org/10.1007/s00330-006-0470-4

Sun, L., et al. (2015). Human action recognition using factorized spatio-temporal convolutional networks. *International Conference on Computer Vision*, 4597–4605.

Sun, P., et al. (2019). Tissue segmentation using sparse non-negative matrix factorization of spherical mean diffusion MRI data. *MICCAI 2019, Mathematics and Visualization, Springer*, 69-76. doi: https://doi.org/10.1007/978-3-030-05831-9_6.

Sundgren, P., et al. (2004). Diffusion tensor imaging of the brain: review of clinical applications. *Neuroradiology, 46*, 339–350.

Taylor, D., and Bushell, M. (1985). The spatial mapping of translational diffusion coefficients by the NMR imaging technique. *Phys. Med. Biol., 30*, 345–349.

Tofts P., et al. (1999). Estimating kinetic parameters from dynamic contrast-enhanced T(1)-weighted MRI of a diffusable tracer: standardized quantities and symbols. *J. Magn. Reson. Imaging, 10*, 223–232.

Tokuda, J., et al. (2008). Adaptive 4D MR imaging using navigator-based respiratory signal for MRI-guided therapy. *Magn. Reson. Med., 59*, 1051–1061.

Tournier, J., et al. (2019). MRtrix3: a fast, flexible and open software framework for medical image processing and visualisation. *Neuroimage, 202*, 116137. doi: 10.1016/j.neuroimage.2019.116137

Tran, D., et al. (2015). Learning spatiotemporal features with 3D convolutional networks. *International Conference on Computer Vision*, 4489–4497.

Tran, D., et al. (2018). A closer look at spatiotemporal convolutions for action recognition. *Conference on Computer Vision and Pattern Recognition*, 6450–6459.

Van Dijk, K., et al. (2012). The influence of head motion on intrinsic functional connectivity MRI. *Neuroimage, 59*, 431–438. doi: 10.1016/j.neuroimage.2011.07.044

Varol, G., et al. (2018). Long-term temporal convolutions for action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 40*, 1510–1517.

Veraart, J., et al. (2016). Denoising of diffusion MRI using random matrix theory. *NeuroImage, 142*, 394-406. doi: 10.1016/j.neuroimage.2016.08.016.

Victoroff, J., et al. (1994). A method to improve interrater reliability of visual inspection of brain MRI scans in dementia. *Neurology, 44*, 2267–2267. doi: 10.1212/wnl.44.12.2267

Vincent, P., et al. (2010). Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *Journal of Machine Learning Research, 11*, 3371-3408.

Wang, L., et al. (2016). Temporal segment networks: Towards good practices for deep action recognition. *European Conference on Computer Vision*, 20–36.

Weaver, J., et al. (1992). Wavelet-encoded MR imaging. *Magnetic resonance in medicine, 24*(2), 275–287. doi: https://doi.org/10.1002/mrm.1910240209

Wood, M., et al. (1985). MR image artifacts from periodic motion. *Med. Phys., 12*(2), 143-51. doi: 10.1118/1.595782.

Xingjian, S., et al. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Advances in Neural Information Processing Systems*, 802–810.

Yue-Hei Ng, J., et al. (2015). Beyond short snippets: Deep networks for video classification. *Conference on Computer Vision and Pattern Recognition*, 4694–4702.

Zaitsev, M., et al. (2004). Point spread function mapping with parallel imaging techniques and high acceleration factors: Fast, robust, and flexible method for echo-planar imaging distortion correction. *Magnetic resonance in medicine, 52*, 1156-66. doi: 10.1002/mrm.20261.

Zhang, W., et al. (2015). Deep convolutional neural networks for multi-modality isointense infant brain image segmentation. *NeuroImage, 108* 214-224. doi: https://doi.org/10.1016/j.neuroimage.2014.12.061.

Zhang, W., et al. (2022). A Multi-task Network with Weight Decay Skip Connection Training for Anomaly Detection in Retinal Fundus Images. *MICCAI 2022*, 13432, 656–666. doi: https://doi.org/10.1007/978-3-031-16434-7_63.

Zhang, X., et al. (2015). Denoising of 3D magnetic resonance images by using higher-order singular value decomposition. *Med. Image. Anal., 19*, 75-86.

Zhao, X., et al. (2018). A deep learning model integrating fcnns and crfs for brain tumor segmentation. *Medical image analysis, 43*, 98–111.

Zhao, Y., et al. (2018). Modeling 4d fmri data via spatio-temporal convolutional neural networks (st-cnn). *International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer*, 181–189.

Zhou, C., et al. (2019). One-pass multi-task networks with cross-task guided attention for brain tumor segmentation. *arXiv preprint*, arXiv:1906.01796.

Zhu, L., et al. (2017). Predictive and prognostic value of intravoxel incoherent motion (IVIM) MR imaging in patients with advanced cervical cancers undergoing concurrent chemo-radiotherapy. *Sci. Rep., 7*, 1-9.

Zhuo, J., and Gullapalli, R. (2006). AAPM/RSNA physics tutorial for residents: MR artifacts, safety, and quality control. *Radiographics, 26*(1), 275–297. https://doi.org/10.1148/rg.261055134

Zimmerer, D., et al. (2019). Unsupervised Anomaly Localization Using Variational Auto-Encoders. *MICCAI 2019*, 11767, 289–297. doi: 10.1007/978-3-030-32251-9_32.

van de Leemput, S., et al. (2020). Stacked Bidirectional Convolutional LSTMs for Deriving 3D Non-Contrast CT From Spatiotemporal 4D CT. *IEEE transactions on medical imaging, 39*(4), 985–996. doi: https://doi.org/10.1109/TMI.2019.2939044

van de Ville, D., et al. (2007). Wspm: wavelet-based statistical parametric mapping. *Neuroimage, 37*, 1205-1217.

Appendix

A Labeling the data - manual inspection

Table 6: Artifact labels resulting from the visual inspection of the 28 slice-timepoint pairs with the highest mean z-scores for patient 1.

t	Slice	Interleaved Motion Artifact	Motion Artifact	Local Signal Dropout	Local Increased Signal
36	9	x		x	x
	11	x		x	x
	3	x		x	x
	1	x		x	x
	5	x		x	x
	17	x		x	x
	7	x		x	x
25	10	x		x	x
	8	x		x	x
	6	x		x	x
	4	x		x	x
	12	x		x	x
	16	x		x	
39	10		x	x	x
	8		x	x	x
	4		x	x	x
	2		x	x	x
54	9		x	x	
37	10	x		x	x
	12	x		x	x
	8	x		x	x
17	8		x	x	x
	10		x	x	
1	13	x		x	x
	17	x			x
	11	x		x	x
	7	x		x	x
	5	x		x	x

Table 7: Artifact labels resulting from the visual inspection of 34 slice-timepoint pairs with a mean z-score of just above the threshold (minimal outliers) for patient 1.

t	Slice	Interleaved Motion Artifact	Motion Artifact	Local Signal Dropout	Local Increased Signal
45	7	x		x	x
	9	x		x	x
	17	x		x	x
46	0		x	x	
	2		x		
	12		x	x	x
	16		x	x	
41	7		x	x	x
28	1	x		x	x
	3	x		x	x
	5	x		x	x
	7	x		x	x
	13	x		x	x
	15	x		x	x
	17	x		x	
27	0	x		x	x
	2	x		x	x
	4	x		x	x
	6	x		x	x
	8	x		x	x
	12	x		x	x
	14	x		x	
21	13	x		x	x
16	3	x		x	x
	5	x		x	x
	13	x		x	x
1	1		x	x	
	3		x	x	
3	1		x	x	x
	3		x	x	x
	9		x	x	x
7	0	x			x
	2	x			x
	4	x			x

Table 8: Visual inspection label versus model output label per timepoint about whether they contain at least one outlier slice (label 'poor') for patient 1.

t	poor (visual inspection)	poor (predicted by model)
0		
1	x	x
2		
3	x	x
4		
5	x	
6		
7	x	x
8		
9	x	x
10		
11		
12	x	x
13		
14		
15		
16	x	x
17	x	x
18		
19		
20		
21	x	x
22		
23	x	x
24		
25	x	x
26		
27	x	x
28	x	x
29		
30	x	x
31		
32		
33		
34		
35		
36	x	x
37	x	x
38		
39	x	x
40	x	x
41		x
42	x	
43	x	
44		
45	x	x
46	x	x
47	x	
48		
49		
50		
51	x	x
52		
53	x	x
54	x	x
55		
56		

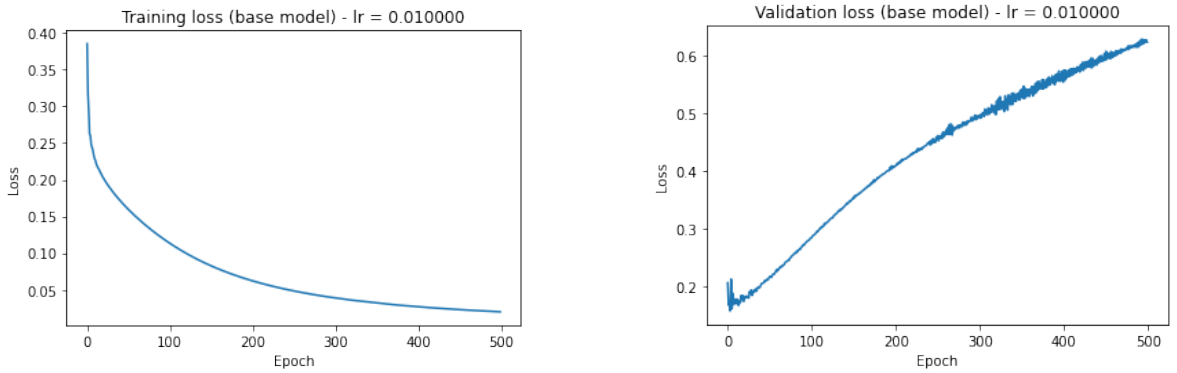
B Training and validation performance of the Base models

The training and validation performance curves and outputs belonging to Section 5.3.1.

B.1 Base classifier

The training and validation performances of the Base classifier.

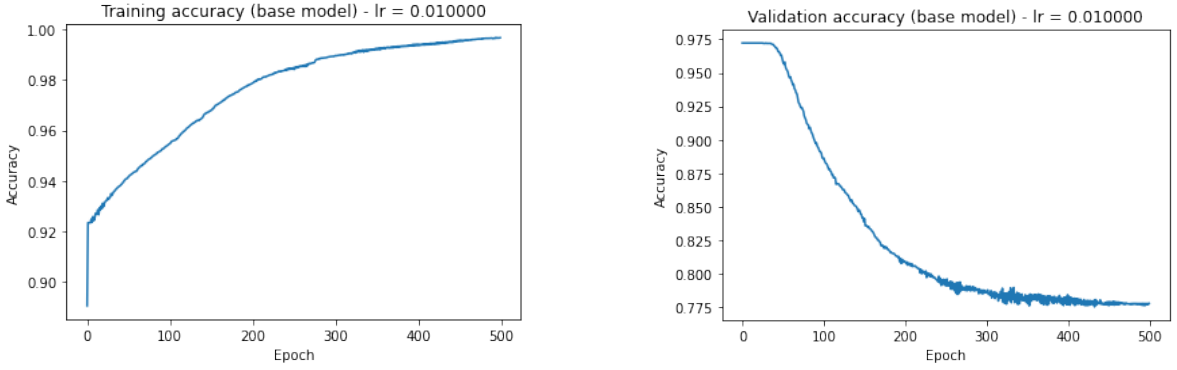
Loss



(a) Training loss (b) Validation loss

Figure 49: Training and validation loss Base classifier (BCE loss)

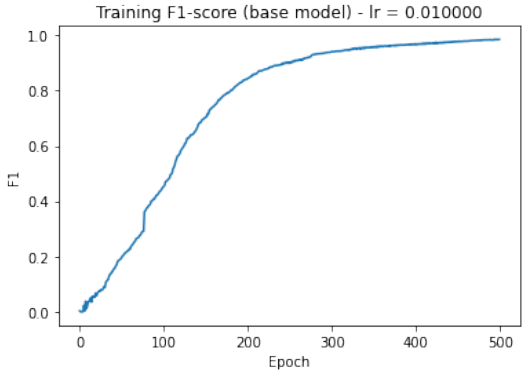
Accuracy



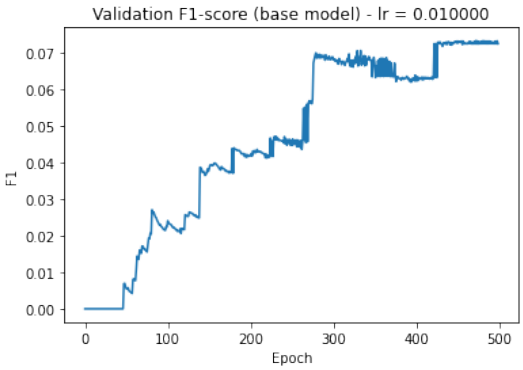
(a) Training accuracy (b) Validation accuracy

Figure 50: Training and validation accuracy Base classifier (BCE loss)

F1-score



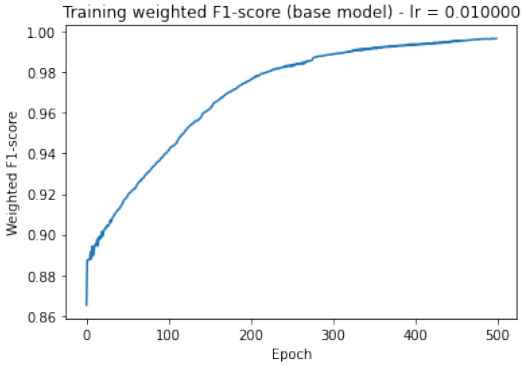
(a) Training F1-score



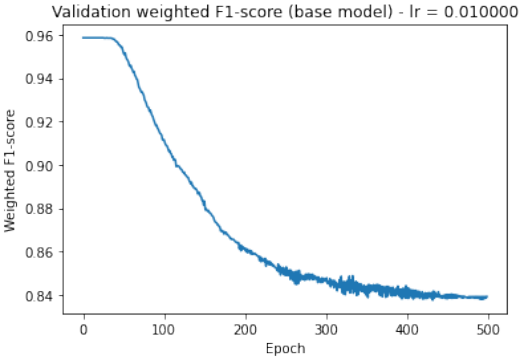
(b) Validation F1-score

Figure 51: Training and validation F1-score Base classifier (BCE loss)

Weighted F1-score



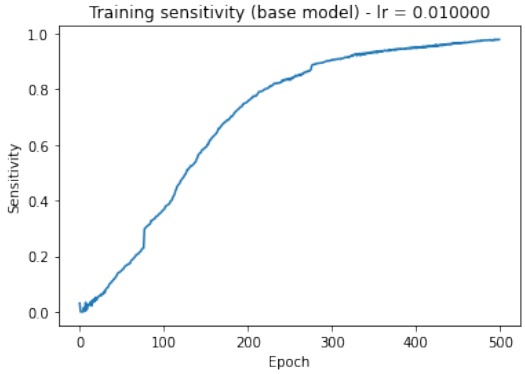
(a) Training weighted F1-score



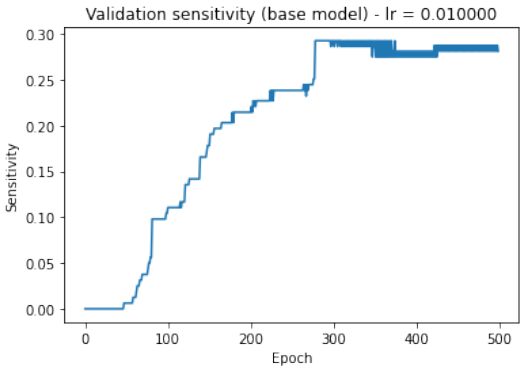
(b) Validation weighted F1-score

Figure 52: Training and validation weighted F1-score Base classifier (BCE loss)

Sensitivity



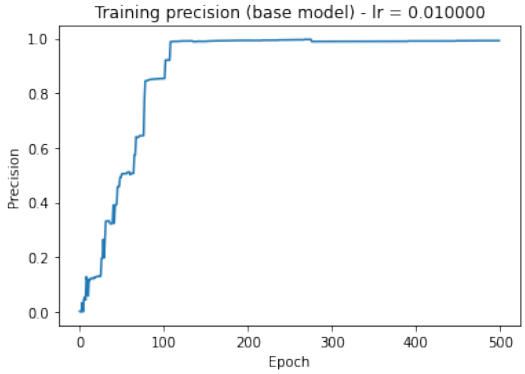
(a) Training sensitivity



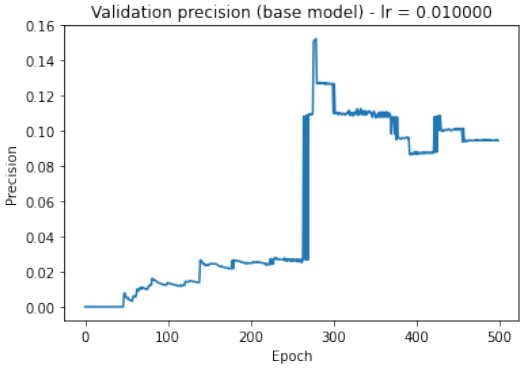
(b) Validation sensitivity

Figure 53: Training and validation sensitivity Base classifier (BCE loss)

Precision



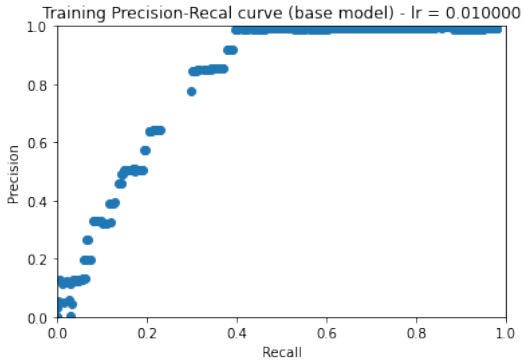
(a) Training precision



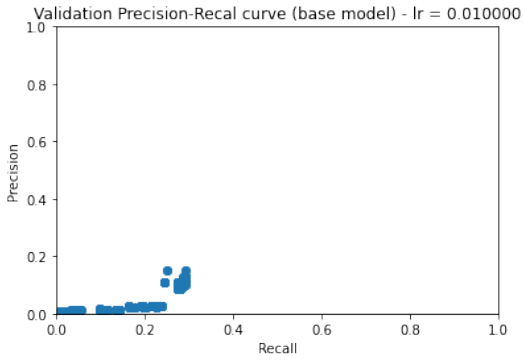
(b) Validation precision

Figure 54: Training and validation precision Base classifier (BCE loss)

Precision vs sensitivity



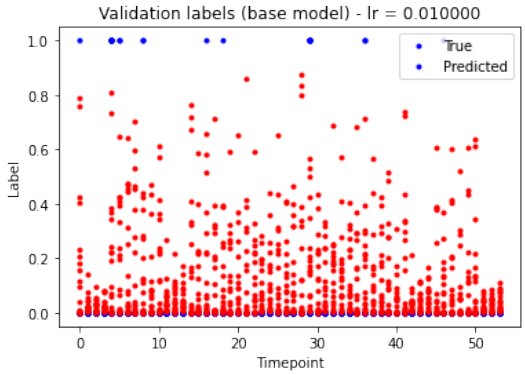
(a) Training precision vs sensitivity



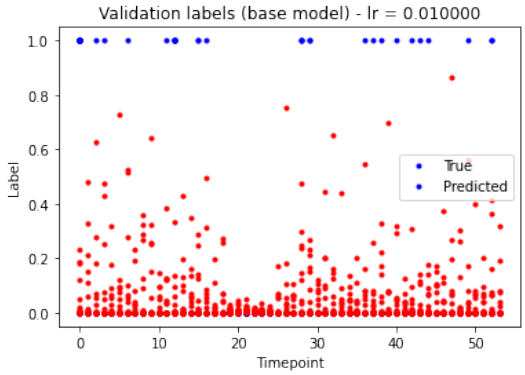
(b) Validation precision vs sensitivity

Figure 55: Training and validation precision vs sensitivity Base classifier (BCE loss)

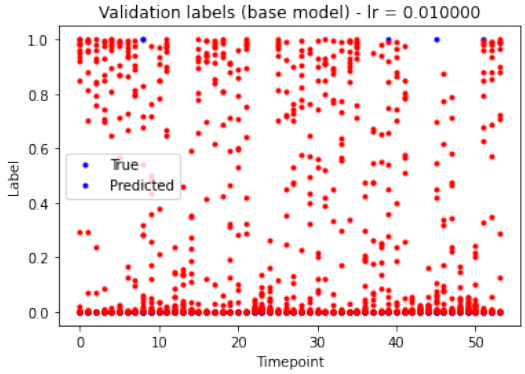
True vs predicted



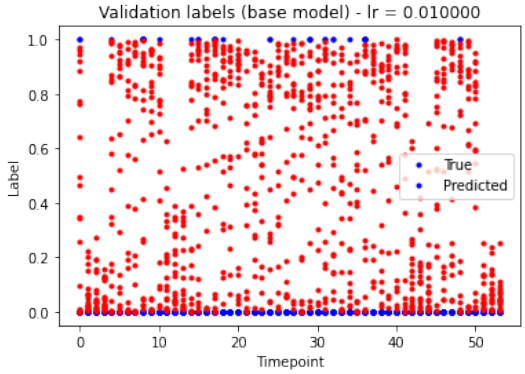
(a) True vs predicted for patient 1



(b) True vs predicted for patient 2



(c) True vs predicted for patient 3



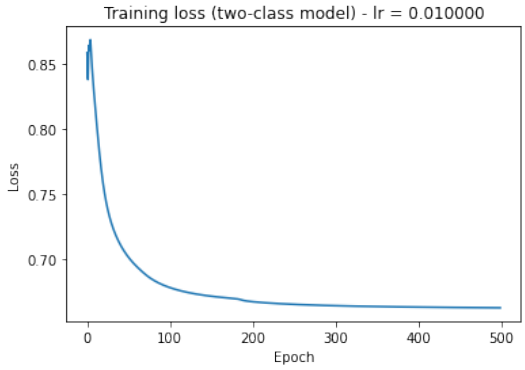
(d) True vs predicted for patient 4

Figure 56: True values vs predicted values for all 4 validation patients Base classifier (BCE loss)

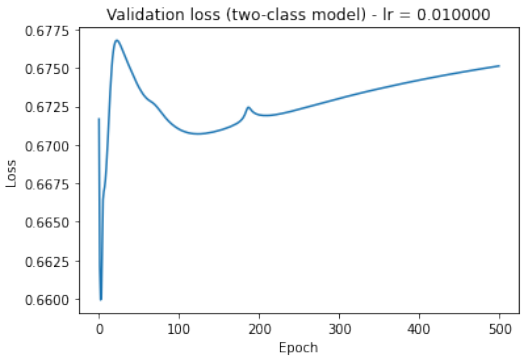
B.2 Two-class model

The training and validation performances of the Two-class model.

Loss



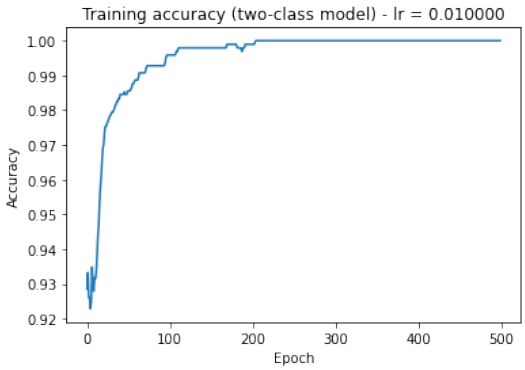
(a) Training loss



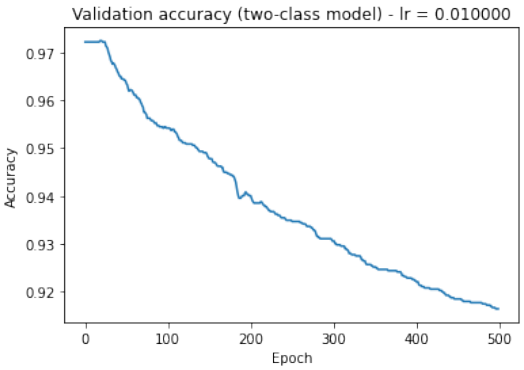
(b) Validation loss

Figure 57: Training and validation loss Two-class model (wBCE loss)

Accuracy



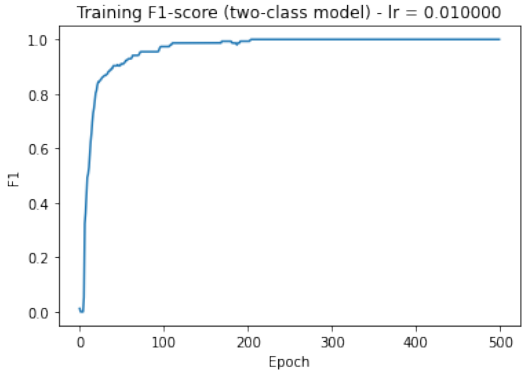
(a) Training accuracy



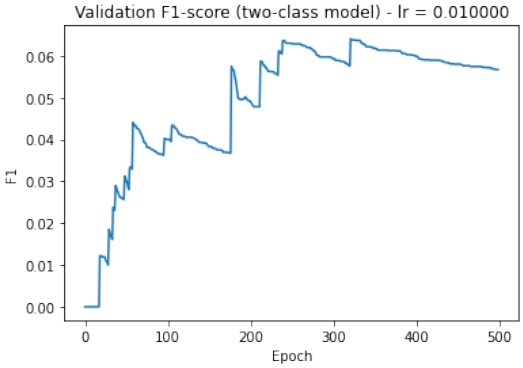
(b) Validation accuracy

Figure 58: Training and validation accuracy Two-class model (wBCE loss)

F1-score



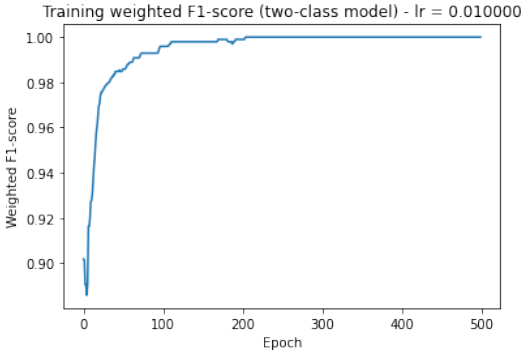
(a) Training F1-score



(b) Validation F1-score

Figure 59: Training and validation F1-score Two-class model (wBCE loss)

Weighted F1-score

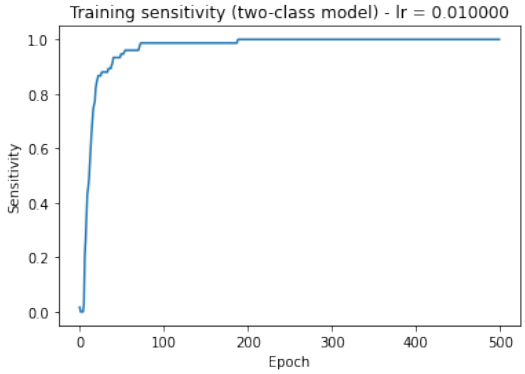


(a) Training weighted F1-score

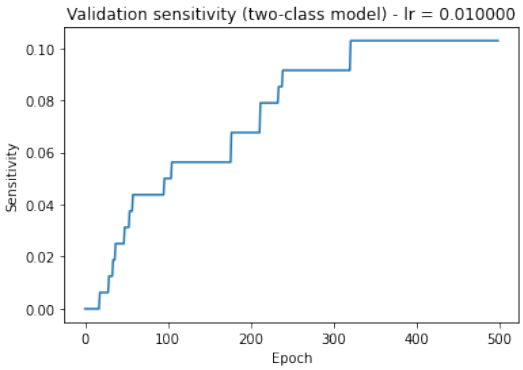
(b) Validation weighted F1-score

Figure 60: Training and validation weighted F1-score Two-class model (wBCE loss)

Sensitivity



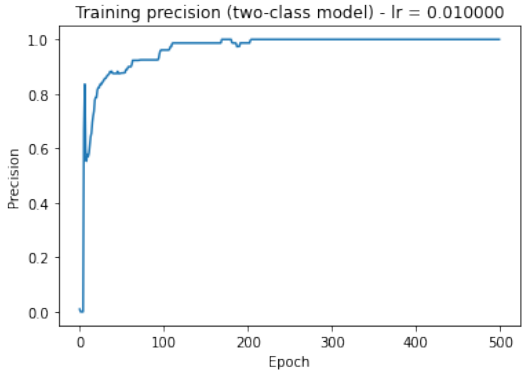
(a) Training sensitivity



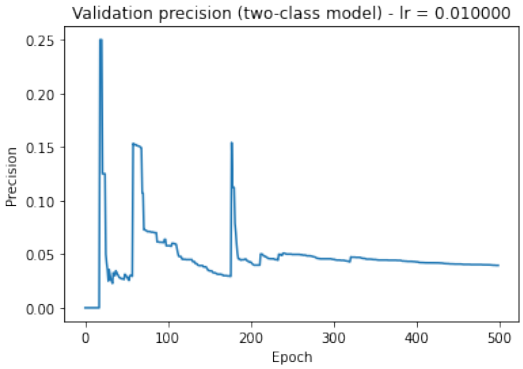
(b) Validation sensitivity

Figure 61: Training and validation sensitivity Two-class model (wBCE loss)

Precision



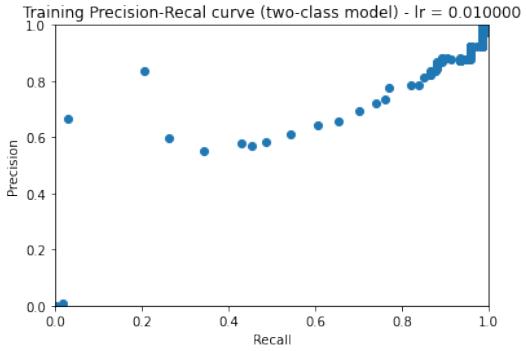
(a) Training precision



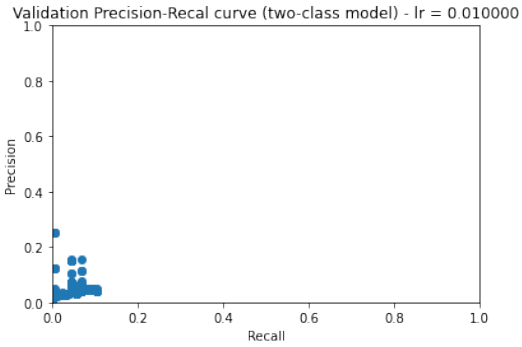
(b) Validation precision

Figure 62: Training and validation precision Two-class model (wBCE loss)

Precision vs sensitivity



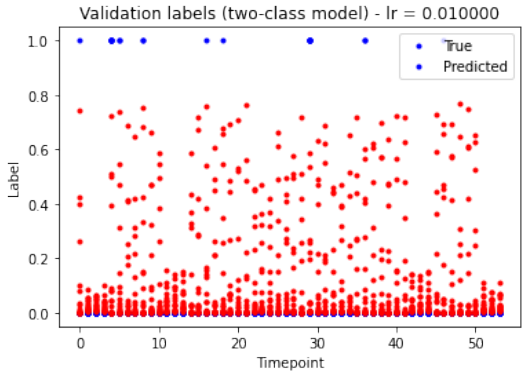
(a) Training precision vs sensitivity



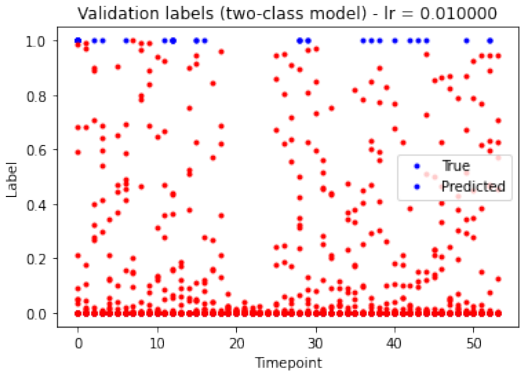
(b) Validation precision vs sensitivity

Figure 63: Training and validation precision vs sensitivity two-class model (wBCE loss)

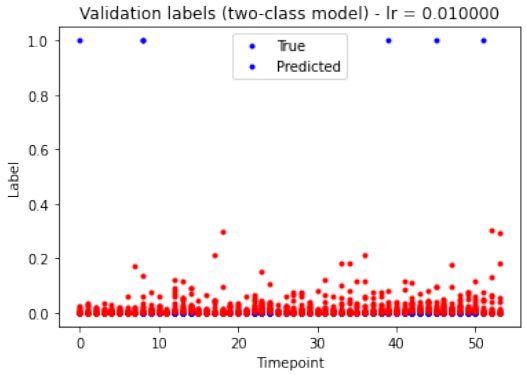
True vs predicted



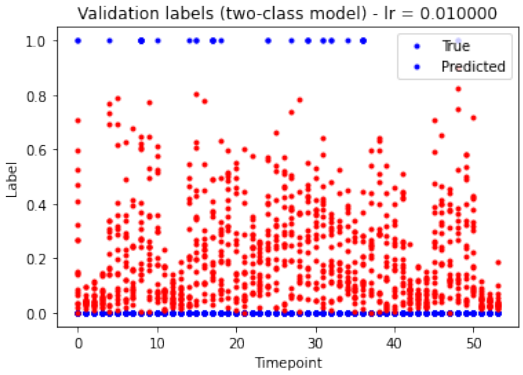
(a) True vs predicted for patient 1



(b) True vs predicted for patient 2



(c) True vs predicted for patient 3



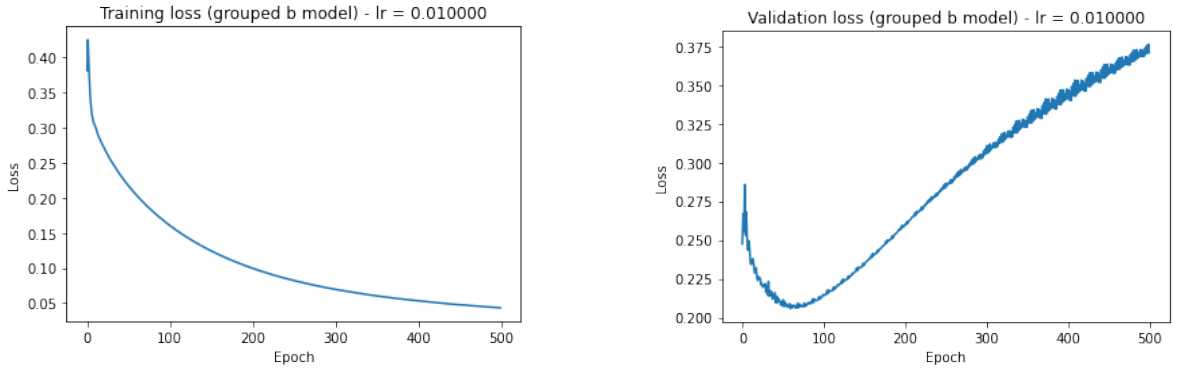
(d) True vs predicted for patient 4

Figure 64: True values vs predicted values for all 4 validation patients two-class model (wBCE loss)

B.3 Grouped b model

The training and validation performances of the Grouped b model.

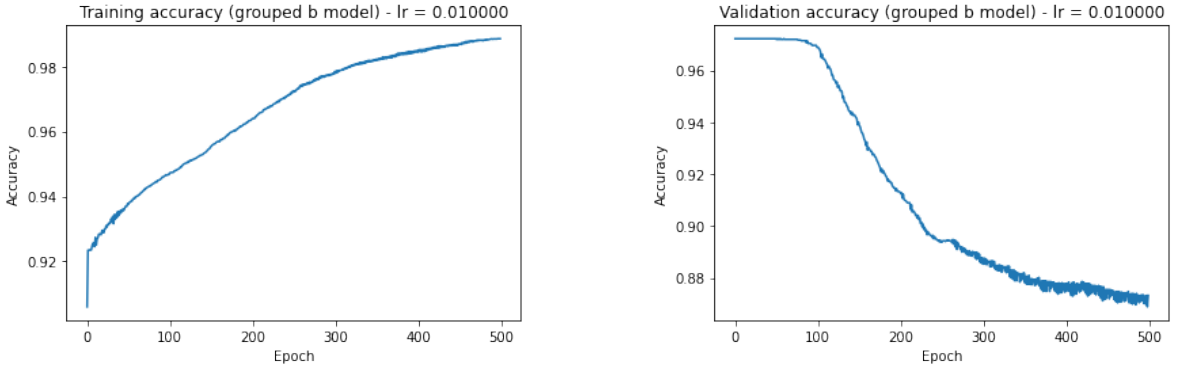
Loss



(a) Training loss (b) Validation loss

Figure 65: Training and validation loss Grouped b model (BCE loss)

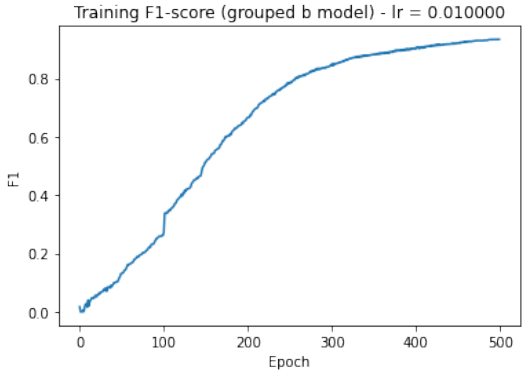
Accuracy



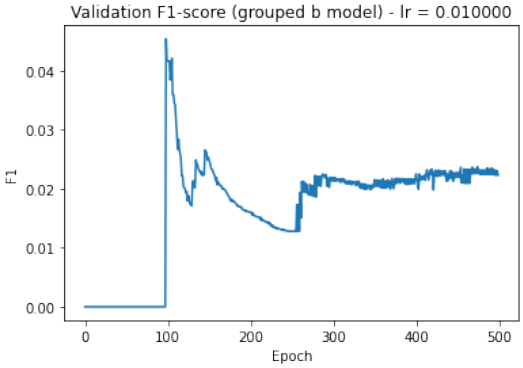
(a) Training accuracy (b) Validation accuracy

Figure 66: Training and validation accuracy Grouped b model (BCE loss)

F1-score



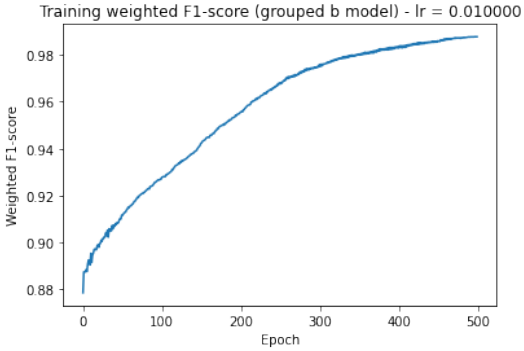
(a) Training F1-score



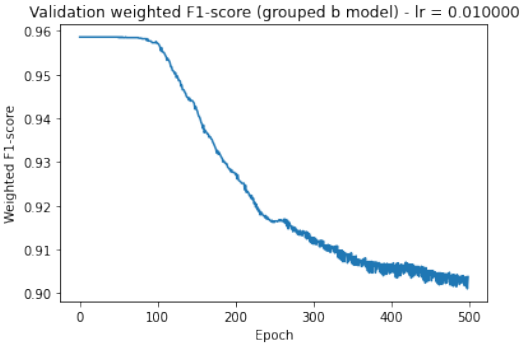
(b) Validation F1-score

Figure 67: Training and validation F1-score Grouped b model (BCE loss)

Weighted F1-score



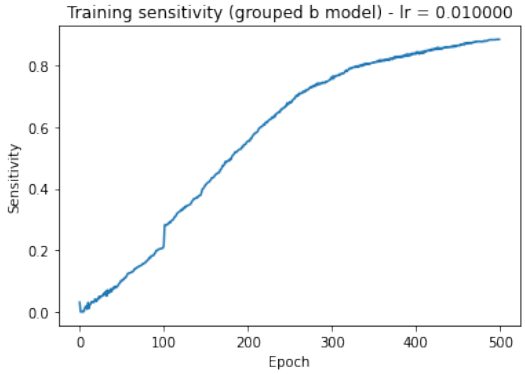
(a) Training weighted F1-score



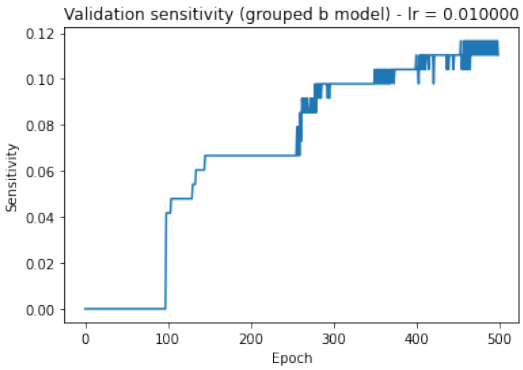
(b) Validation weighted F1-score

Figure 68: Training and validation weighted F1-score Grouped b model (BCE loss)

Sensitivity



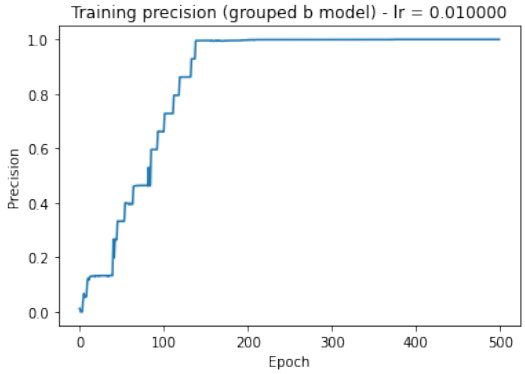
(a) Training sensitivity



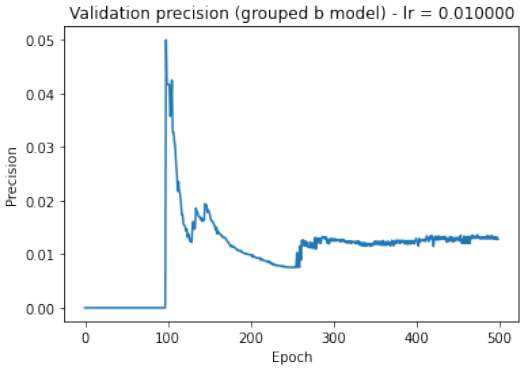
(b) Validation sensitivity

Figure 69: Training and validation sensitivity Grouped b model (BCE loss)

Precision



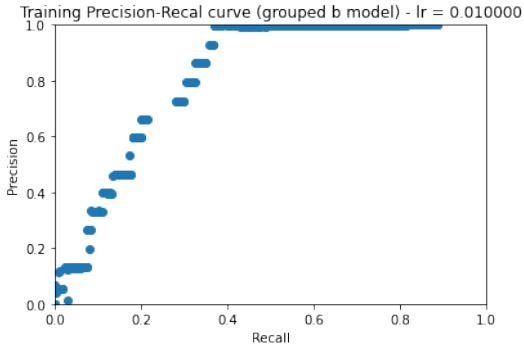
(a) Training precision



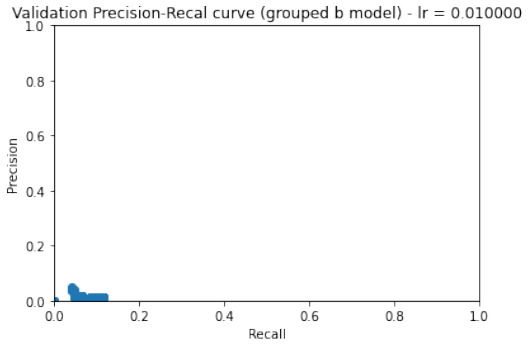
(b) Validation precision

Figure 70: Training and validation precision Grouped b model (BCE loss)

Precision vs sensitivity



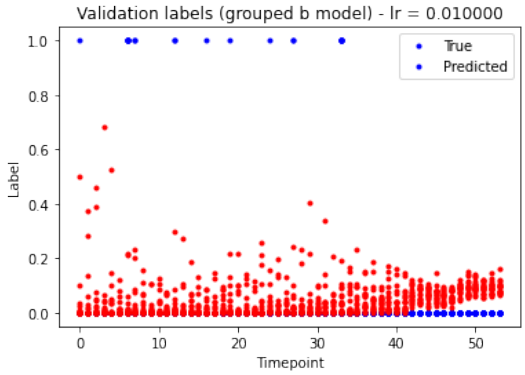
(a) Training precision vs sensitivity



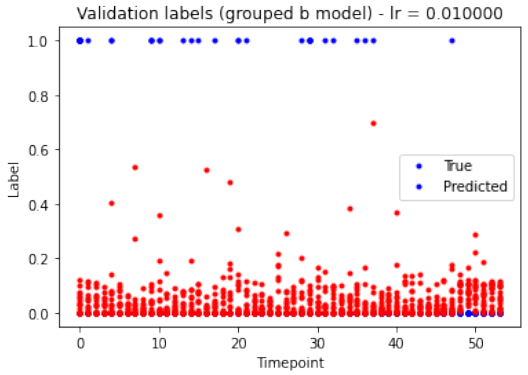
(b) Validation precision vs sensitivity

Figure 71: Training and validation precision vs sensitivity Grouped b model (BCE loss)

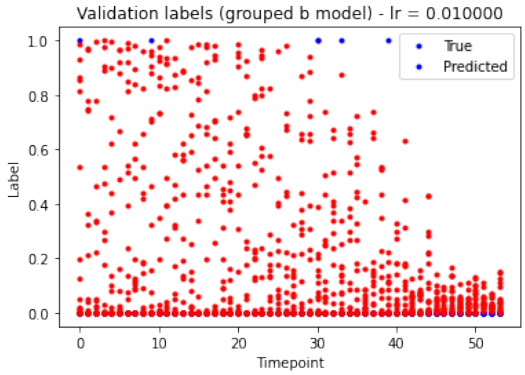
True vs predicted



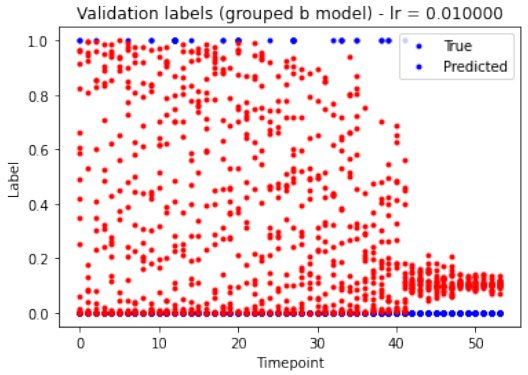
(a) True vs predicted for patient 1



(b) True vs predicted for patient 2



(c) True vs predicted for patient 3



(d) True vs predicted for patient 4

Figure 72: True values vs predicted values for all 4 validation patients Grouped b model (BCE loss)

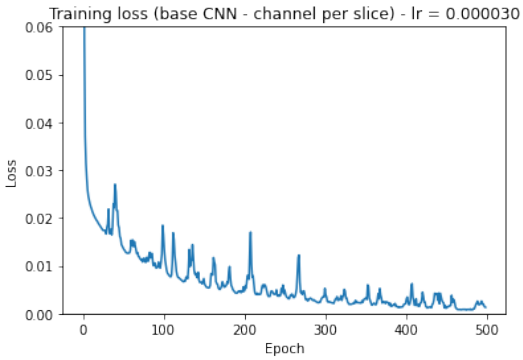
C Training and validation performance of the CNNs

The training and validation performance curves and outputs belonging to Section 5.3.2.

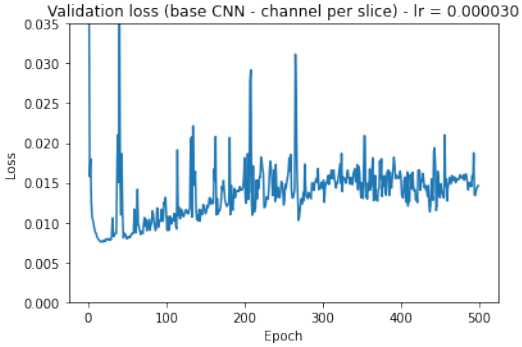
C.1 CNN - channel per slice

The training and validation performances of the CNN_{slices} and regularized CNN_{slices}.

Loss



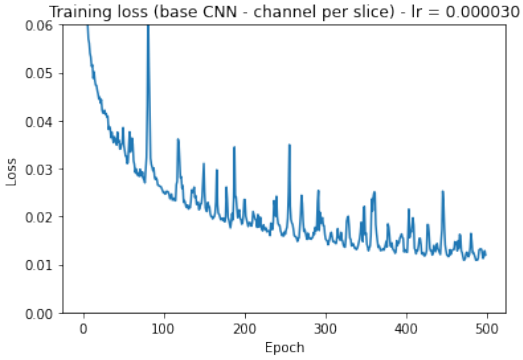
(a) Training loss



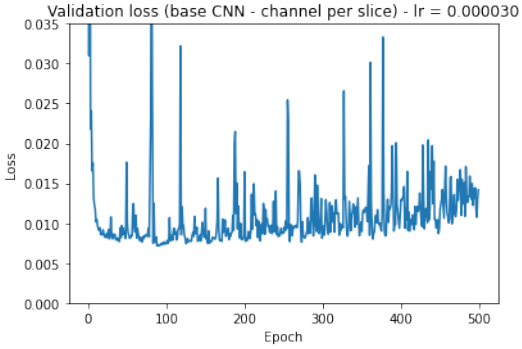
(b) Validation loss

Figure 73: Training and validation loss CNN_{slices} (MSE loss)

Loss - with regularization



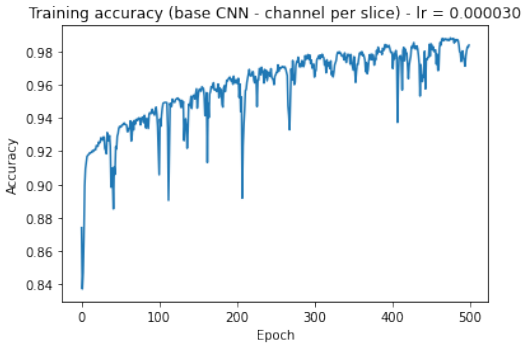
(a) Training loss - with regularization



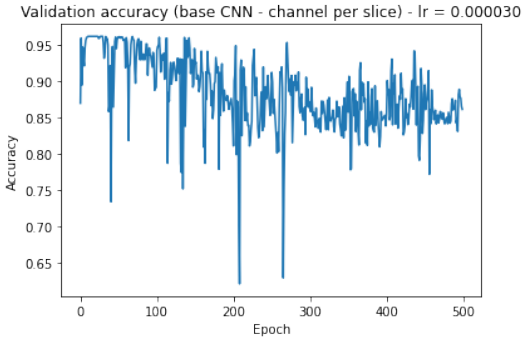
(b) Validation loss - with regularization

Figure 74: Training and validation loss CNN_{slices} - with regularization (MSE loss)

Accuracy



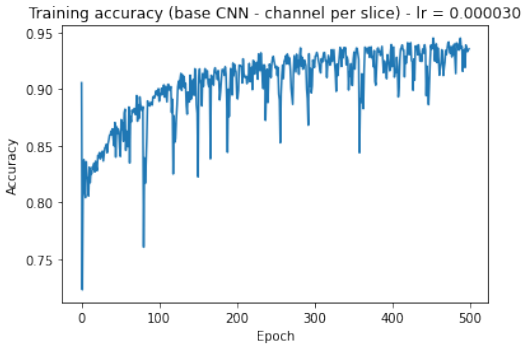
(a) Training accuracy



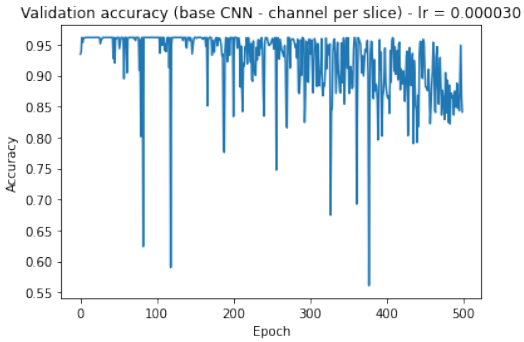
(b) Validation accuracy

Figure 75: Training and validation accuracy CNN_{slices} (MSE loss)

Accuracy - with regularization



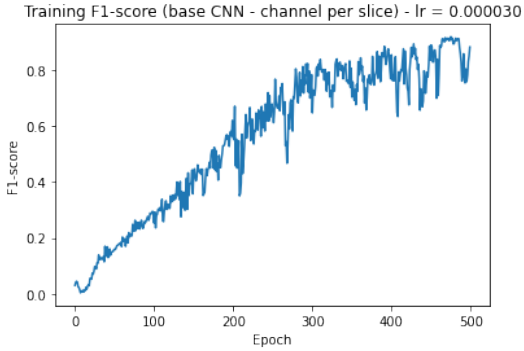
(a) Training accuracy - with regularization



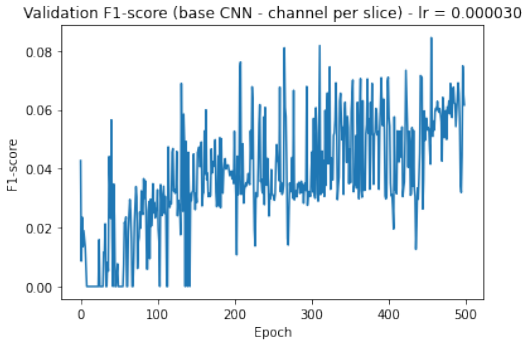
(b) Validation accuracy - with regularization

Figure 76: Training and validation accuracy CNN_{slices} - with regularization (MSE loss)

F1-score



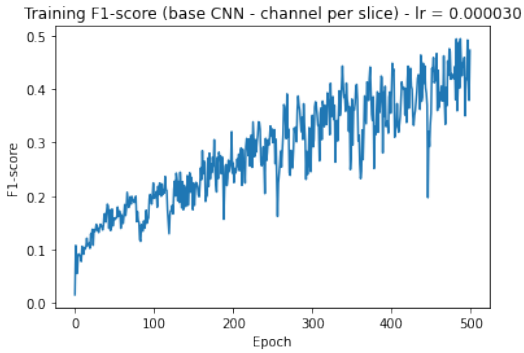
(a) Training F1-score



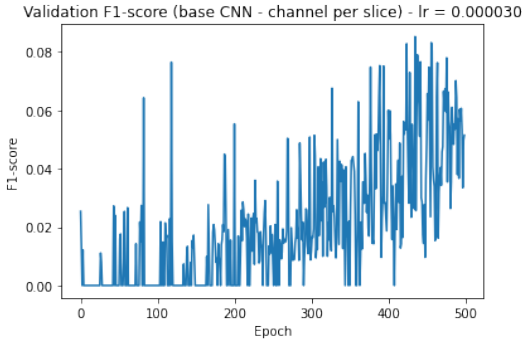
(b) Validation F1-score

Figure 77: Training and validation F1-score CNN_{slices} (MSE loss)

F1-score - with regularization



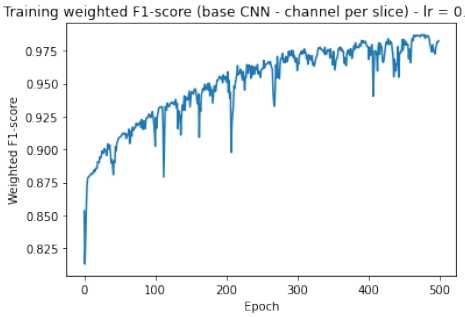
(a) Training F1-score - with regularization



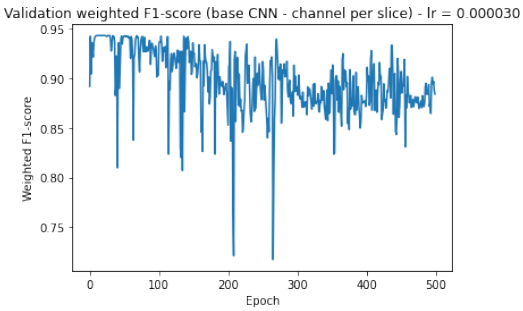
(b) Validation F1-score - with regularization

Figure 78: Training and validation F1-score CNN_{slices} - with regularization (MSE loss)

Weighted F1-score



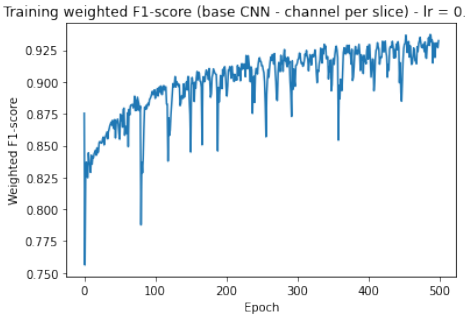
(a) Training weighted F1-score



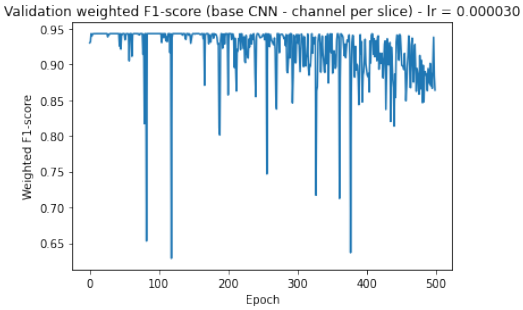
(b) Validation weighted F1-score

Figure 79: Training and validation weighted F1-score CNN_{slices} (MSE loss)

Weighted F1-score - with regularization



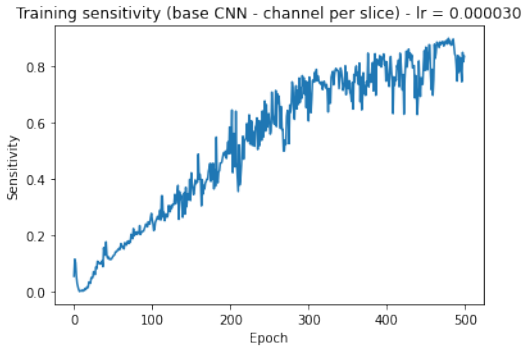
(a) Training weighted F1-score - with regularization



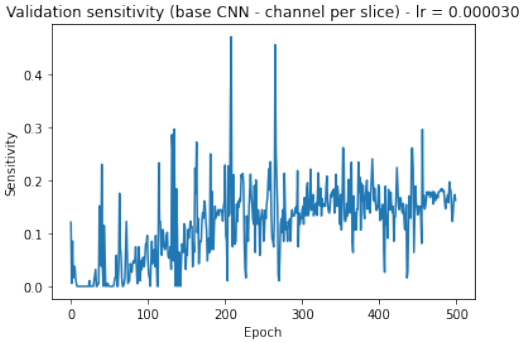
(b) Validation weighted F1-score - with regularization

Figure 80: Training and validation weighted F1-score CNN_{slices} - with regularization (MSE loss)

Sensitivity



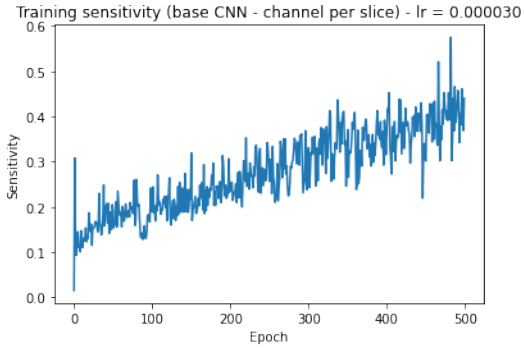
(a) Training sensitivity



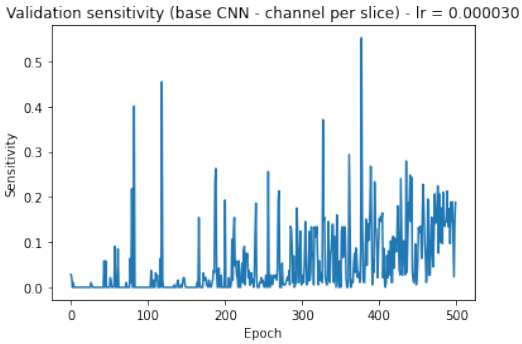
(b) Validation sensitivity

Figure 81: Training and validation sensitivity CNN_{slices} (MSE loss)

Sensitivity - with regularization



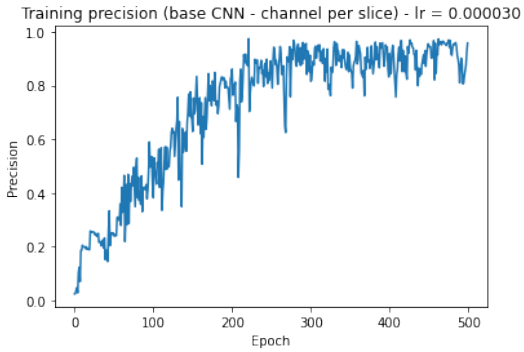
(a) Training sensitivity - with regularization



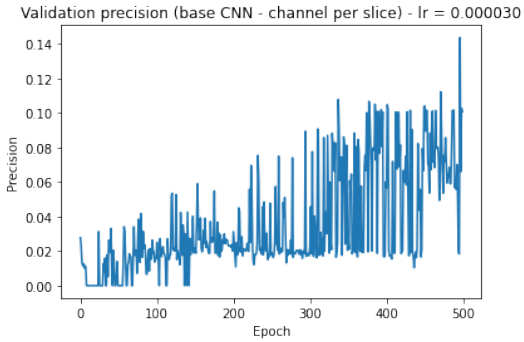
(b) Validation sensitivity - with regularization

Figure 82: Training and validation sensitivity CNN_{slices} - with regularization (MSE loss)

Precision



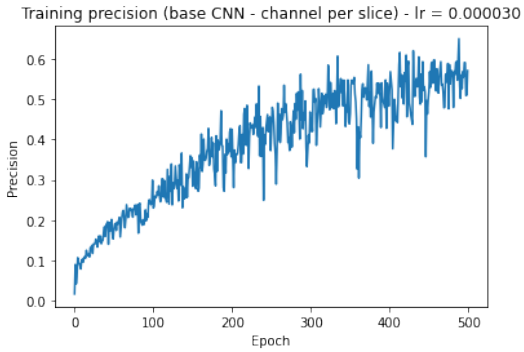
(a) Training precision



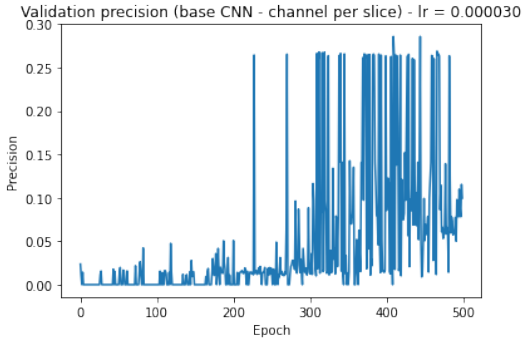
(b) Validation precision

Figure 83: Training and validation precision CNN_{slices} (MSE loss)

Precision - with regularization



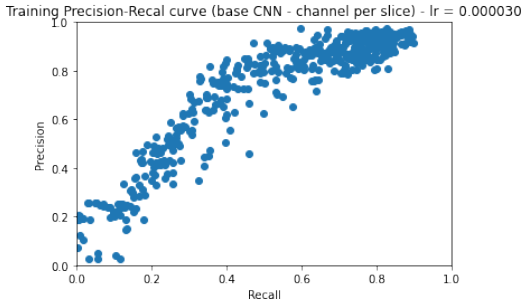
(a) Training precision - with regularization



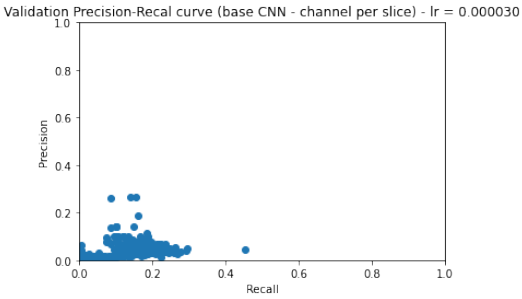
(b) Validation precision - with regularization

Figure 84: Training and validation precision CNN_{slices} - with regularization (MSE loss)

Precision vs sensitivity



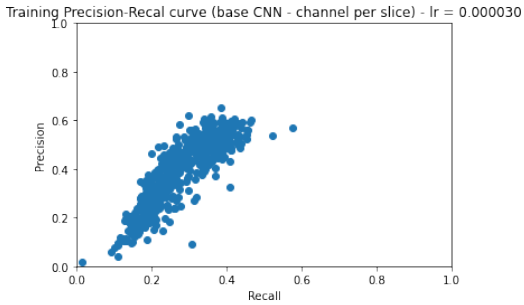
(a) Training precision vs sensitivity



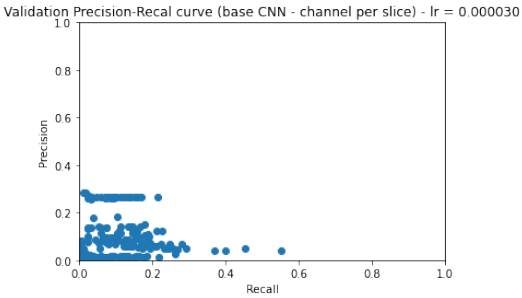
(b) Validation precision vs sensitivity

Figure 85: Training and validation precision vs sensitivity CNN_{slices} (MSE loss)

Precision vs sensitivity - with regularization



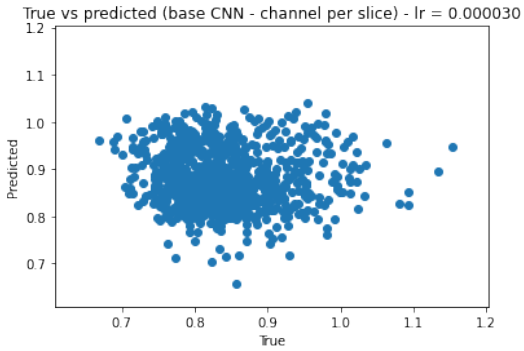
(a) Training precision vs sensitivity - with regularization



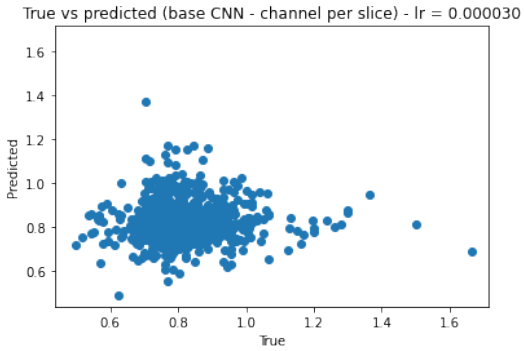
(b) Validation precision vs sensitivity - with regularization

Figure 86: Training and validation precision vs sensitivity CNN_{slices} - with regularization (MSE loss)

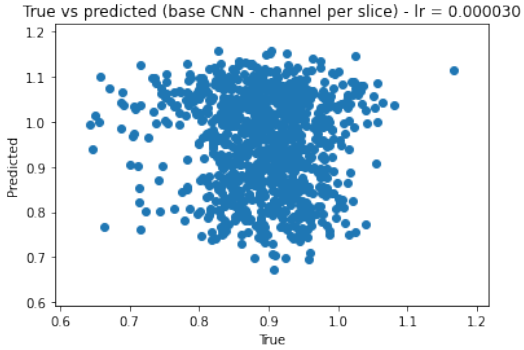
True vs predicted



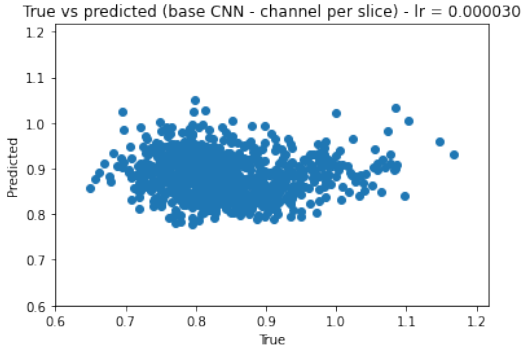
(a) True vs predicted for patient 1



(b) True vs predicted for patient 2



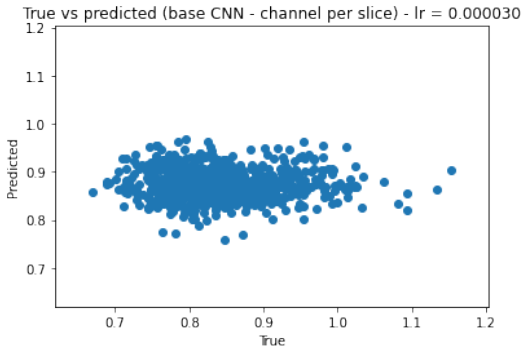
(c) True vs predicted for patient 3



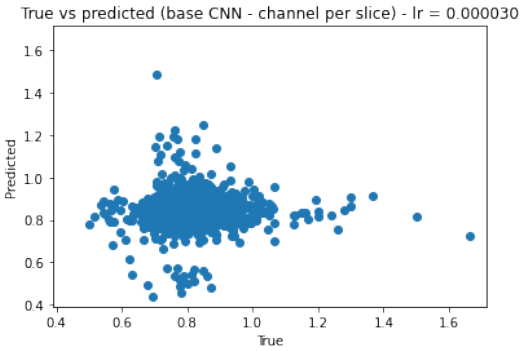
(d) True vs predicted for patient 4

Figure 87: True values vs predicted values for all 4 validation patients CNN_{slices} (MSE loss)

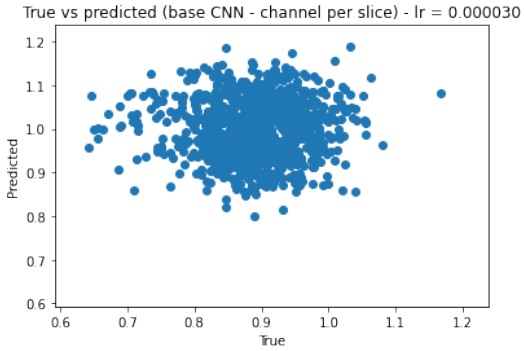
True vs predicted - with regularization



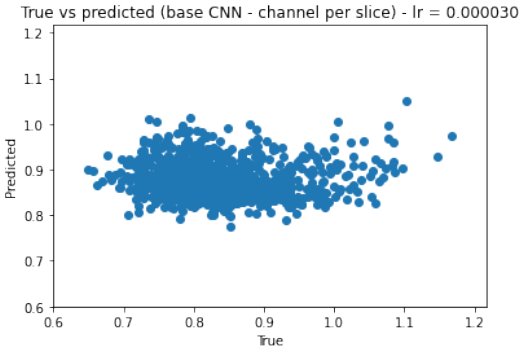
(a) True vs predicted for patient 1 - with regularization



(b) True vs predicted for patient 2 - with regularization



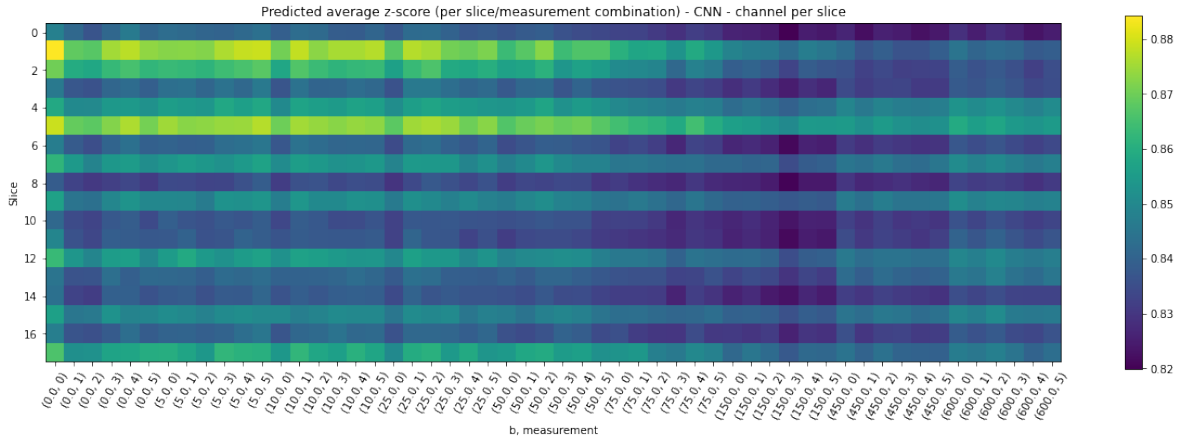
(c) True vs predicted for patient 3 - with regularization



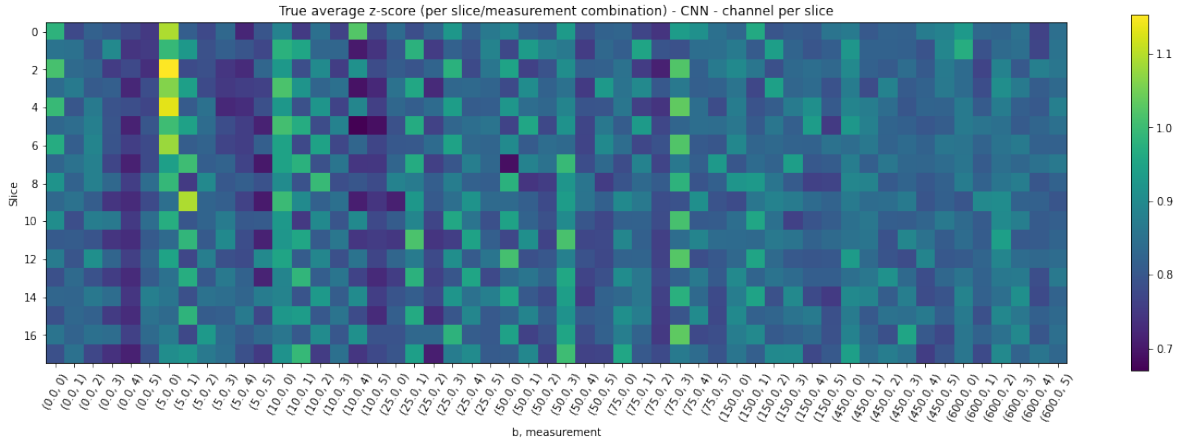
(d) True vs predicted for patient 4 - with regularization

Figure 88: True values vs predicted values for all 4 validation patients CNN_{slices} - with regularization (MSE loss)

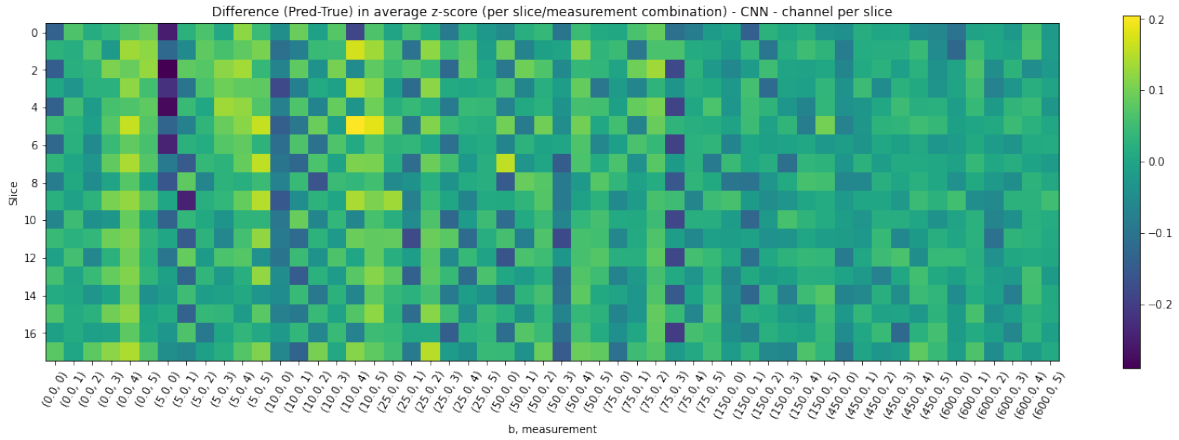
Inspect outputs



(a) Predicted mean Z-scores - test patient 1

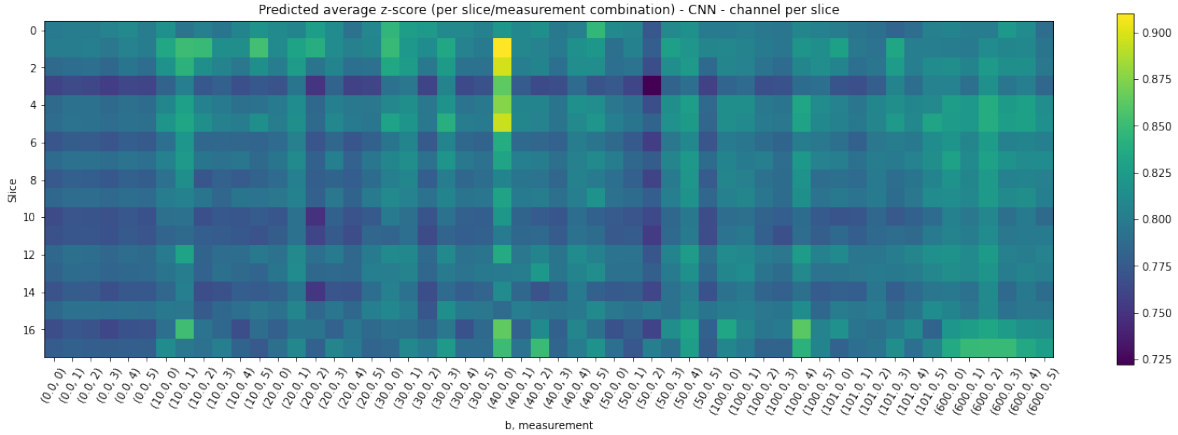


(b) True mean Z-scores - test patient 1

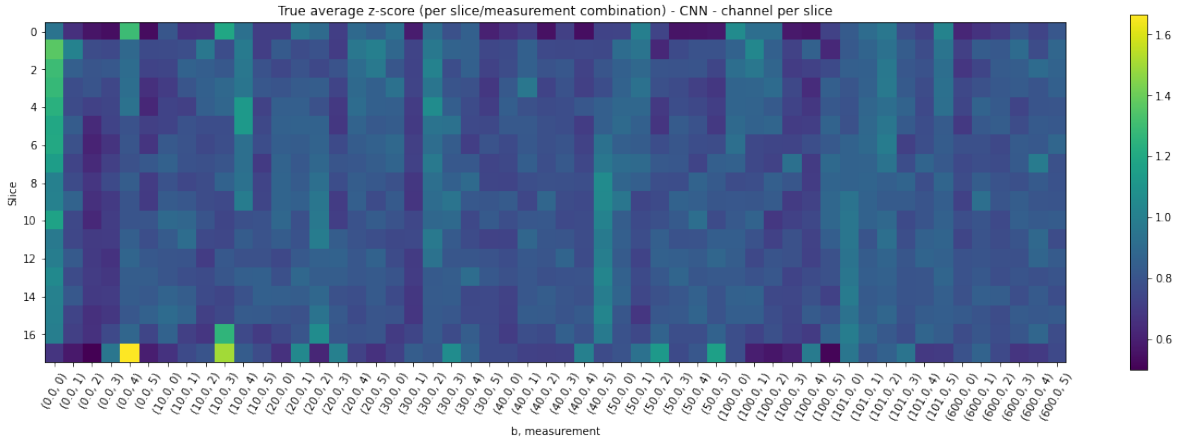


(c) Difference (Pred - True) in mean Z-scores - test patient 1

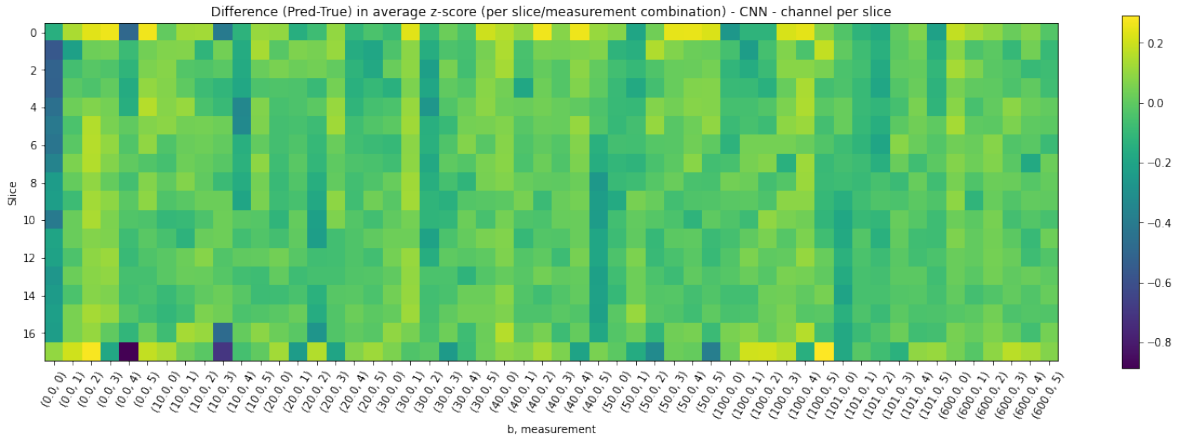
Figure 89: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 1 - CNN_{slices} (MSE loss)



(a) Predicted mean Z-scores - test patient 2

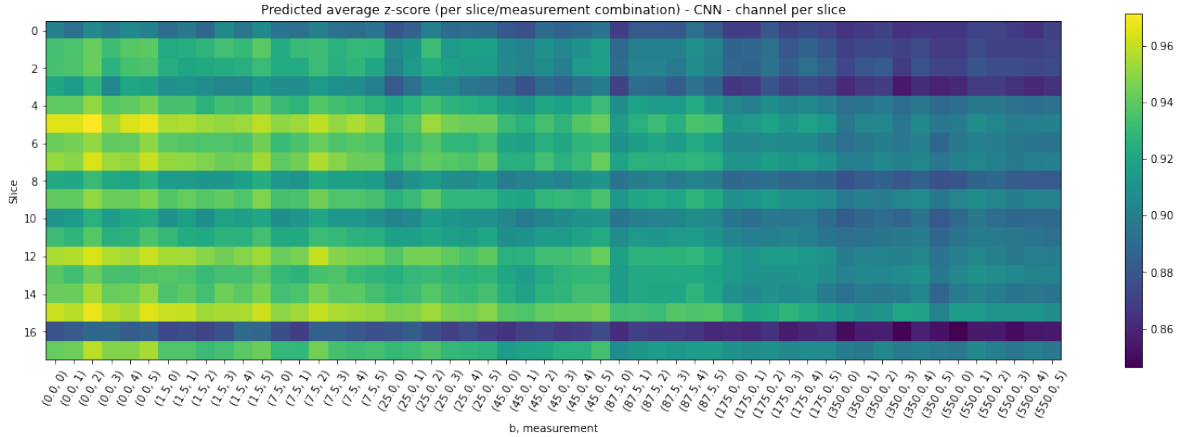


(b) True mean Z-scores - test patient 2

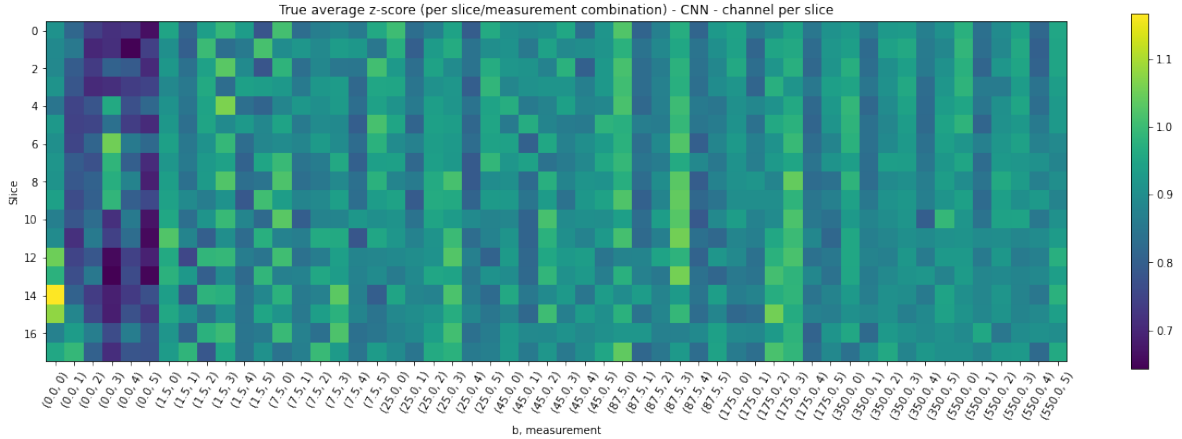


(c) Difference (Pred - True) in mean Z-scores - test patient 2

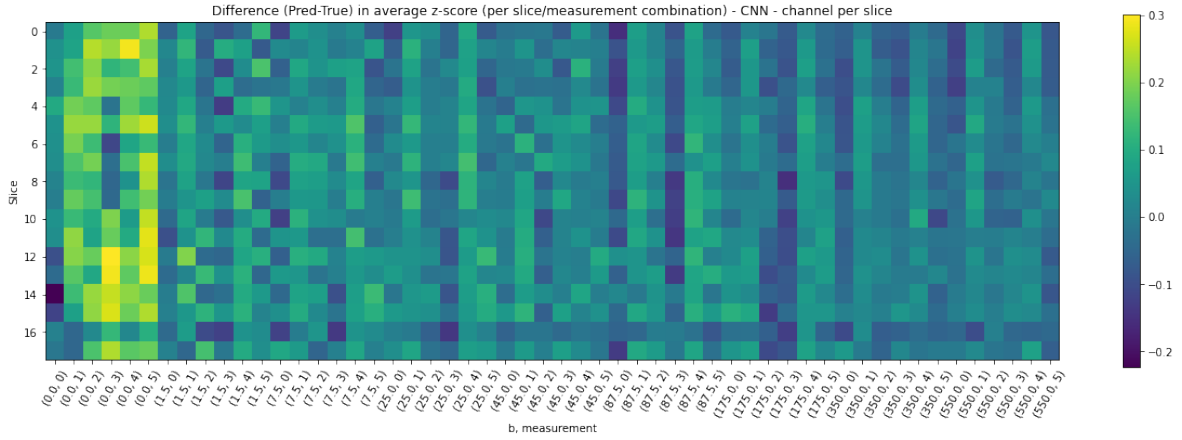
Figure 90: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 2 - CNN_{slices} (MSE loss)



(a) Predicted mean Z-scores - test patient 3

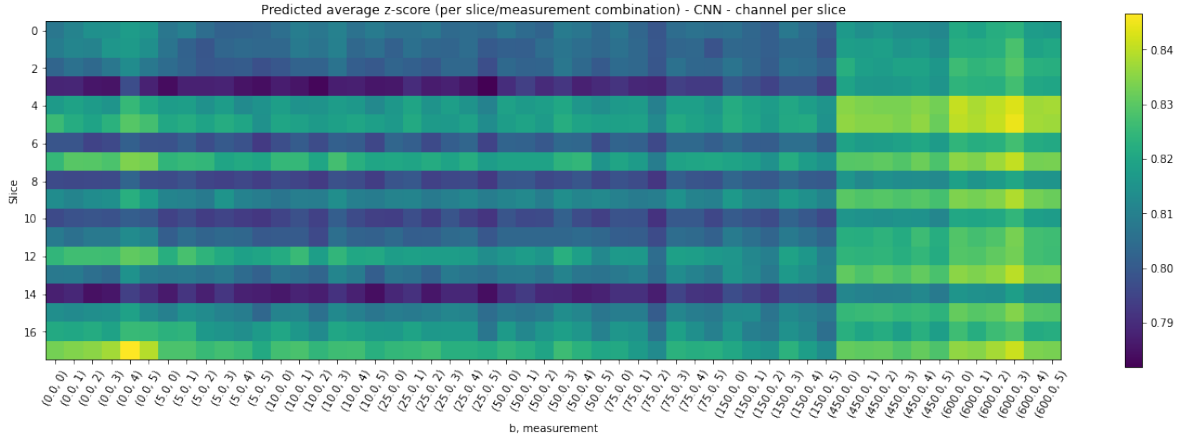


(b) True mean Z-scores - test patient 3

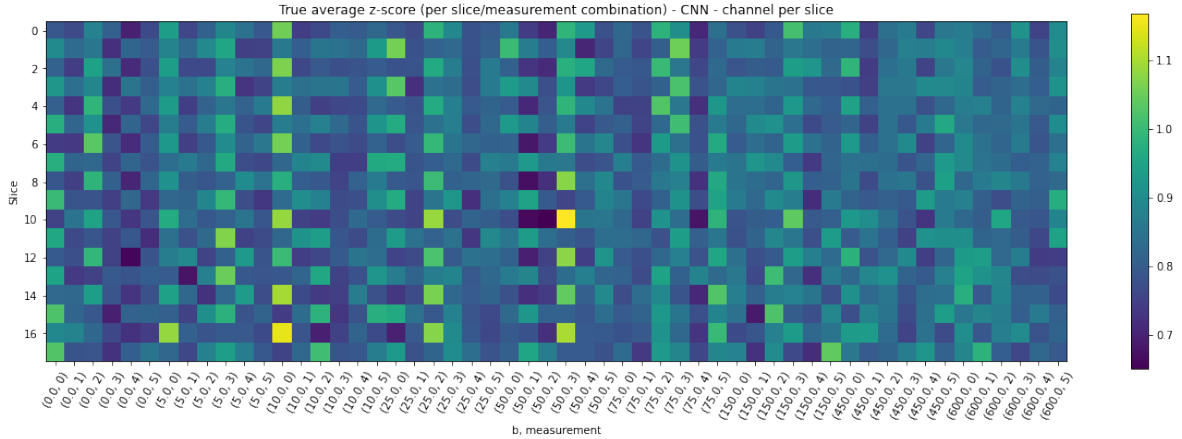


(c) Difference (Pred - True) in mean Z-scores - test patient 3

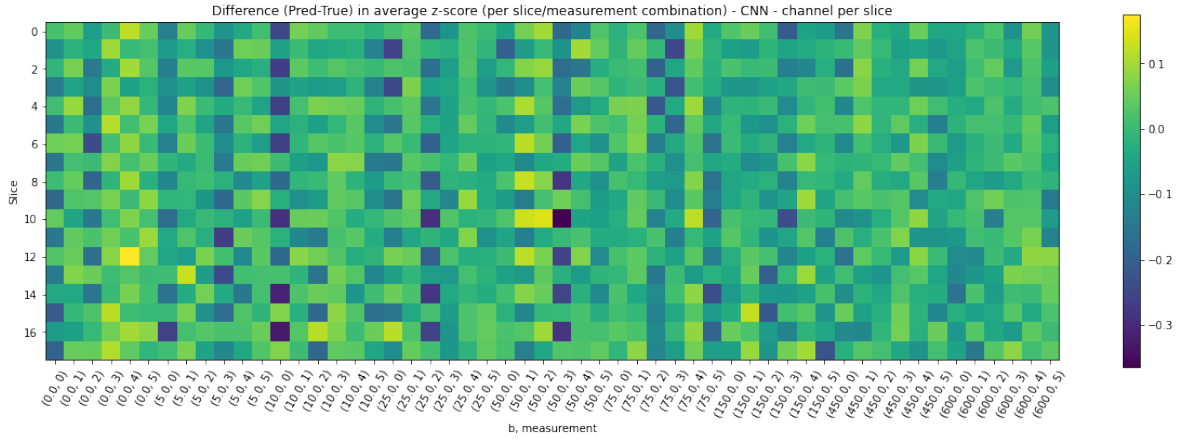
Figure 91: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 3 - CNN_{slices} (MSE loss)



(a) Predicted mean Z-scores - test patient 4



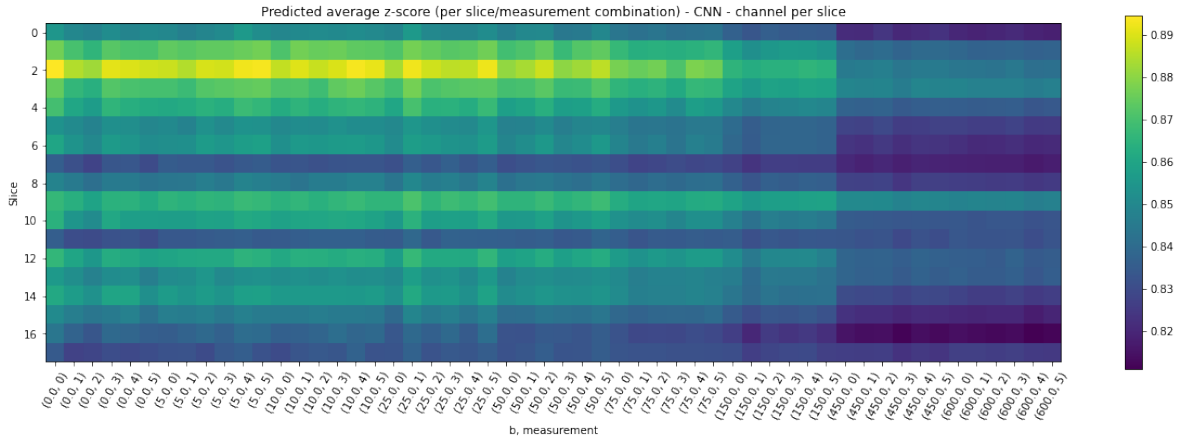
(b) True mean Z-scores - test patient 4



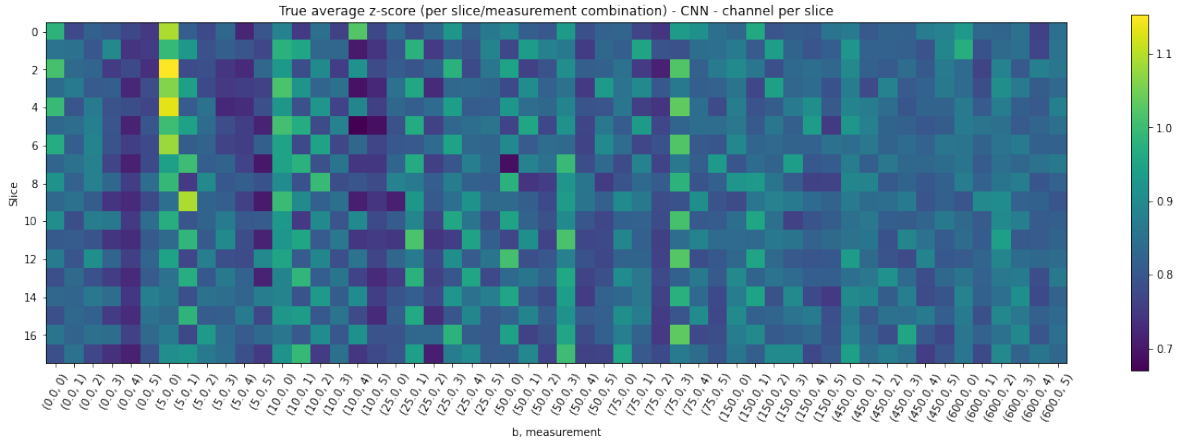
(c) Difference (Pred - True) in mean Z-scores - test patient 4

Figure 92: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 4 - CNN_{slices} (MSE loss)

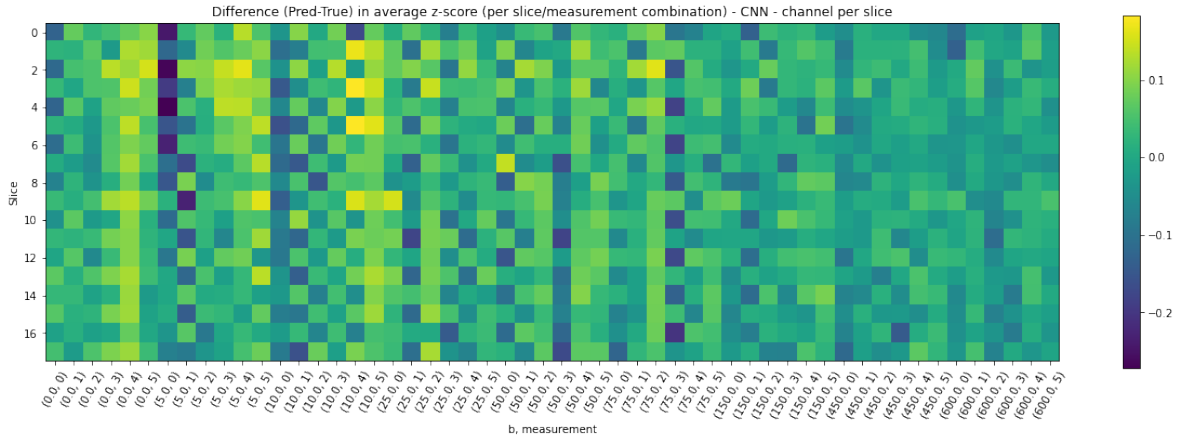
Inspect outputs - with regularization



(a) Predicted mean Z-scores - with regularization - test patient 1

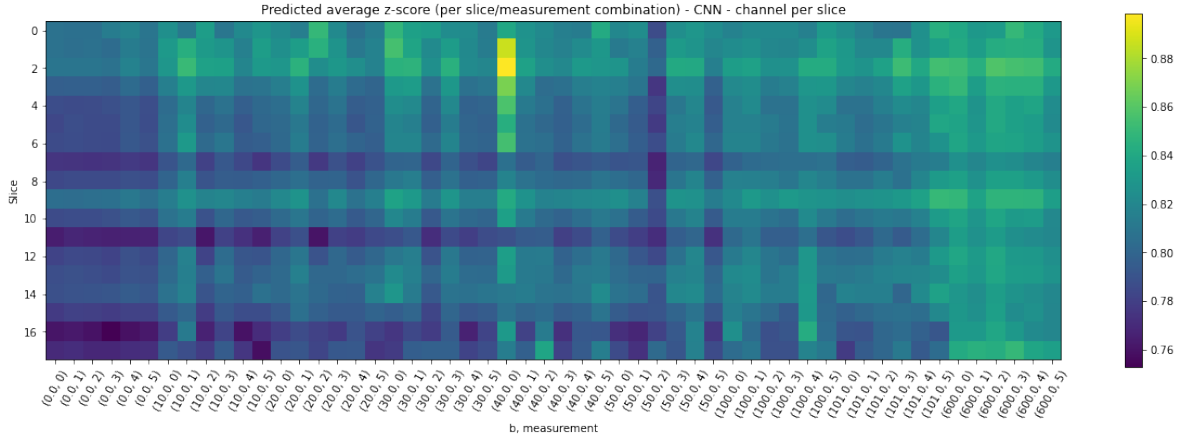


(b) True mean Z-scores - with regularization - test patient 1

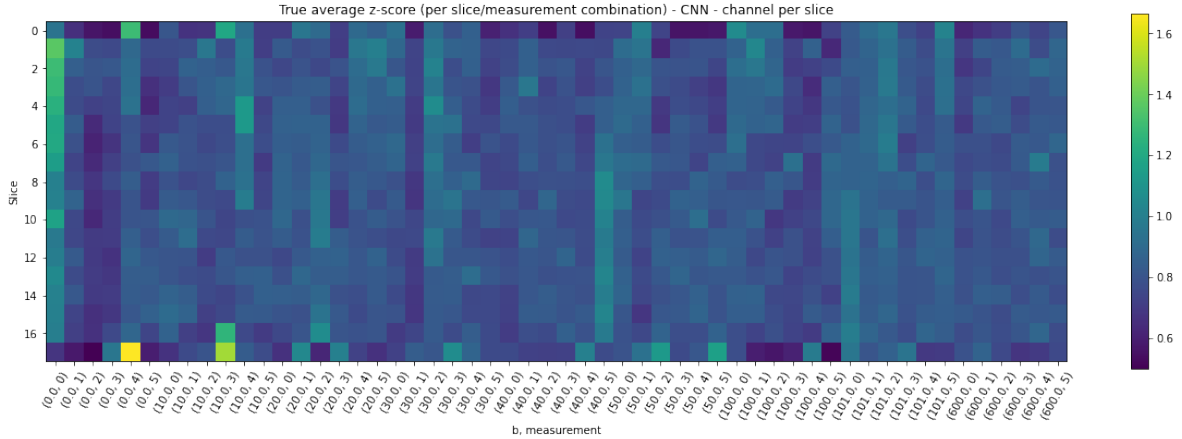


(c) Difference (Pred - True) in mean Z-scores - with regularization - test patient 1

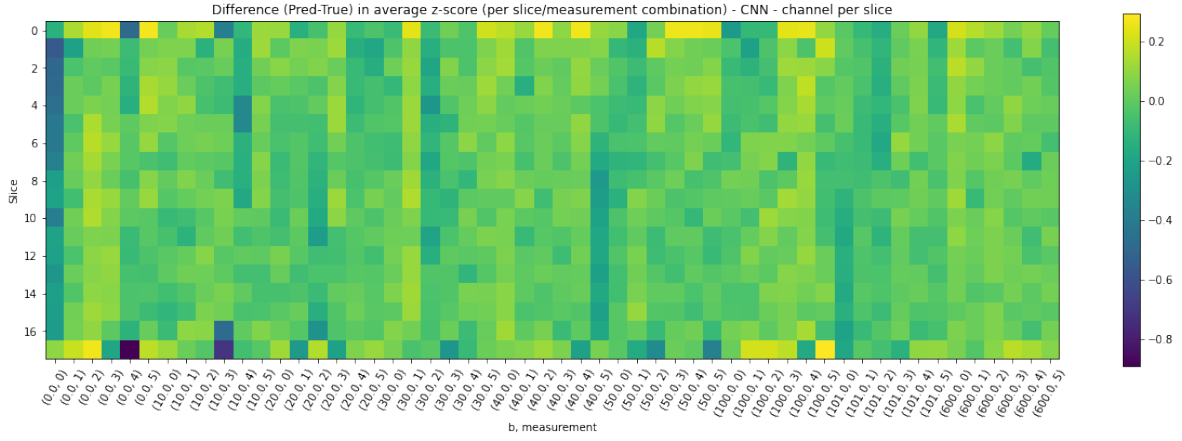
Figure 93: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 1 - CNN_{slices} - with regularization (MSE loss)



(a) Predicted mean Z-scores - with regularization - test patient 2

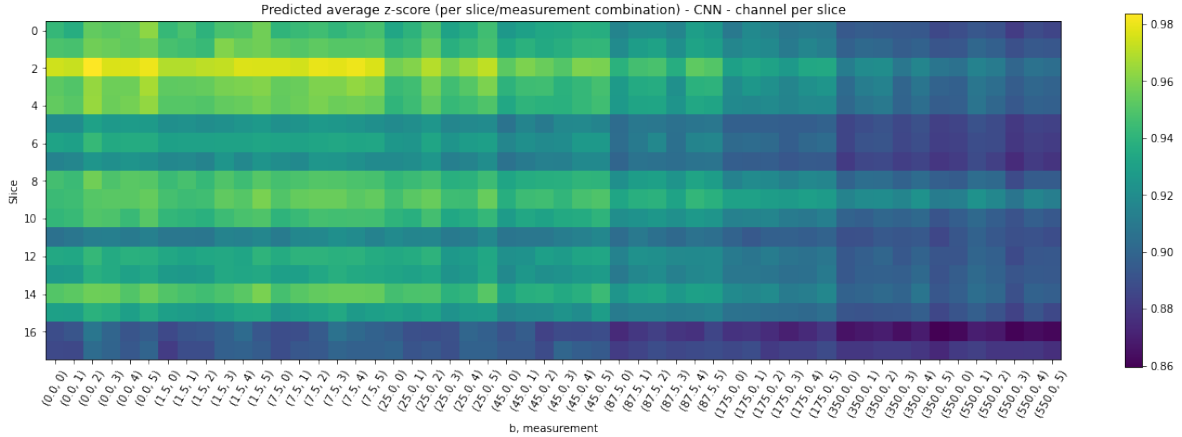


(b) True mean Z-scores - with regularization - test patient 2

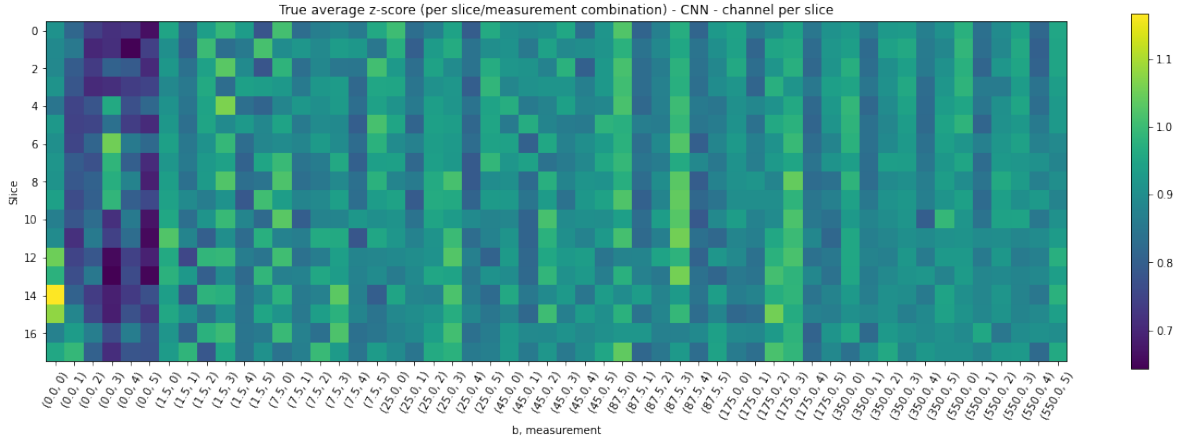


(c) Difference (Pred - True) in mean Z-scores - with regularization - test patient 2

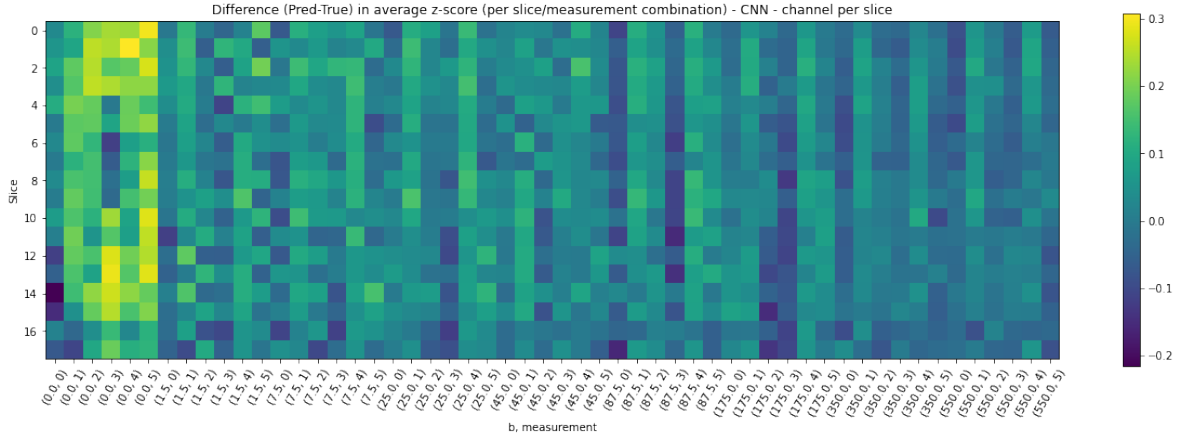
Figure 94: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 2 - CNN_{slices} - with regularization (MSE loss)



(a) Predicted mean Z-scores - with regularization - test patient 3

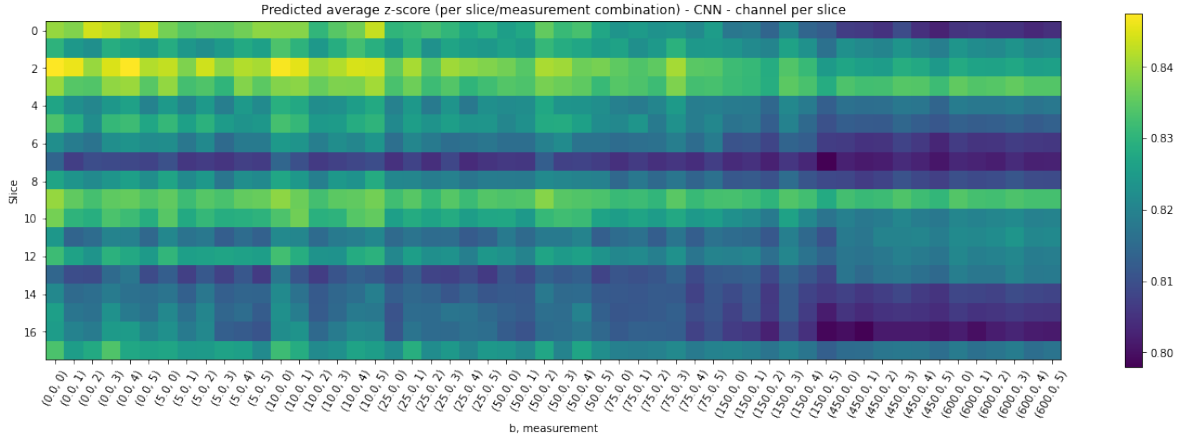


(b) True mean Z-scores - with regularization - test patient 3

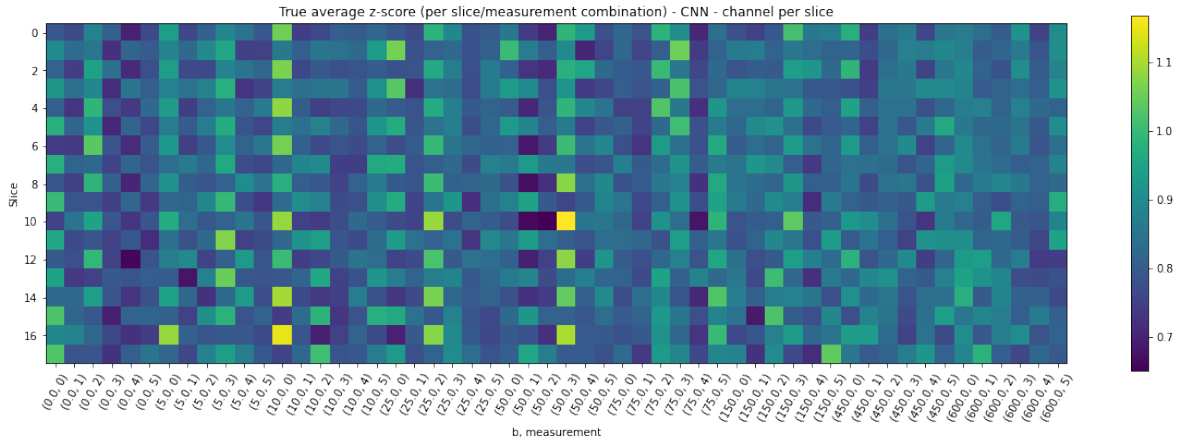


(c) Difference (Pred - True) in mean Z-scores - with regularization - test patient 3

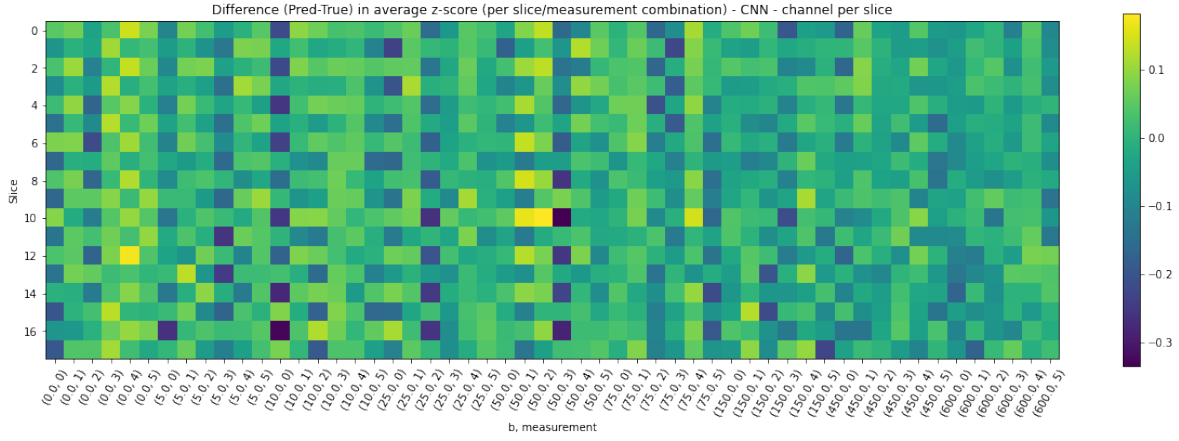
Figure 95: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 3 - CNN_{slices} - with regularization (MSE loss)



(a) Predicted mean Z-scores - with regularization - test patient 4



(b) True mean Z-scores - with regularization - test patient 4



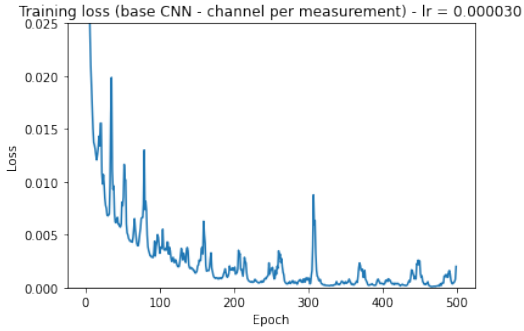
(c) Difference (Pred - True) in mean Z-scores - with regularization - test patient 4

Figure 96: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 4 - CNN_{slices} - with regularization (MSE loss)

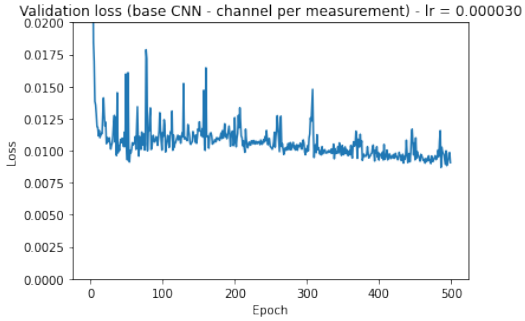
C.2 CNN - channel per measurement

The training and validation performances of the CNN_{meas} and regularized CNN_{meas}.

Loss



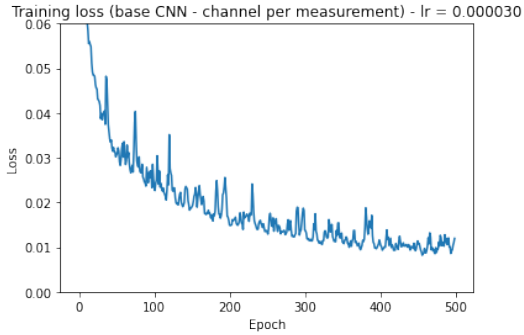
(a) Training loss



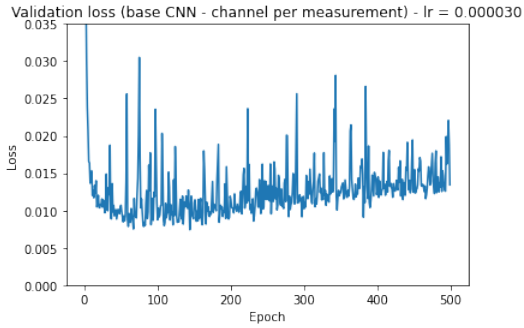
(b) Validation loss

Figure 97: Training and validation loss CNN_{meas} (MSE loss)

Loss - with regularization



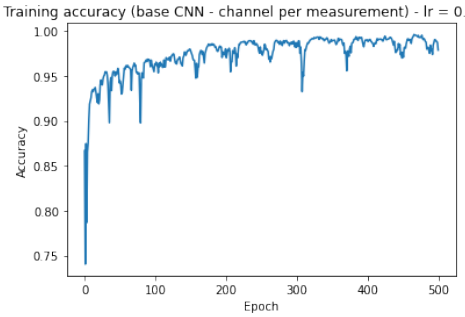
(a) Training loss - with regularization



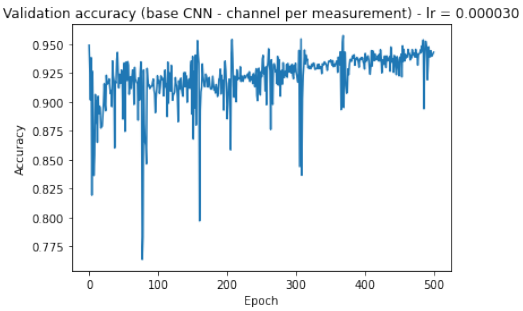
(b) Validation loss - with regularization

Figure 98: Training and validation loss CNN_{meas} - with regularization (MSE loss)

Accuracy



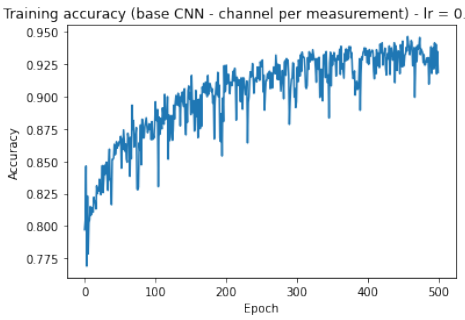
(a) Training accuracy



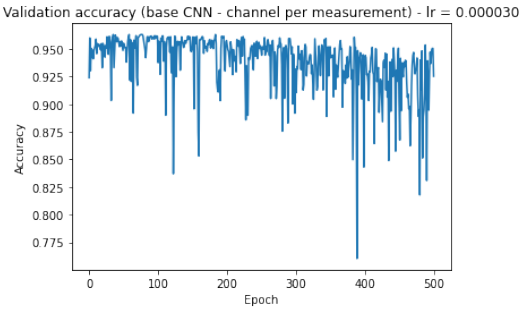
(b) Validation accuracy

Figure 99: Training and validation accuracy CNN_{meas} (MSE loss)

Accuracy - with regularization



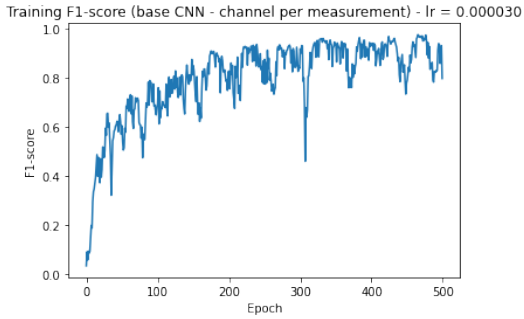
(a) Training accuracy - with regularization



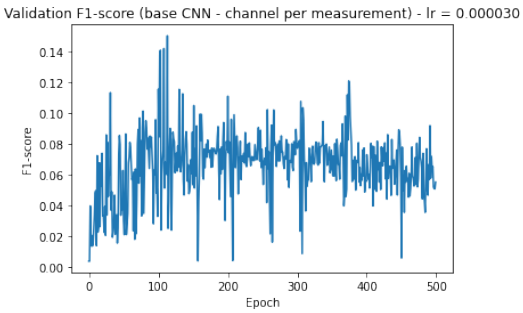
(b) Validation accuracy - with regularization

Figure 100: Training and validation accuracy CNN_{meas} - with regularization (MSE loss)

F1-score



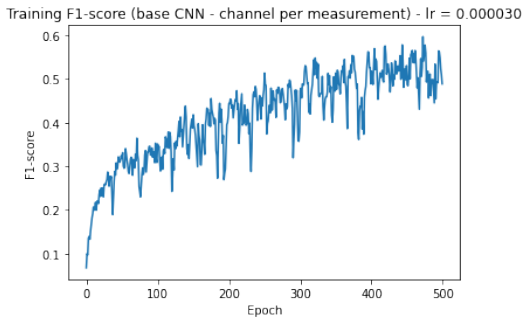
(a) Training F1-score



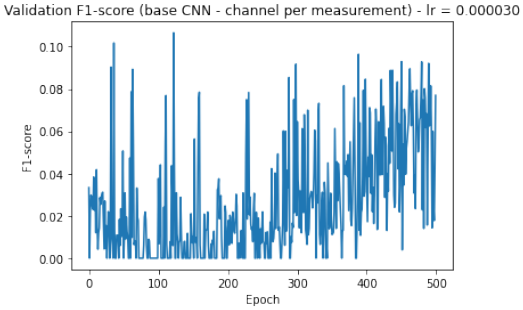
(b) Validation F1-score

Figure 101: Training and validation F1-score CNN_{meas} (MSE loss)

F1-score - with regularization



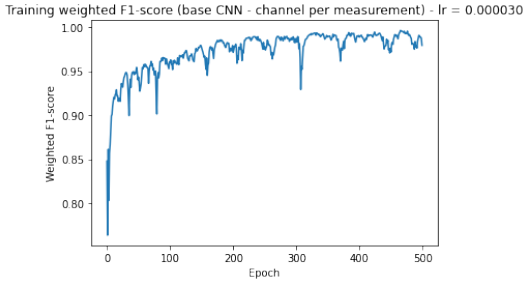
(a) Training F1-score - with regularization



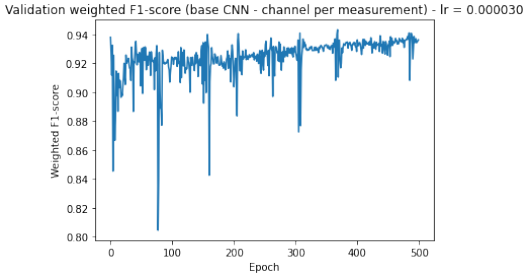
(b) Validation F1-score - with regularization

Figure 102: Training and validation F1-score CNN_{meas} - with regularization (MSE loss)

Weighted F1-score



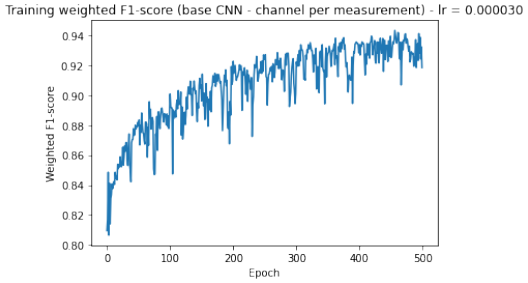
(a) Training weighted F1-score



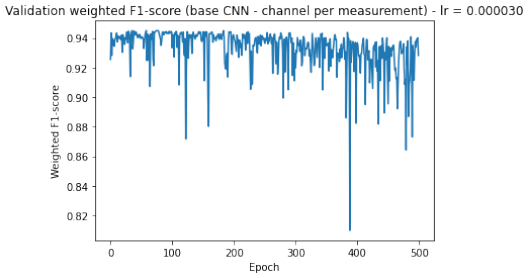
(b) Validation weighted F1-score

Figure 103: Training and validation weighted F1-score CNN_{meas} (MSE loss)

Weighted F1-score - with regularization



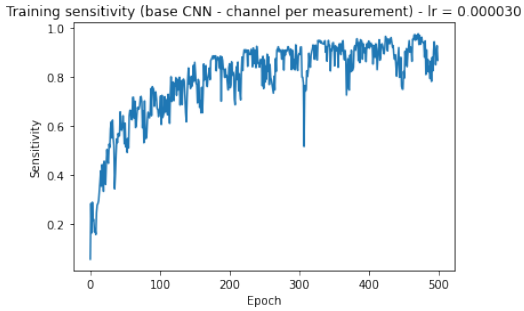
(a) Training weighted F1-score - with regularization



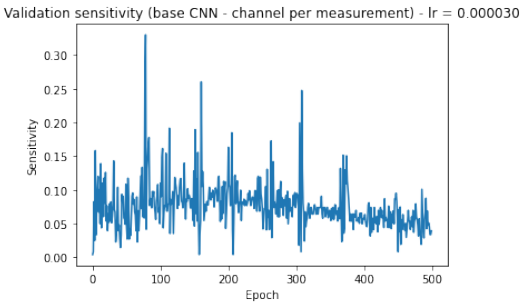
(b) Validation weighted F1-score - with regularization

Figure 104: Training and validation weighted F1-score CNN_{meas} - with regularization (MSE loss)

Sensitivity



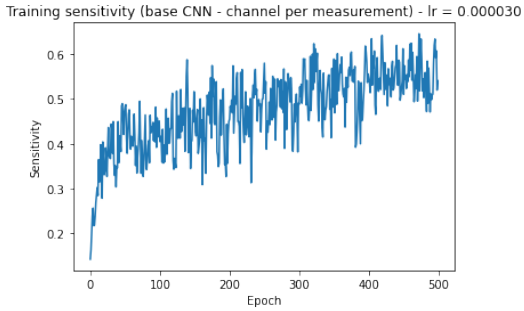
(a) Training sensitivity



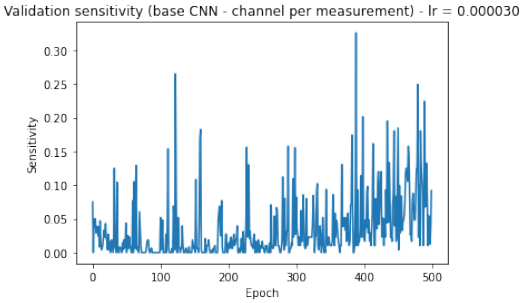
(b) Validation sensitivity

Figure 105: Training and validation sensitivity CNN_{meas} (MSE loss)

Sensitivity - with regularization



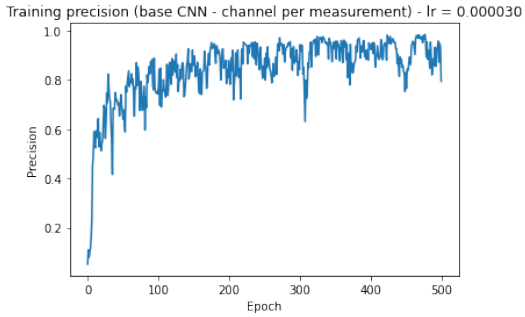
(a) Training sensitivity - with regularization



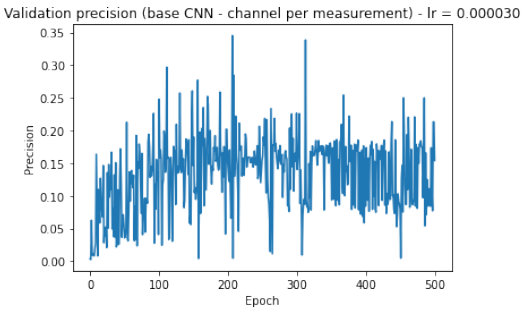
(b) Validation sensitivity - with regularization

Figure 106: Training and validation sensitivity CNN_{meas} - with regularization (MSE loss)

Precision



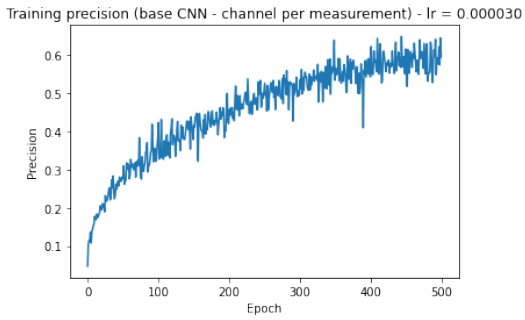
(a) Training precision



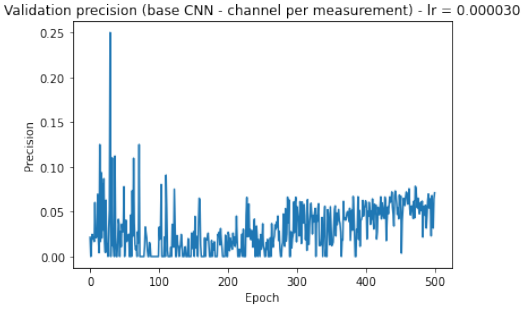
(b) Validation precision

Figure 107: Training and validation precision CNN_{meas} (MSE loss)

Precision - with regularization



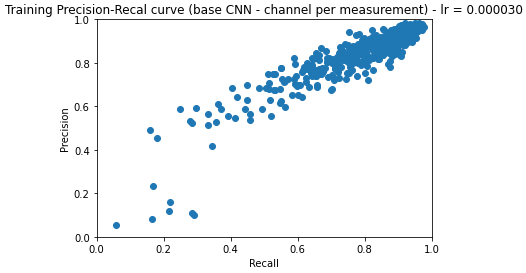
(a) Training precision - with regularization



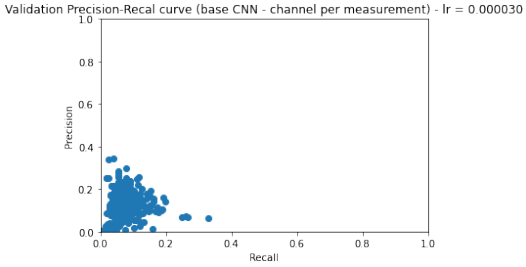
(b) Validation precision - with regularization

Figure 108: Training and validation precision CNN_{meas} - with regularization (MSE loss)

Precision vs sensitivity



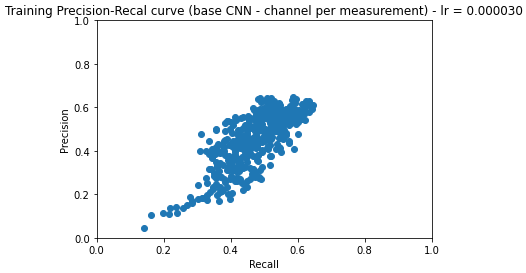
(a) Training precision vs sensitivity



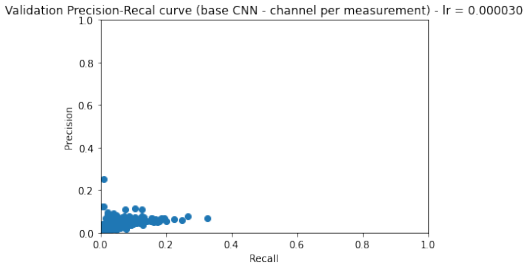
(b) Validation precision vs sensitivity

Figure 109: Training and validation precision vs sensitivity CNN_{meas} (MSE loss)

Precision vs sensitivity - with regularization



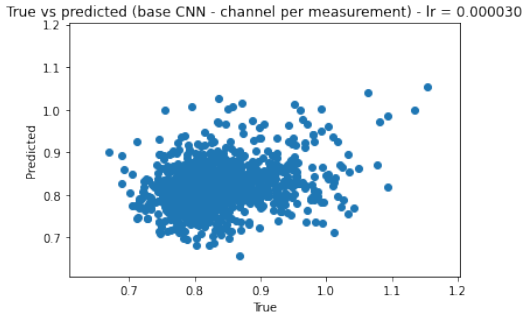
(a) Training precision vs sensitivity - with regularization



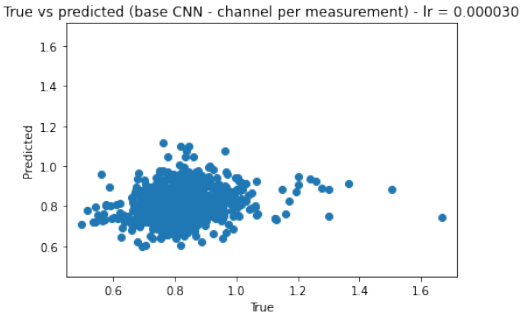
(b) Validation precision vs sensitivity - with regularization

Figure 110: Training and validation precision vs sensitivity CNN_{meas} - with regularization (MSE loss)

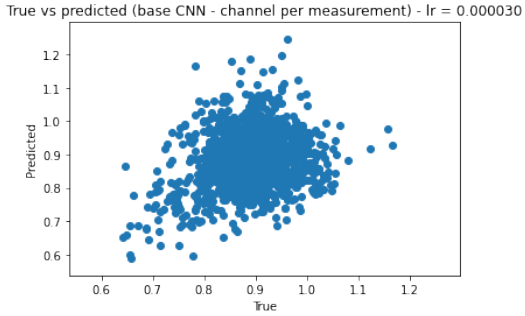
True vs predicted



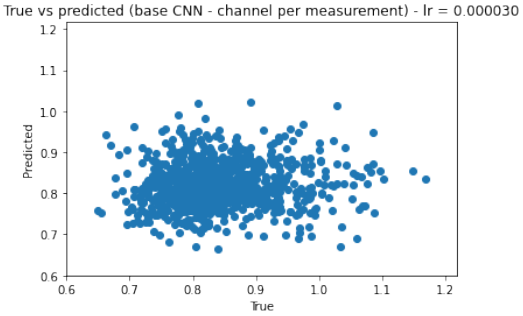
(a) True vs predicted for patient 1



(b) True vs predicted for patient 2



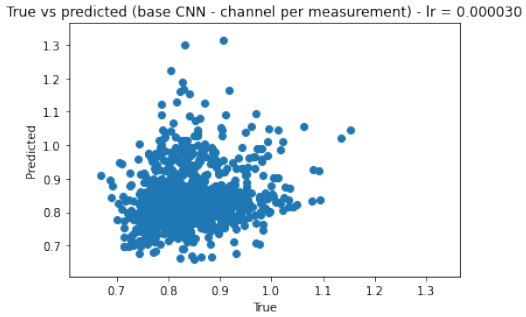
(c) True vs predicted for patient 3



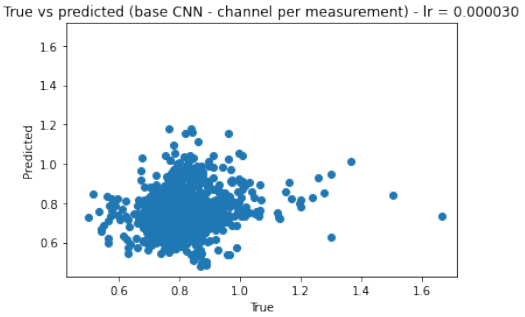
(d) True vs predicted for patient 4

Figure 111: True values vs predicted values for all 4 validation patients CNN_{meas} (MSE loss)

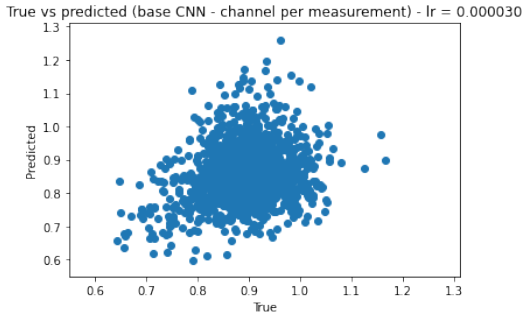
True vs predicted - with regularization



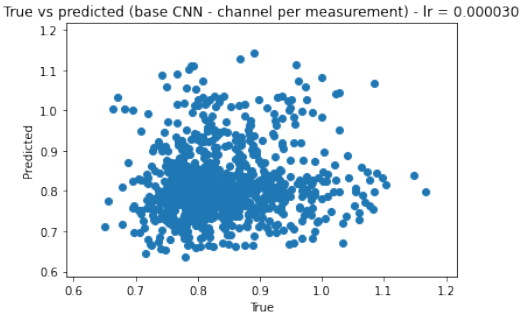
(a) True vs predicted for patient 1 - with regularization



(b) True vs predicted for patient 2 - with regularization



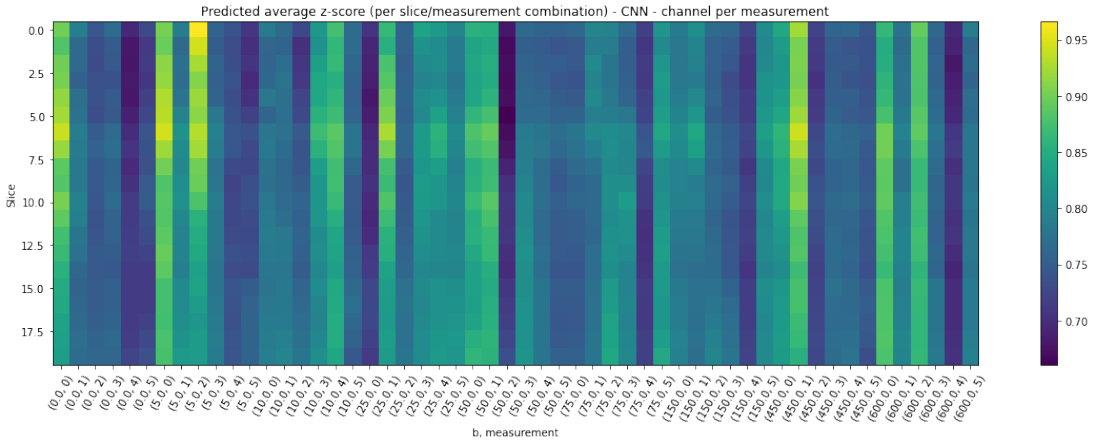
(c) True vs predicted for patient 3 - with regularization



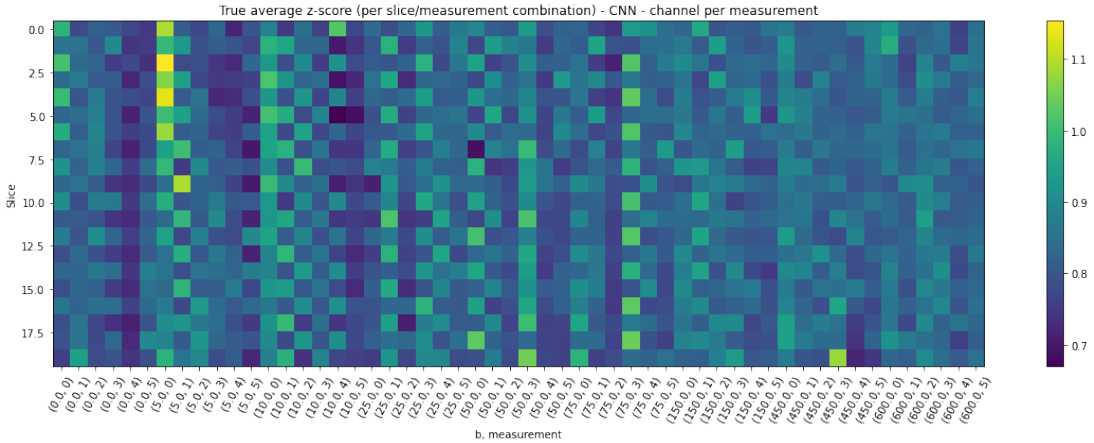
(d) True vs predicted for patient 4 - with regularization

Figure 112: True values vs predicted values for all 4 validation patients CNN_{meas} - with regularization (MSE loss)

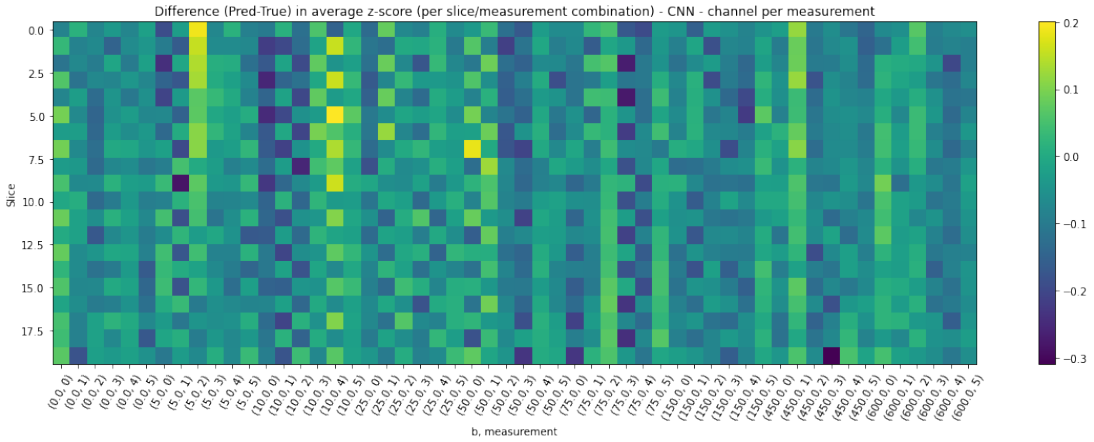
Inspect outputs



(a) Predicted mean Z-scores - test patient 1

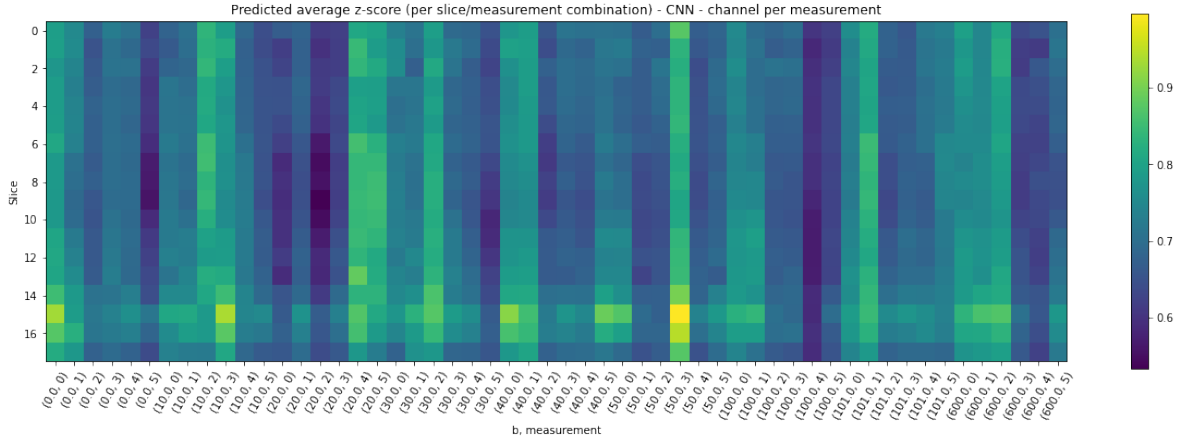


(b) True mean Z-scores - test patient 1

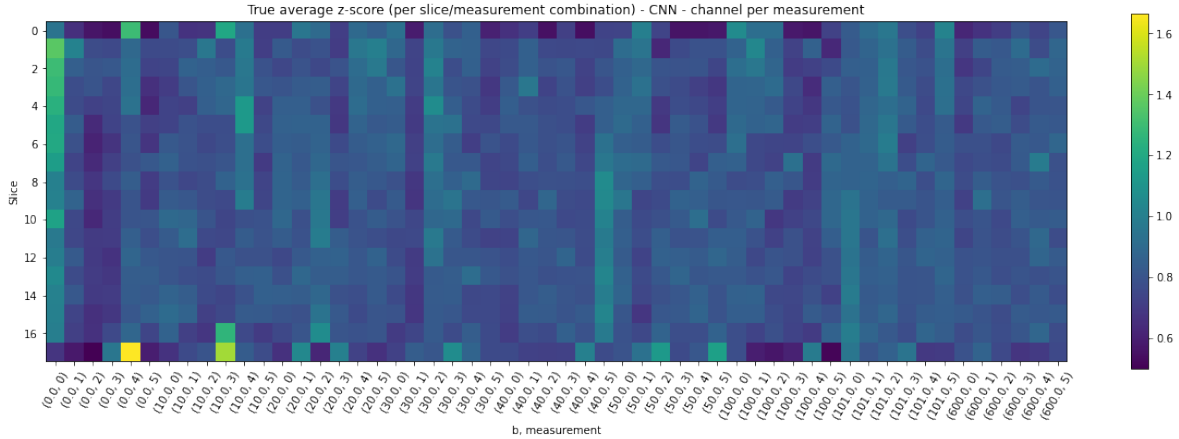


(c) Difference (Pred - True) in mean Z-scores - test patient 1

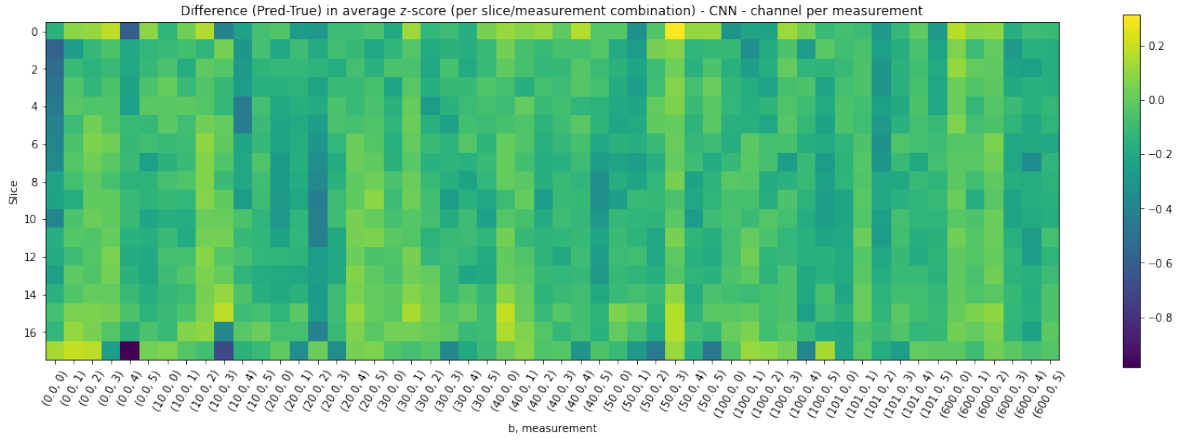
Figure 113: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 1 - CNN$_{meas}$ (MSE loss)



(a) Predicted mean Z-scores - test patient 2

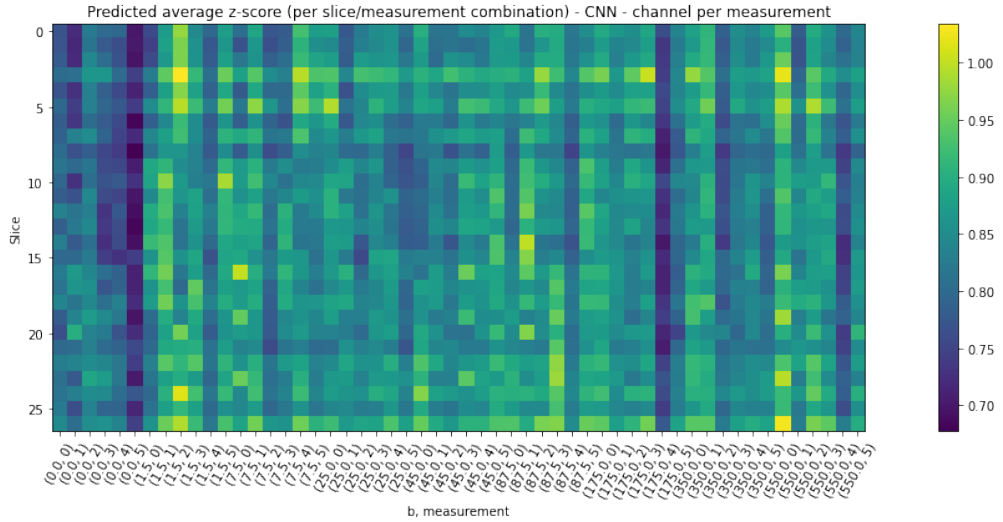


(b) True mean Z-scores - test patient 2

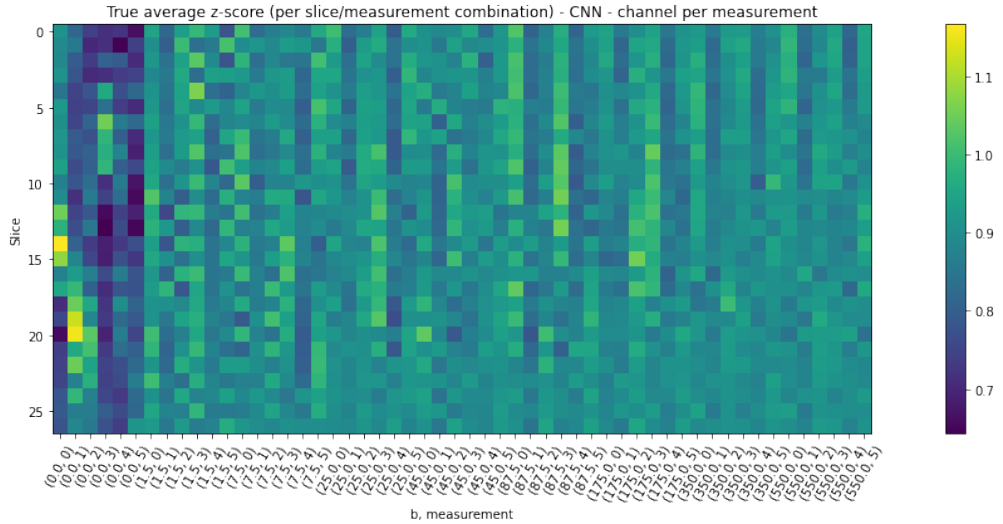


(c) Difference (Pred - True) in mean Z-scores - test patient 2

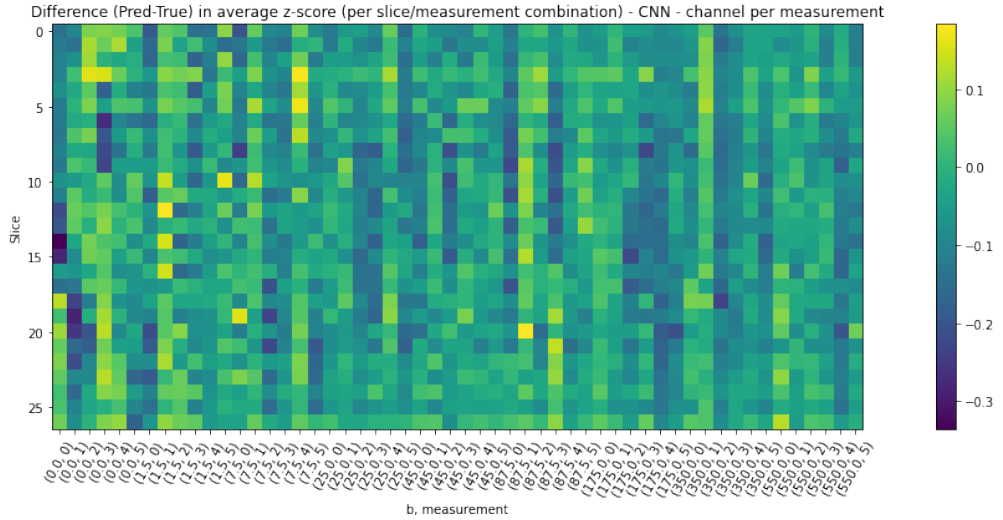
Figure 114: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 2 - CNN_{meas} (MSE loss)



(a) Predicted mean Z-scores - test patient 3

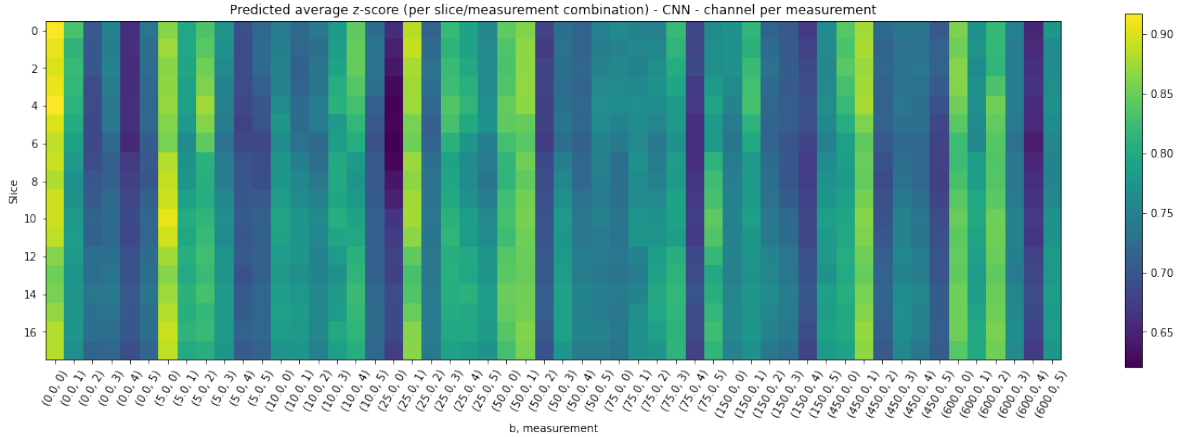


(b) True mean Z-scores - test patient 3

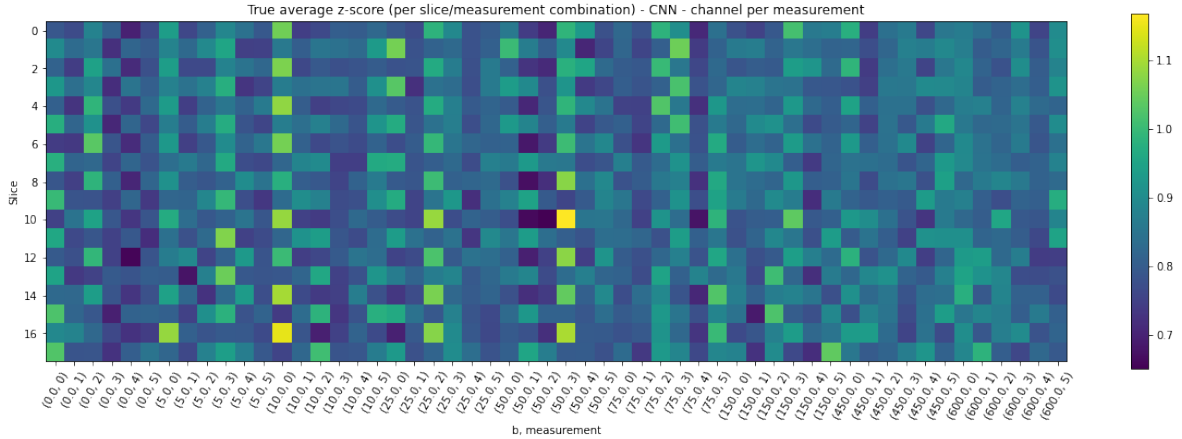


(c) Difference (Pred - True) in mean Z-scores - test patient 3

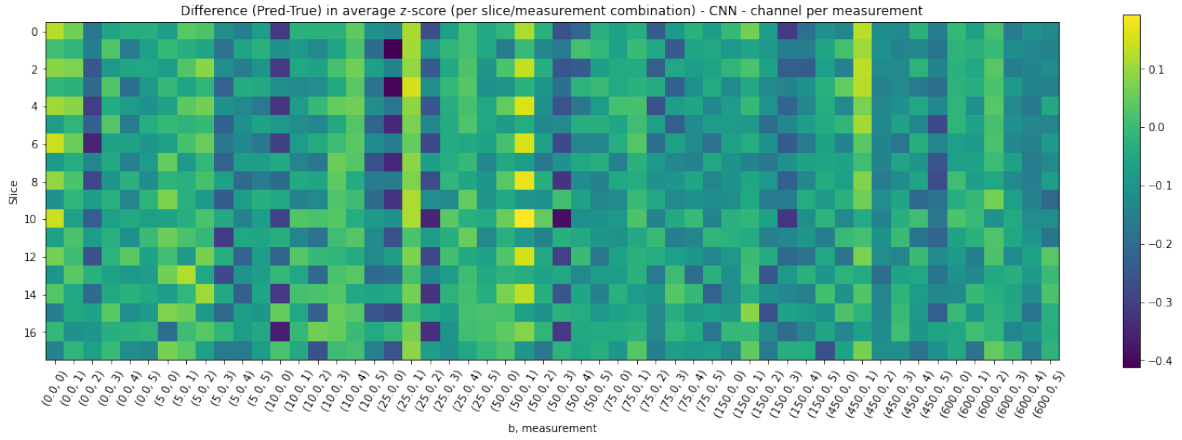
Figure 115: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 3 - CNN$_{meas}$ (MSE loss)



(a) Predicted mean Z-scores - test patient 4



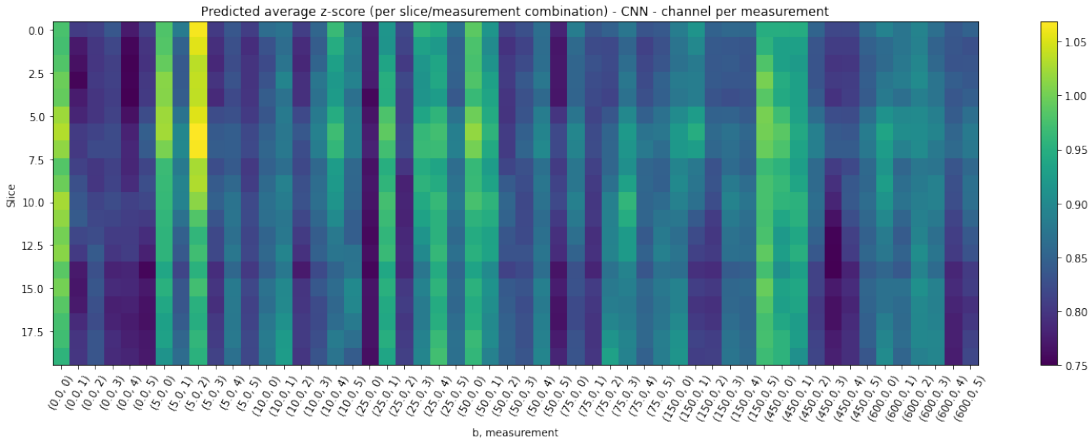
(b) True mean Z-scores - test patient 4



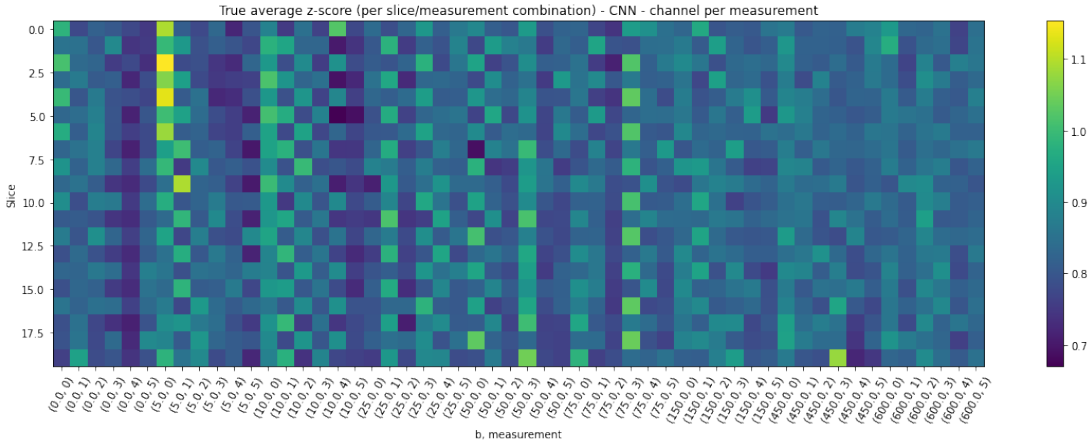
(c) Difference (Pred - True) in mean Z-scores - test patient 4

Figure 116: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 4 - CNN$_{meas}$ (MSE loss)

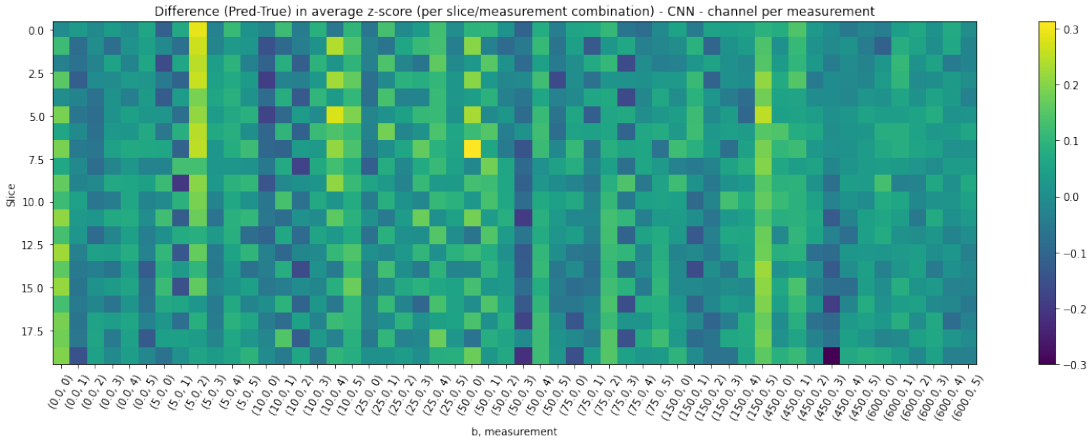
Inspect outputs - with regularization



(a) Predicted mean Z-scores - with regularization - test patient 1

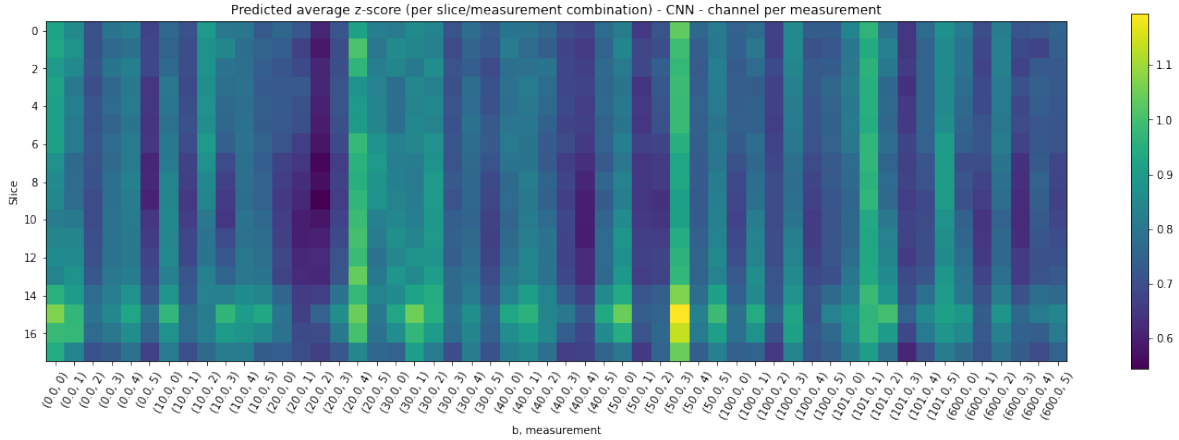


(b) True mean Z-scores - with regularization - test patient 1

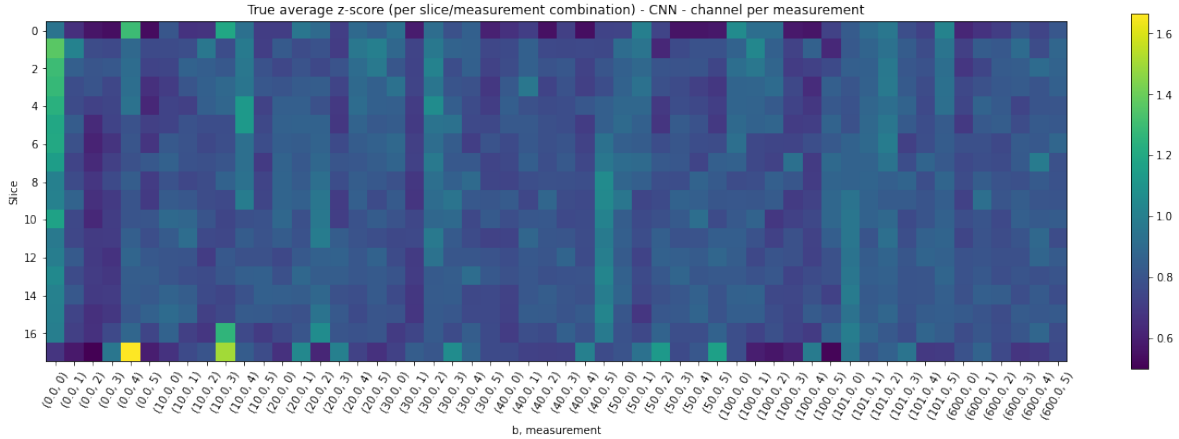


(c) Difference (Pred - True) in mean Z-scores - with regularization - test patient 1

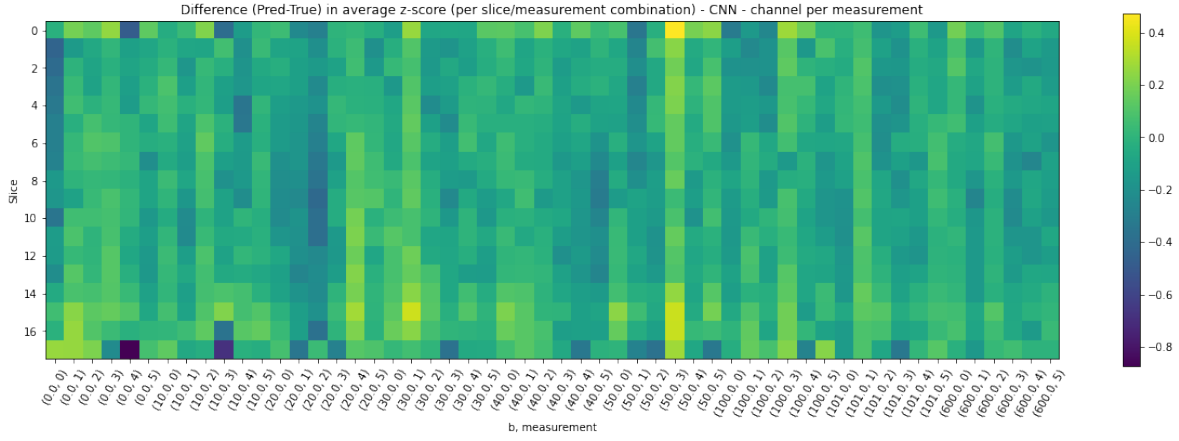
Figure 117: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 1 - CNN$_{meas}$ - with regularization (MSE loss)



(a) Predicted mean Z-scores - with regularization - test patient 2

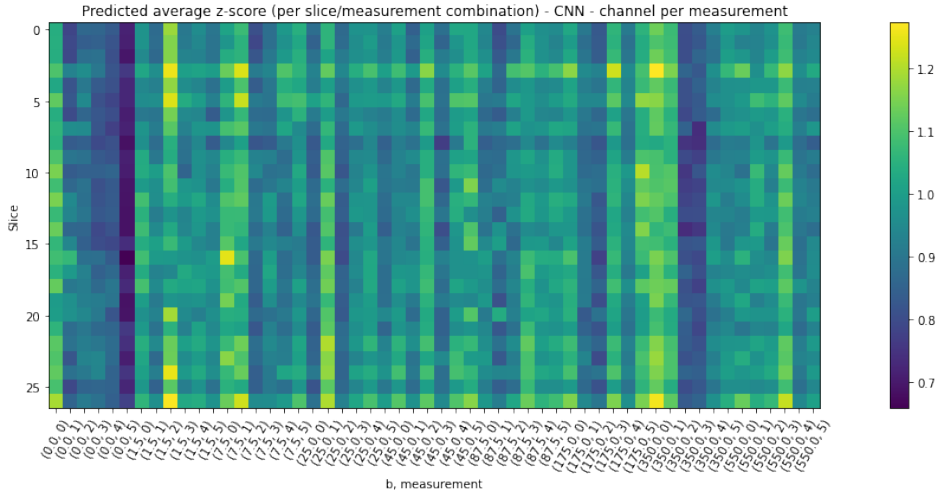


(b) True mean Z-scores - with regularization - test patient 2

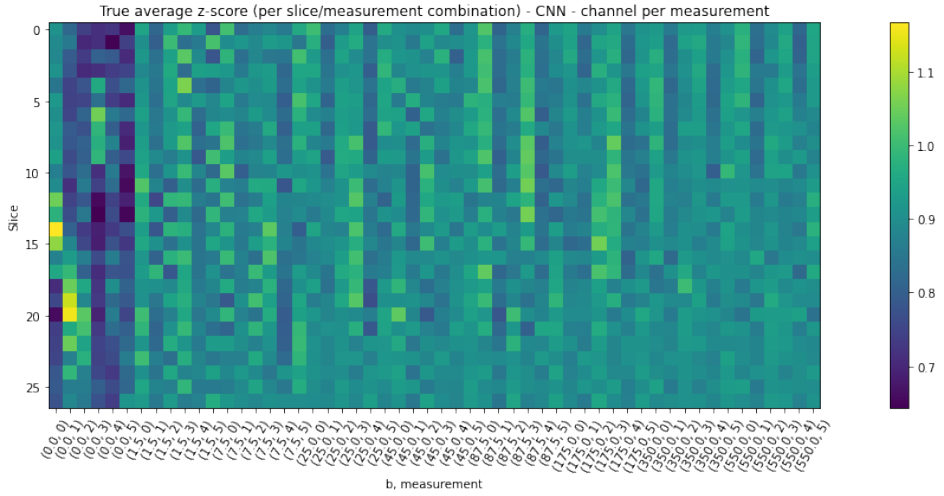


(c) Difference (Pred - True) in mean Z-scores - with regularization - test patient 2

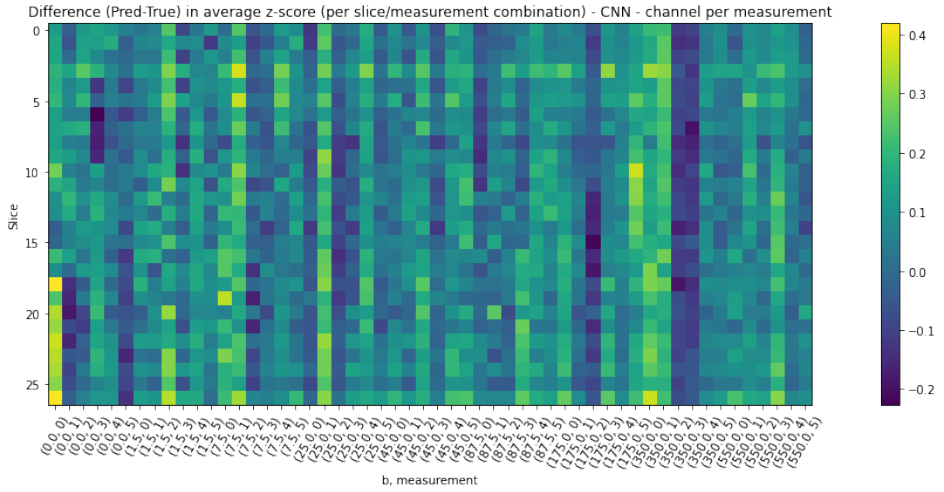
Figure 118: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 2 - CNN_{meas} - with regularization (MSE loss)



(a) Predicted mean Z-scores - with regularization - test patient 3

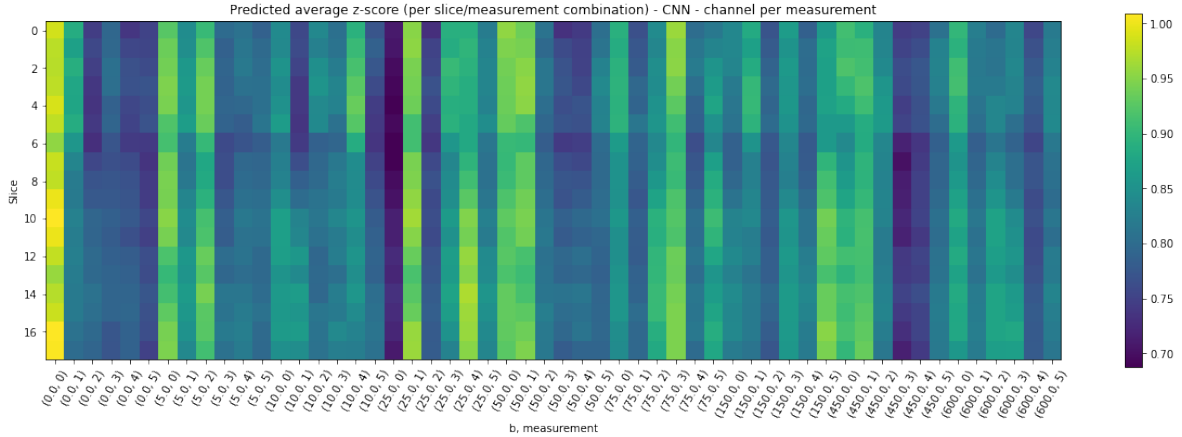


(b) True mean Z-scores - with regularization - test patient 3

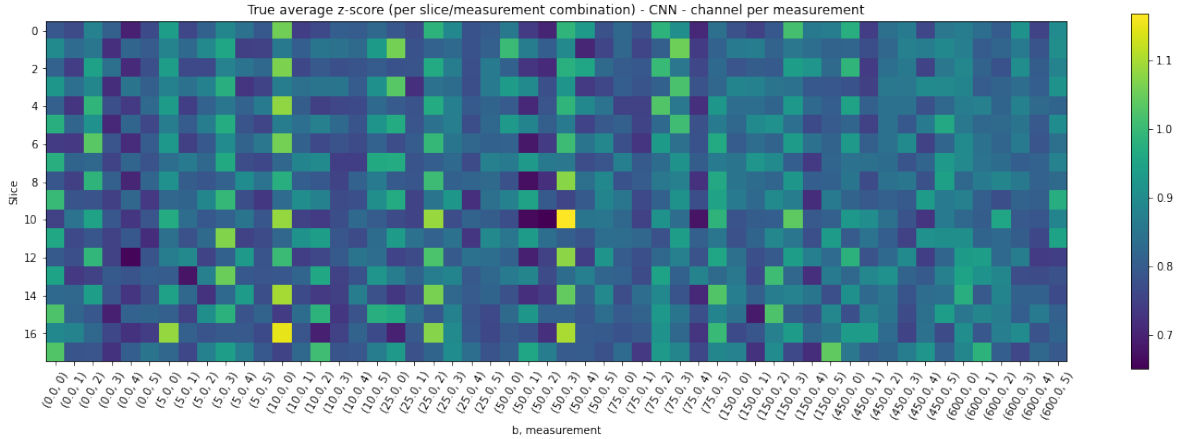


(c) Difference (Pred - True) in mean Z-scores - with regularization - test patient 3

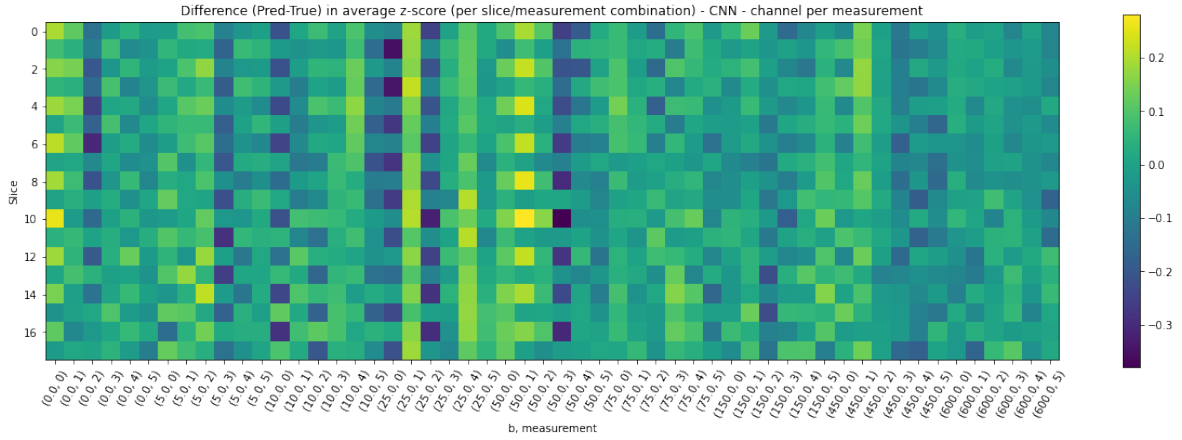
Figure 119: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 3 - CNN$_{meas}$ - with regularization (MSE loss)



(a) Predicted mean Z-scores - with regularization - test patient 4



(b) True mean Z-scores - with regularization - test patient 4



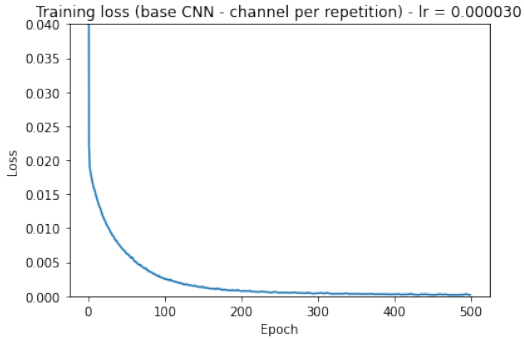
(c) Difference (Pred - True) in mean Z-scores - with regularization - test patient 4

Figure 120: Predicted, true and difference in mean Z-score per slice/timepoint pair for test patient 4 - CNN$_{meas}$ - with regularization (MSE loss)

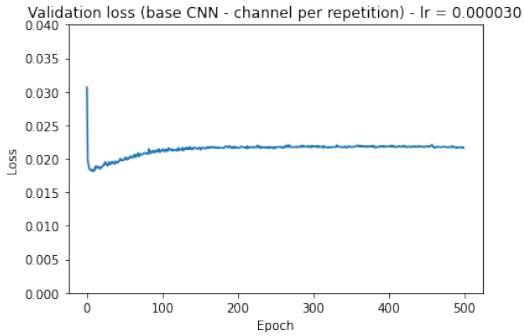
C.3 CNN - channel per repetition

The training and validation performances of the CNN_{reps} and regularized CNN_{reps}.

Loss



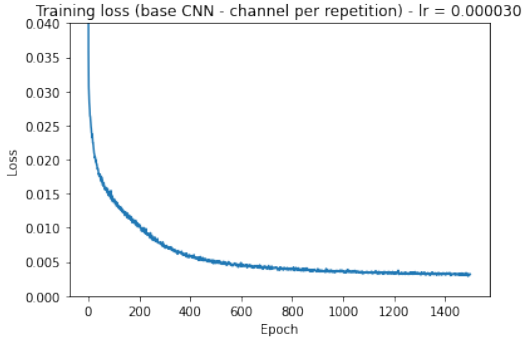
(a) Training loss



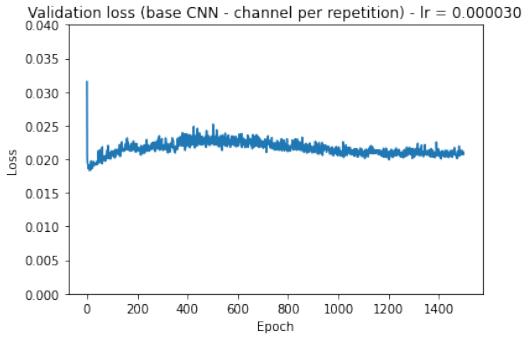
(b) Validation loss

Figure 121: Training and validation loss CNN_{reps} model (MSE loss)

Loss - with regularization



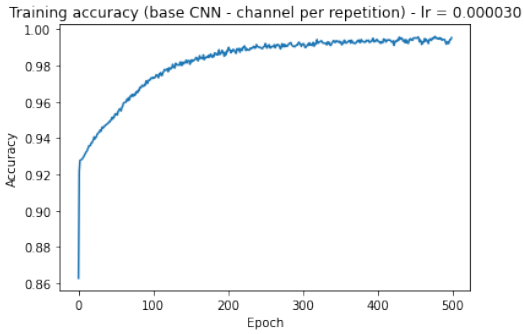
(a) Training loss - with regularization



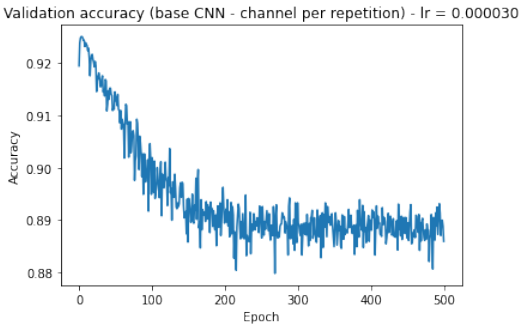
(b) Validation loss - with regularization

Figure 122: Training and validation loss CNN_{reps} model - with regularization (MSE loss)

Accuracy



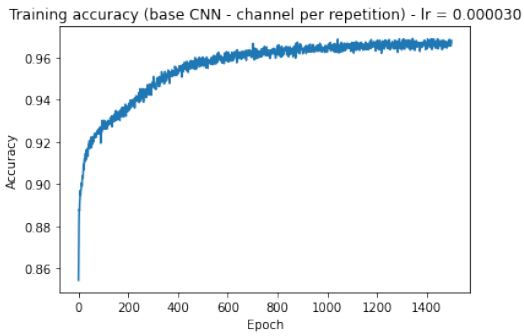
(a) Training accuracy



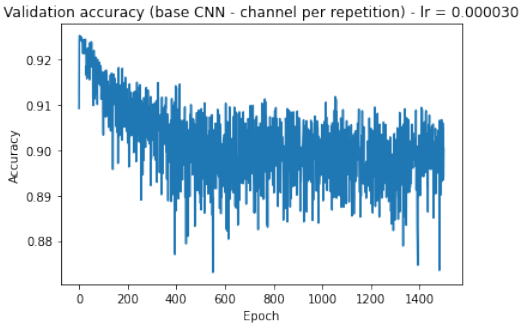
(b) Validation accuracy

Figure 123: Training and validation accuracy CNN_{reps} model (MSE loss)

Accuracy - with regularization



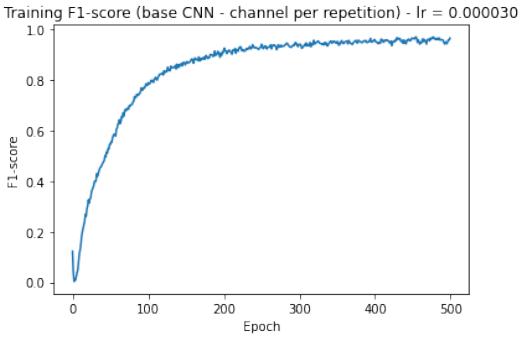
(a) Training accuracy - with regularization



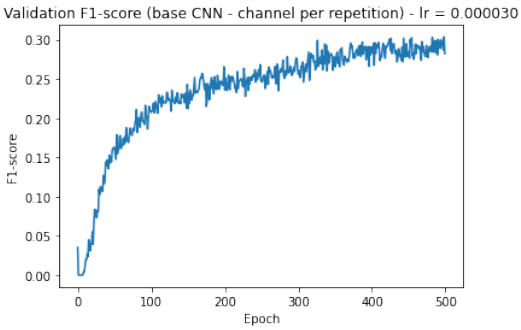
(b) Validation accuracy - with regularization

Figure 124: Training and validation accuracy CNN_{reps} model - with regularization (MSE loss)

F1-score



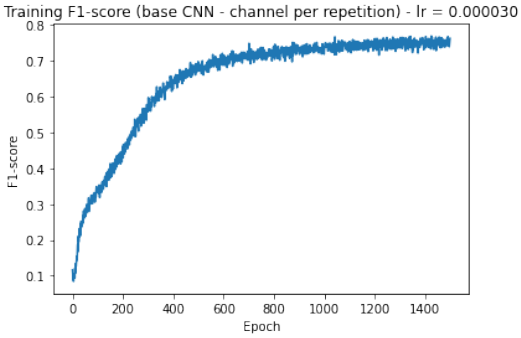
(a) Training F1-score



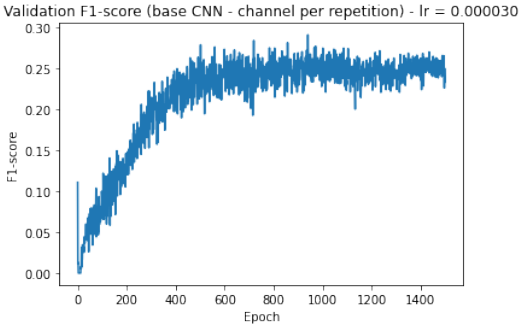
(b) Validation F1-score

Figure 125: Training and validation F1-score CNN_{reps} model (MSE loss)

F1-score - with regularization



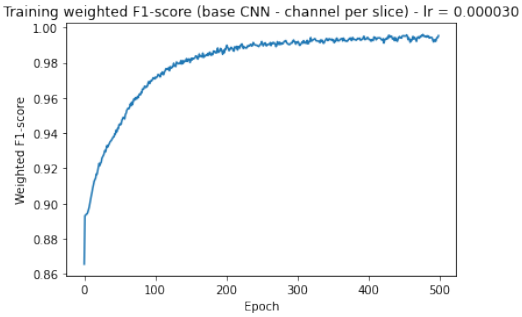
(a) Training F1-score - with regularization



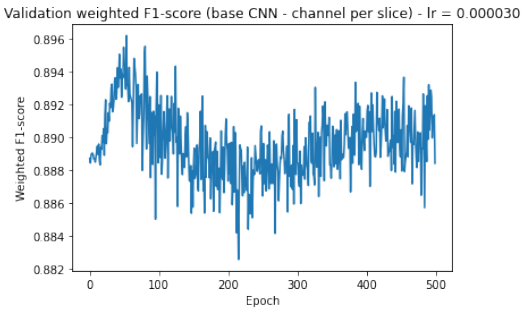
(b) Validation F1-score - with regularization

Figure 126: Training and validation F1-score CNN_{reps} model - with regularization (MSE loss)

Weighted F1-score



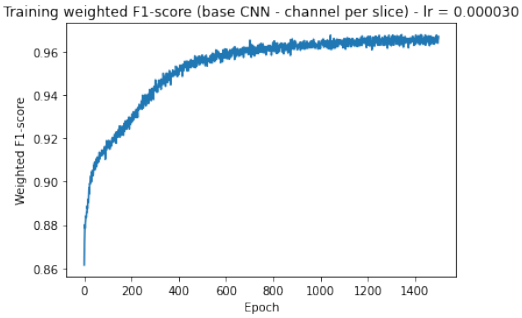
(a) Training weighted F1-score



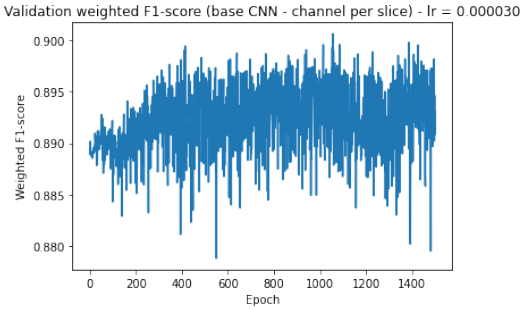
(b) Validation weighted F1-score

Figure 127: Training and validation weighted F1-score CNN_{reps} model (MSE loss)

Weighted F1-score - with regularization



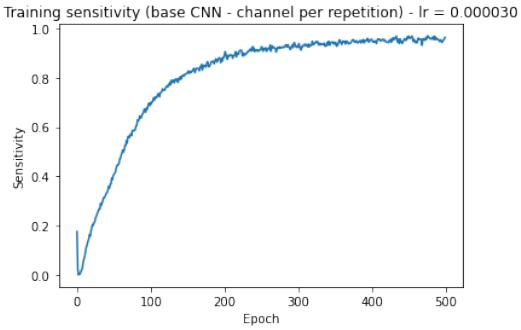
(a) Training weighted F1-score - with regularization



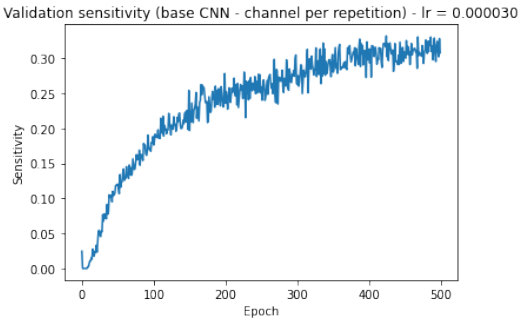
(b) Validation weighted F1-score - with regularization

Figure 128: Training and validation weighted F1-score CNN_{reps} model - with regularization (MSE loss)

Sensitivity



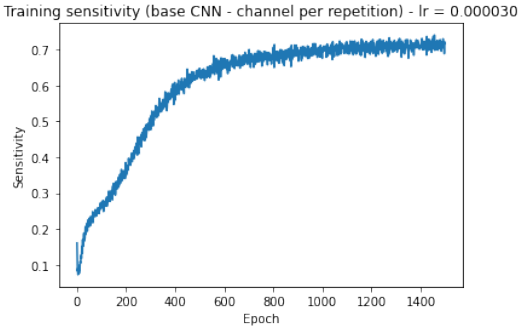
(a) Training sensitivity



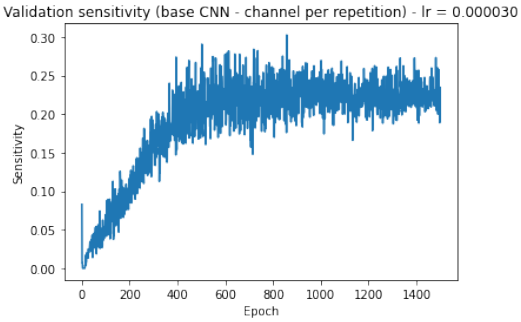
(b) Validation sensitivity

Figure 129: Training and validation sensitivity CNN_{reps} model (MSE loss)

Sensitivity - with regularization



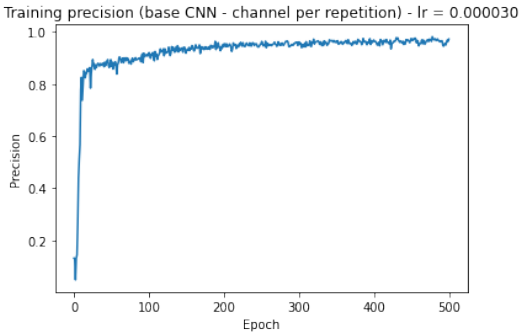
(a) Training sensitivity - with regularization



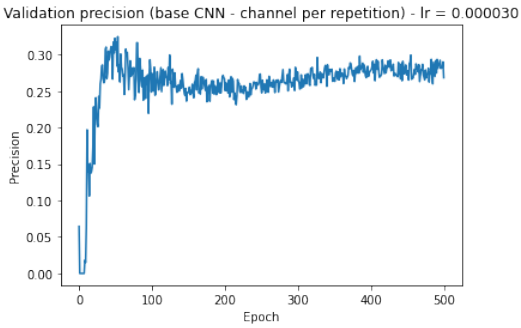
(b) Validation sensitivity - with regularization

Figure 130: Training and validation sensitivity CNN_{reps} model - with regularization (MSE loss)

Precision



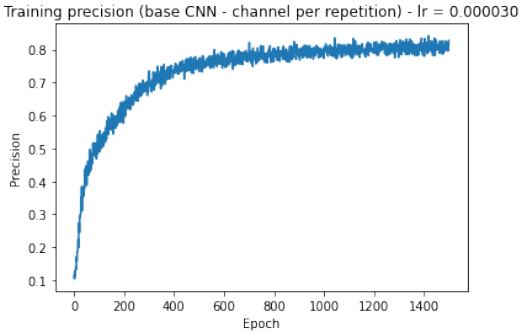
(a) Training precision



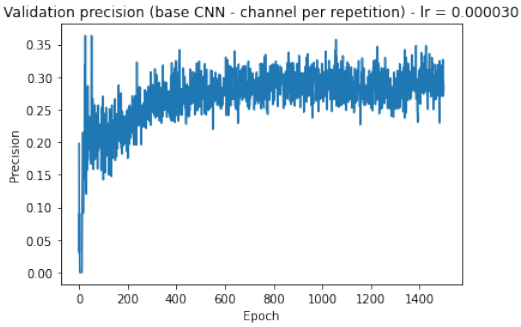
(b) Validation precision

Figure 131: Training and validation precision CNN_{reps} model (MSE loss)

Precision - with regularization



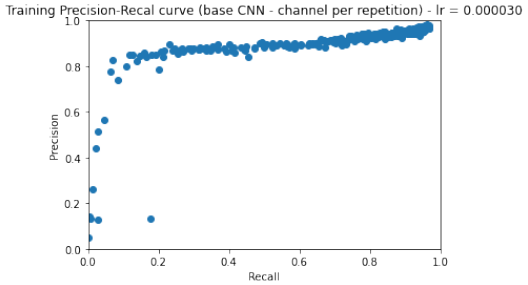
(a) Training precision - with regularization



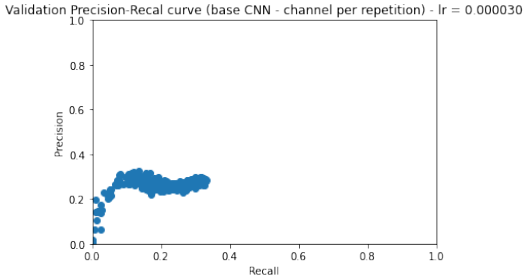
(b) Validation precision - with regularization

Figure 132: Training and validation precision CNN_{reps} model - with regularization (MSE loss)

Precision vs sensitivity



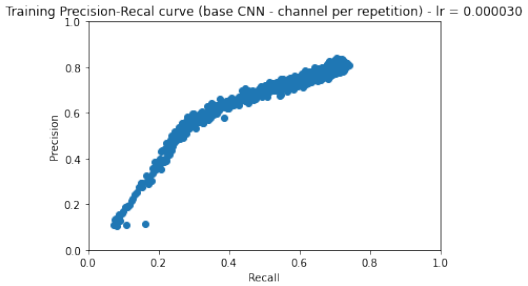
(a) Training precision vs sensitivity



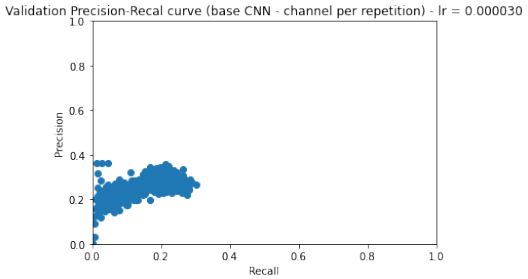
(b) Validation precision vs sensitivity

Figure 133: Training and validation precision vs sensitivity CNN_{reps} model (MSE loss)

Precision vs sensitivity - with regularization



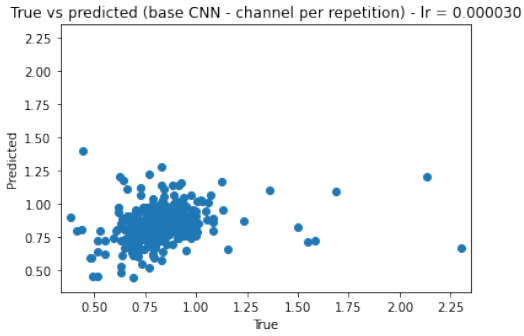
(a) Training precision vs sensitivity - with regularization



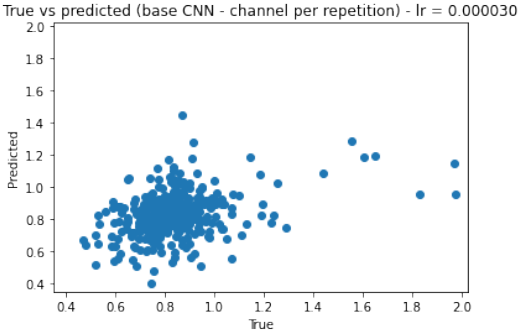
(b) Validation precision vs sensitivity - with regularization

Figure 134: Training and validation precision vs sensitivity CNN_{reps} model - with regularization (MSE loss)

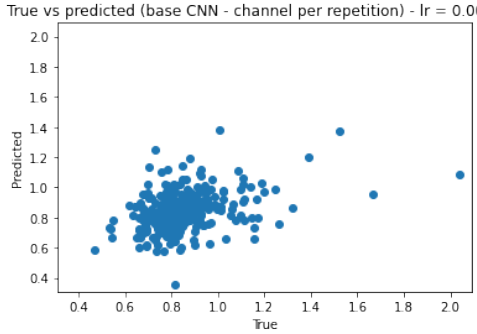
True vs predicted



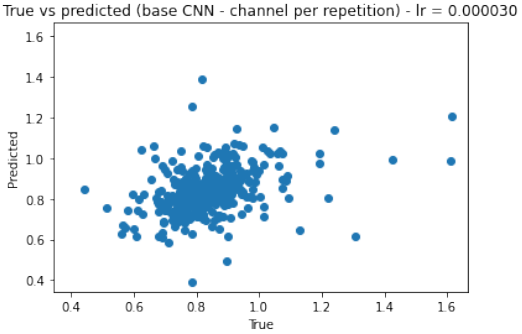
(a) True vs predicted for patient 1



(b) True vs predicted for patient 2



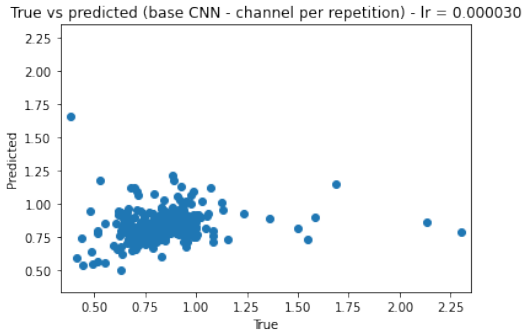
(c) True vs predicted for patient 3



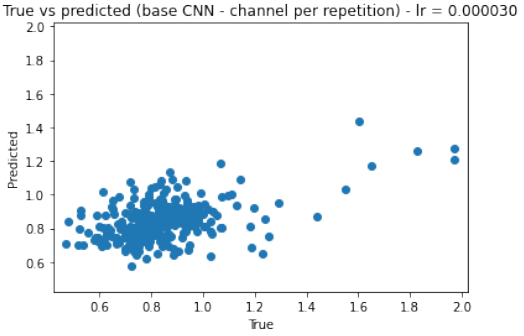
(d) True vs predicted for patient 4

Figure 135: True values vs predicted values for all 4 validation patients CNN_{reps} model (MSE loss)

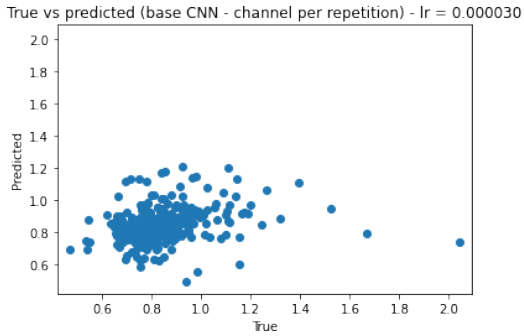
True vs predicted - with regularization



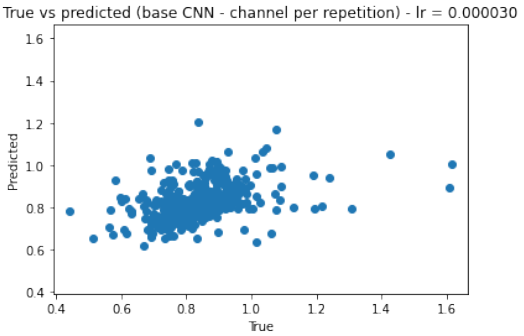
(a) True vs predicted for patient 1 - with regularization



(b) True vs predicted for patient 2 - with regularization



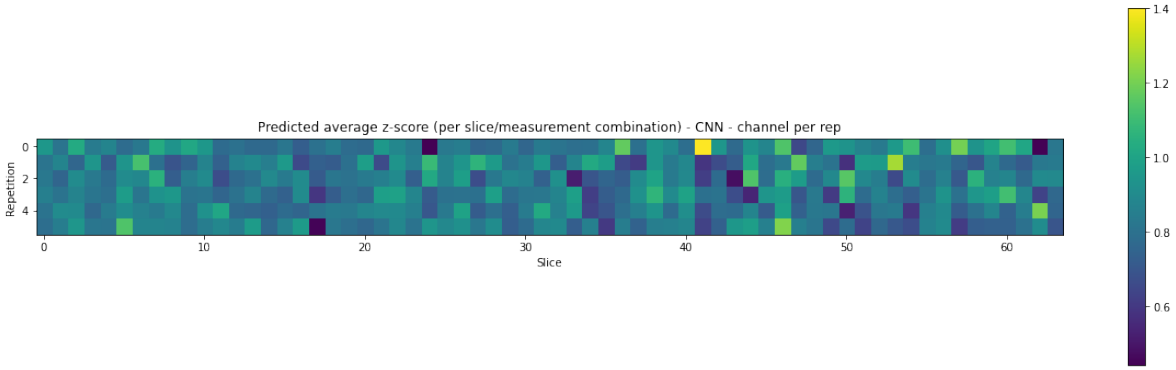
(c) True vs predicted for patient 3 - with regularization



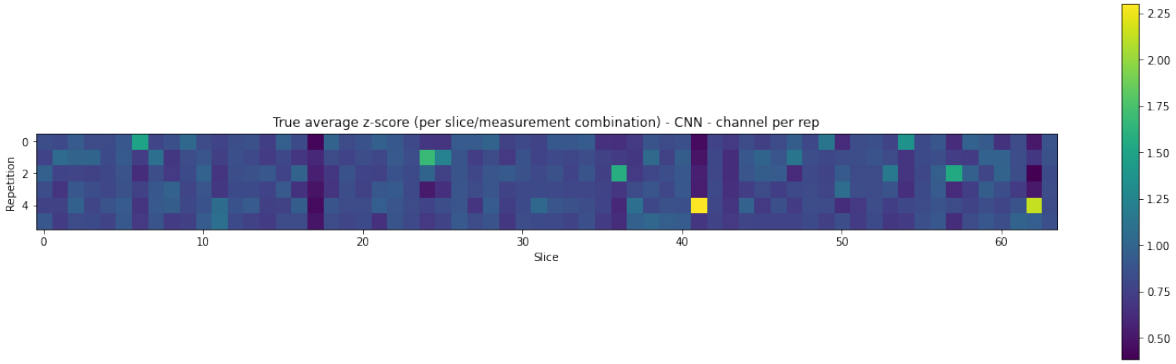
(d) True vs predicted for patient 4 - with regularization

Figure 136: True values vs predicted values for all 4 validation patients CNN_{reps} model - with regularization (MSE loss)

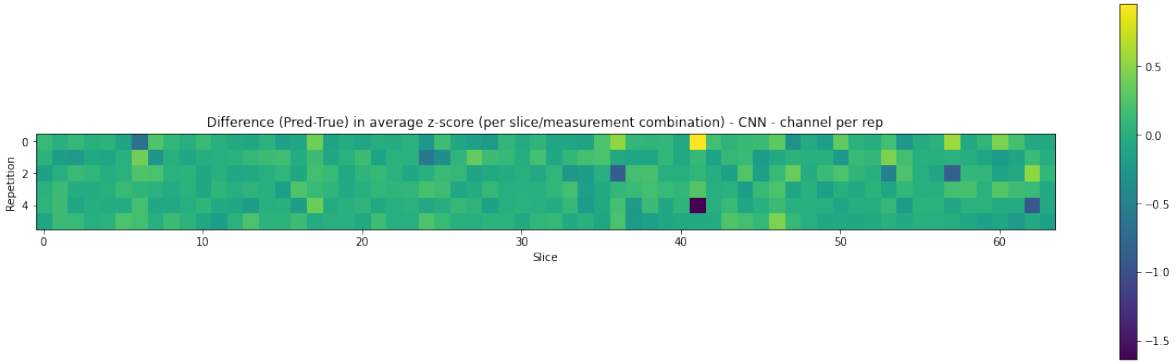
Inspect outputs



(a) Predicted mean Z-scores - test batch 1

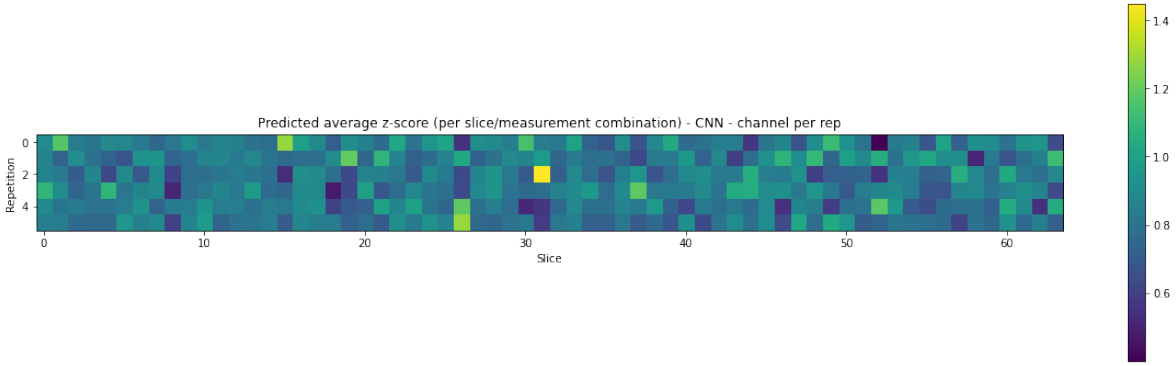


(b) True mean Z-scores - test batch 1

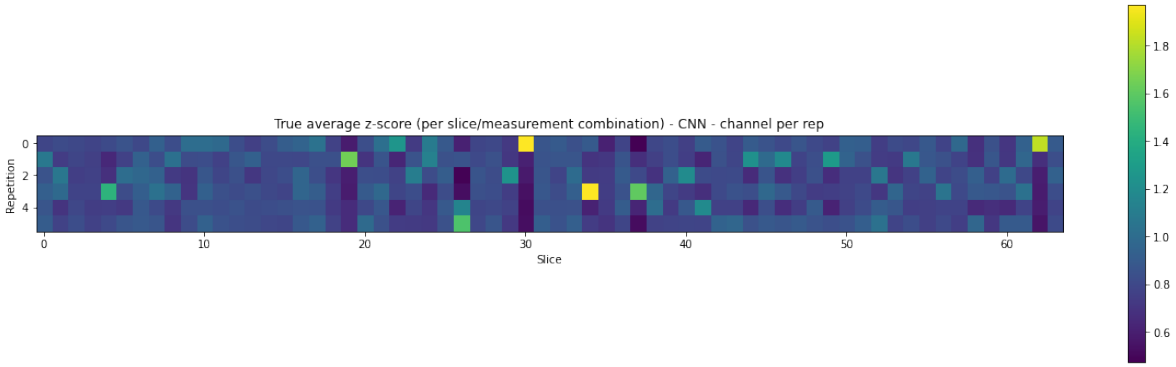


(c) Difference (Pred - True) in mean Z-scores - test batch 1

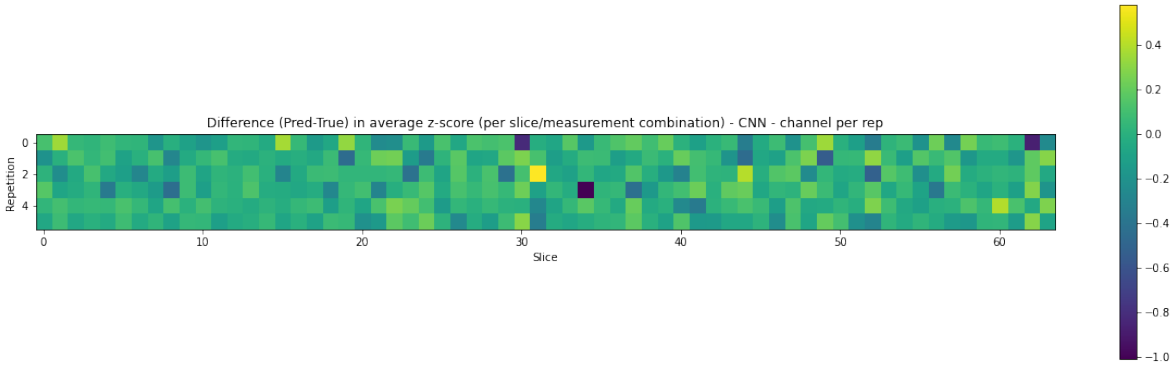
Figure 137: Predicted, true and difference in mean Z-score per slice/timepoint pair for test batch 1 - CNN$_{reps}$ model (MSE loss)



(a) Predicted mean Z-scores - test batch 2

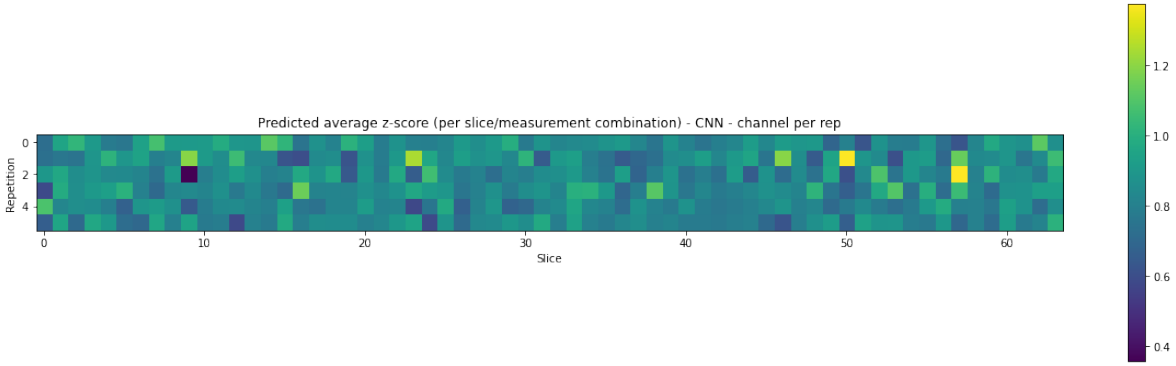


(b) True mean Z-scores - test batch 2

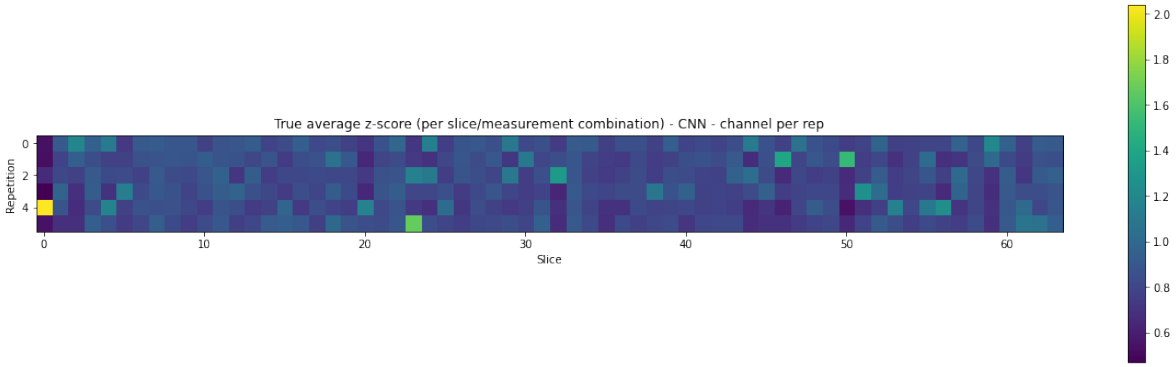


(c) Difference (Pred - True) in mean Z-scores - test batch 2

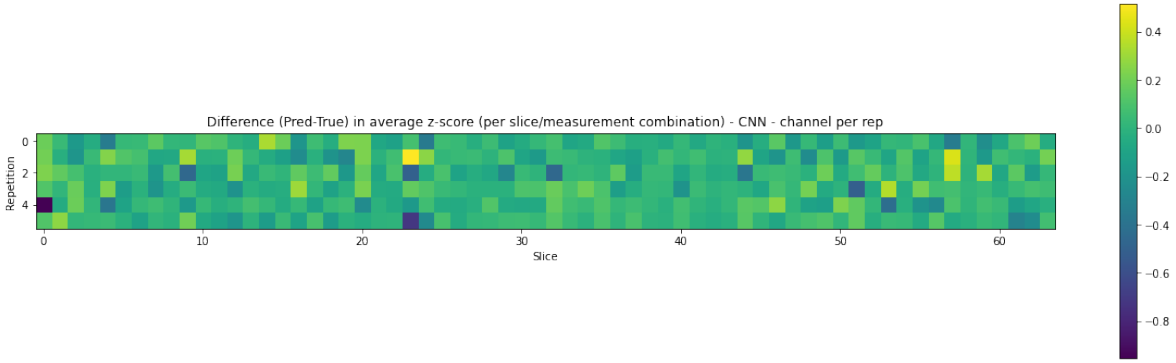
Figure 138: Predicted, true and difference in mean Z-score per slice/timepoint pair for test batch 2 - CNN$_{reps}$ model (MSE loss)



(a) Predicted mean Z-scores - test batch 3

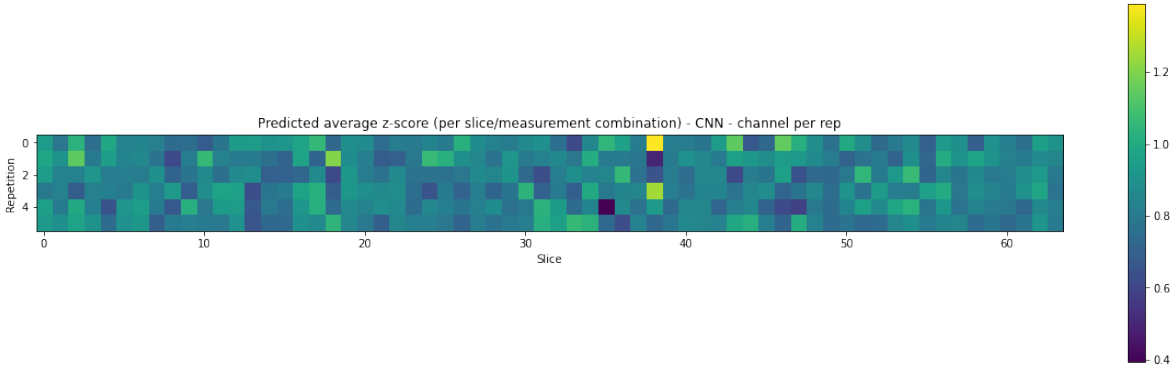


(b) True mean Z-scores - test batch 3

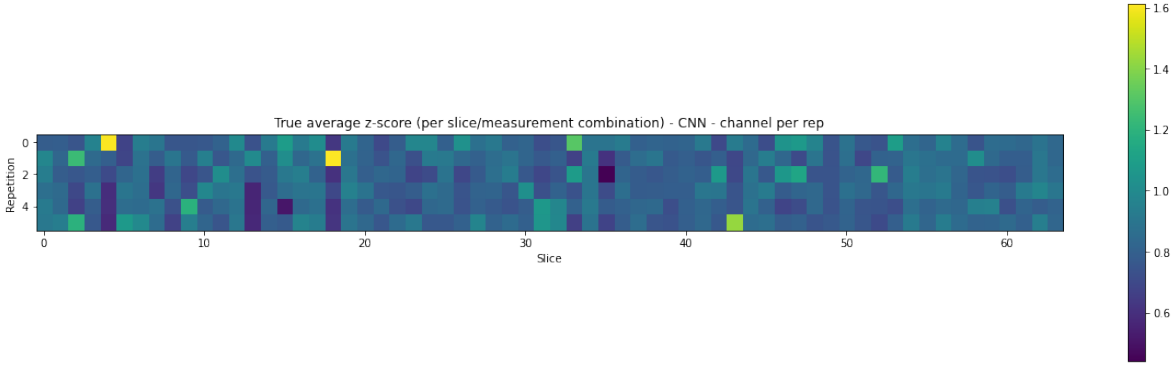


(c) Difference (Pred - True) in mean Z-scores - test batch 3

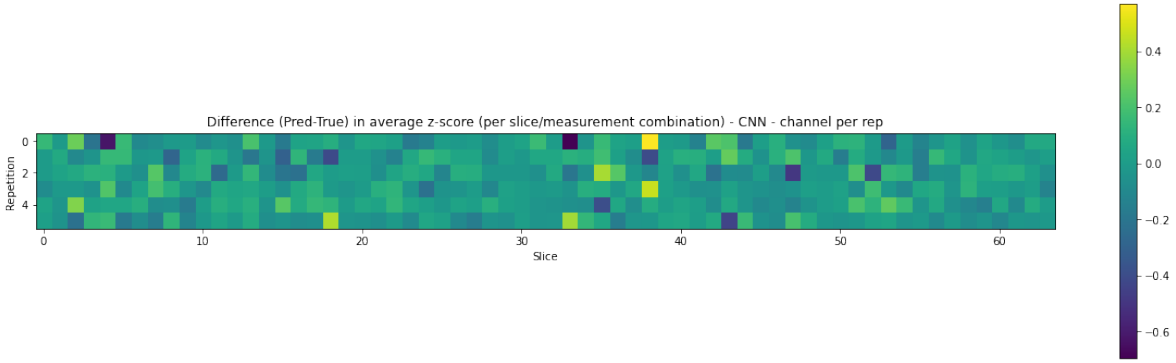
Figure 139: Predicted, true and difference in mean Z-score per slice/timepoint pair for test batch 3 - CNN$_{reps}$ model (MSE loss)



(a) Predicted mean Z-scores - test batch 4



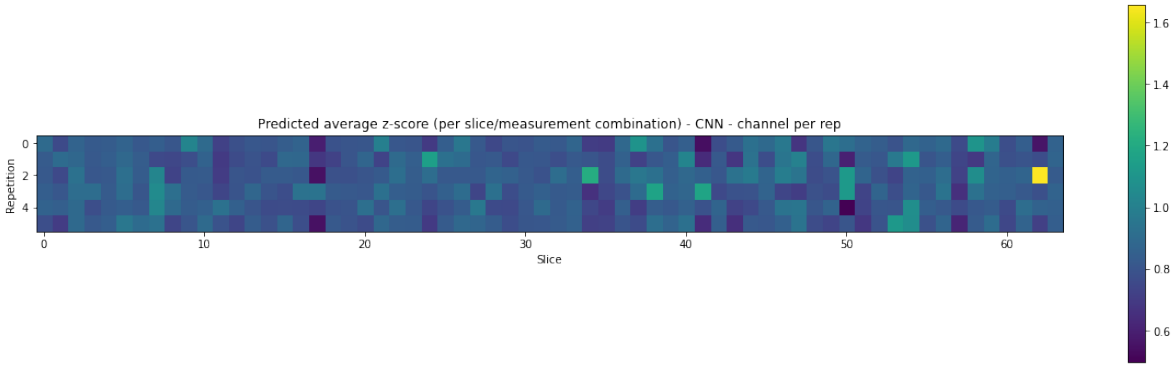
(b) True mean Z-scores - test batch 4



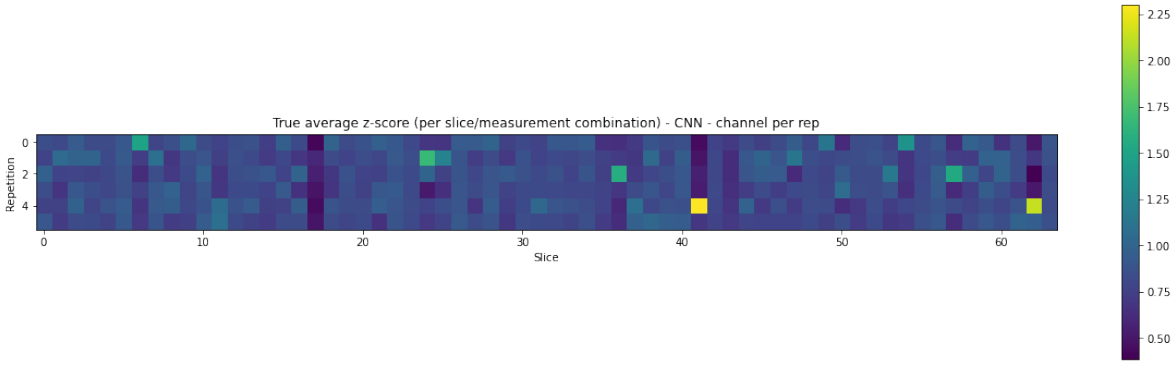
(c) Difference (Pred - True) in mean Z-scores - test batch 4

Figure 140: Predicted, true and difference in mean Z-score per slice/timepoint pair for test batch 4 - CNN_{reps} model (MSE loss)

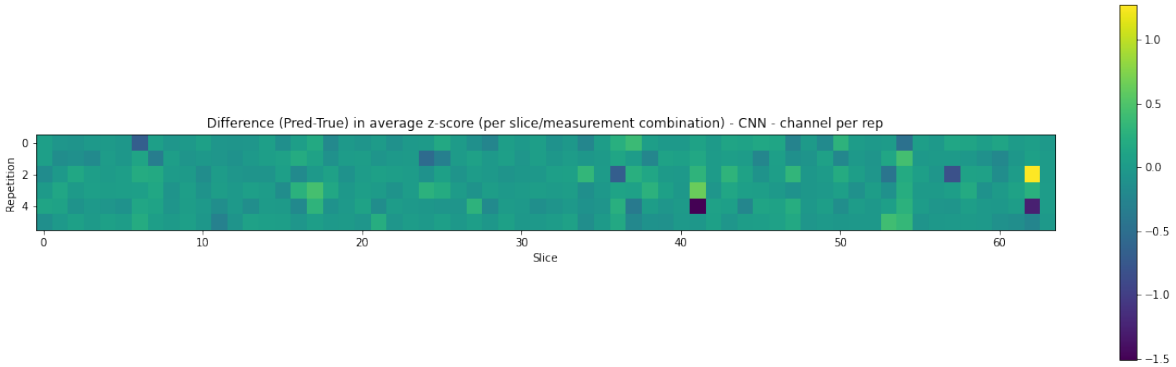
Inspect outputs - with regularization



(a) Predicted mean Z-scores - with regularization - test batch 1

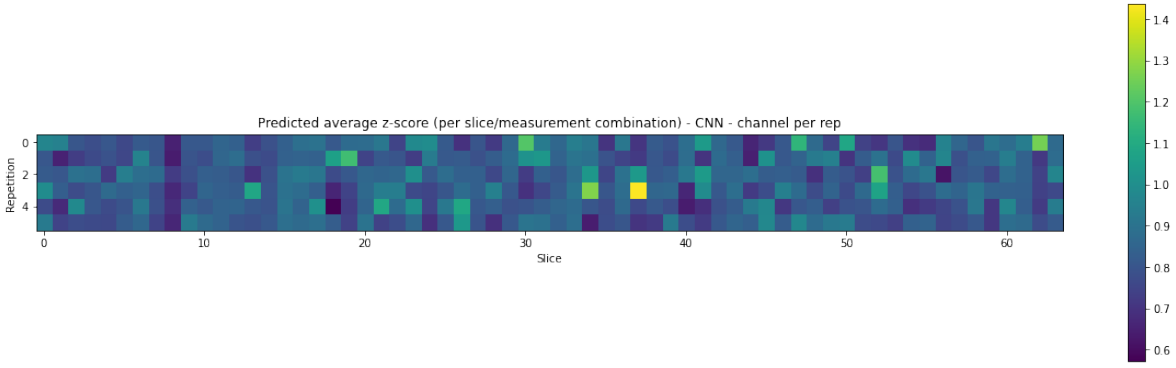


(b) True mean Z-scores - with regularization - test batch 1

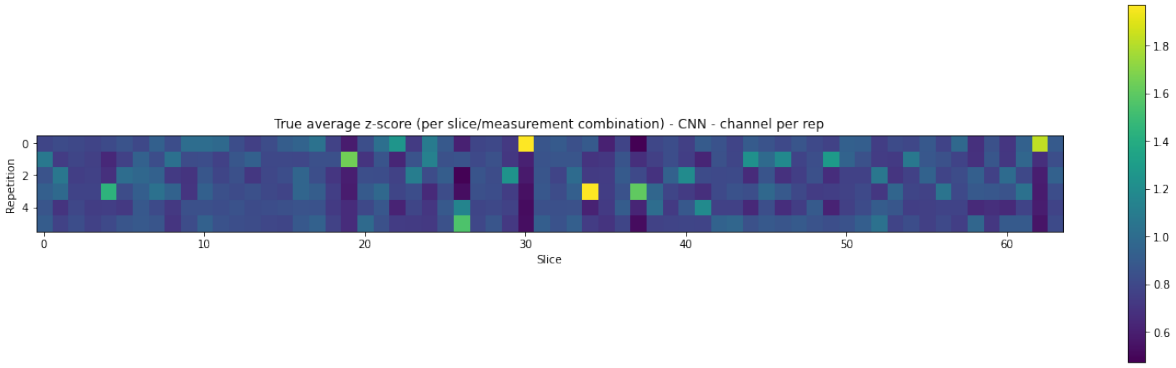


(c) Difference (Pred - True) in mean Z-scores - with regularization - test batch 1

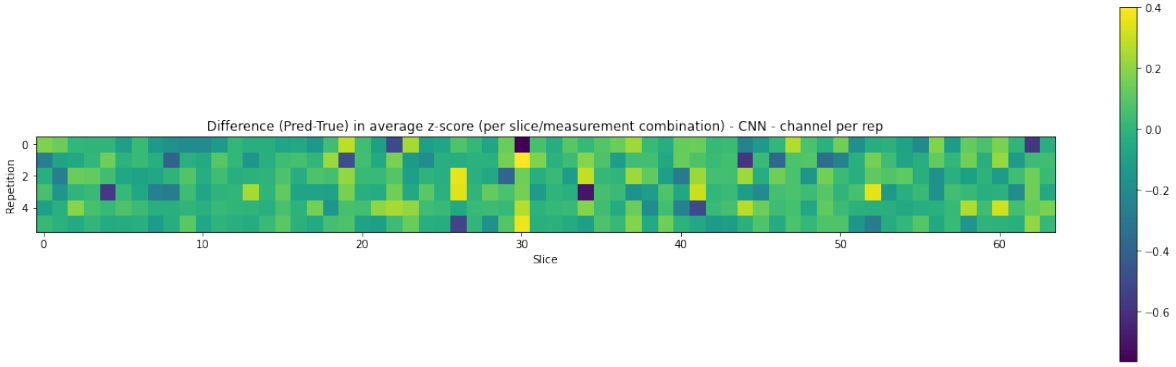
Figure 141: Predicted, true and difference in mean Z-score per slice/timepoint pair for test batch 1 - CNN_{reps} model - with regularization (MSE loss)



(a) Predicted mean Z-scores - with regularization - test batch 2

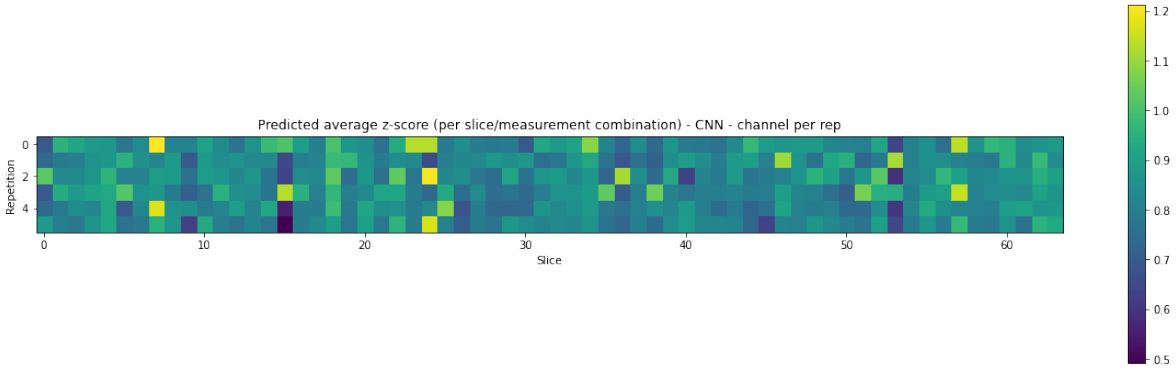


(b) True mean Z-scores - with regularization - test batch 2

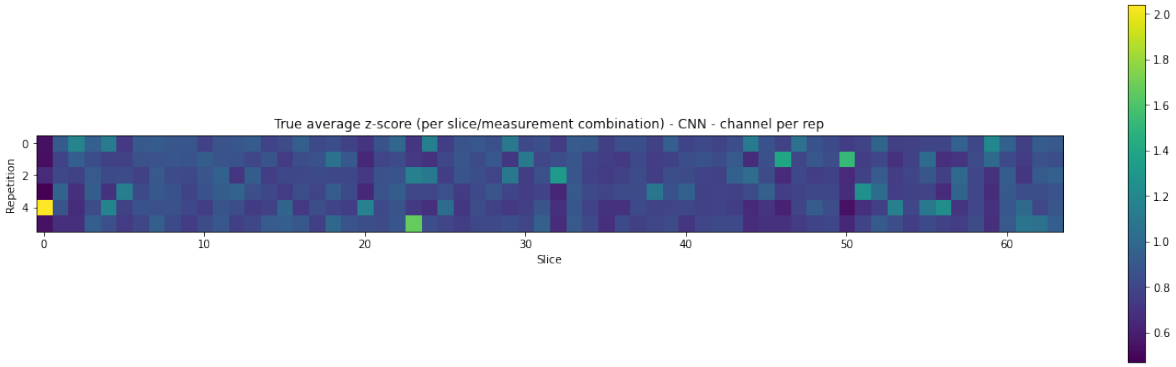


(c) Difference (Pred - True) in mean Z-scores - with regularization - test batch 2

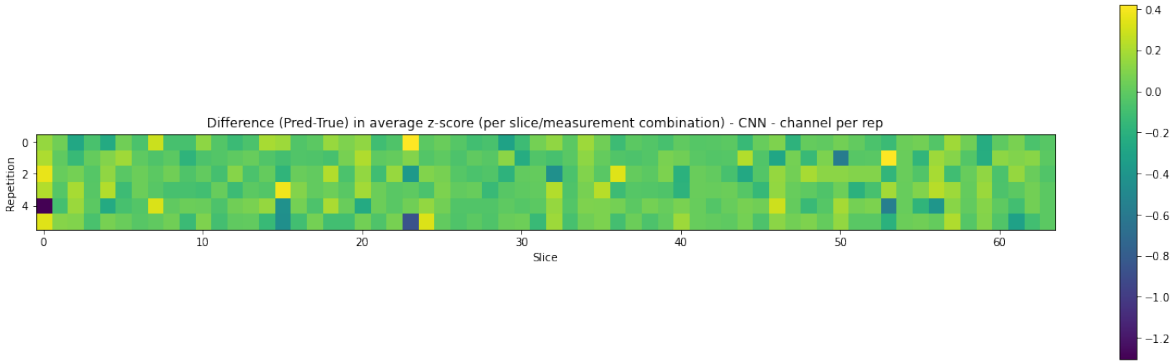
Figure 142: Predicted, true and difference in mean Z-score per slice/timepoint pair for test batch 2 - CNN_{reps} model - with regularization (MSE loss)



(a) Predicted mean Z-scores - with regularization - test batch 3

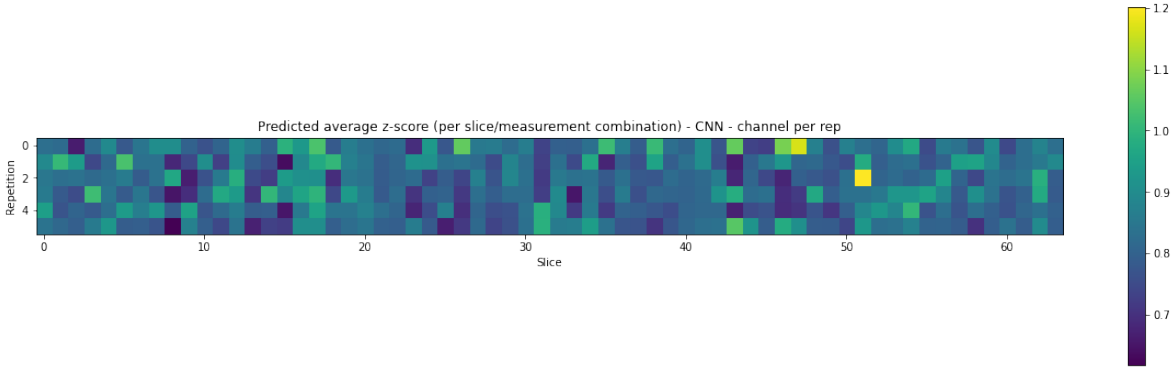


(b) True mean Z-scores - with regularization - test batch 3

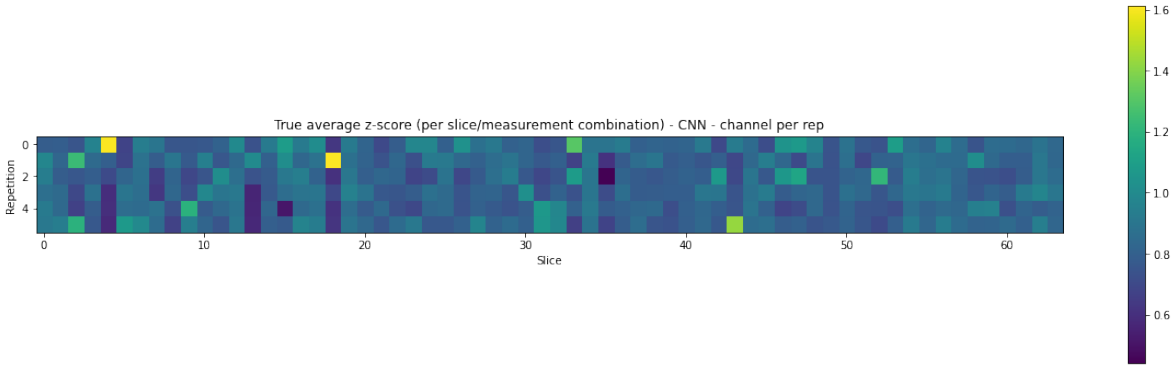


(c) Difference (Pred - True) in mean Z-scores - with regularization - test batch 3

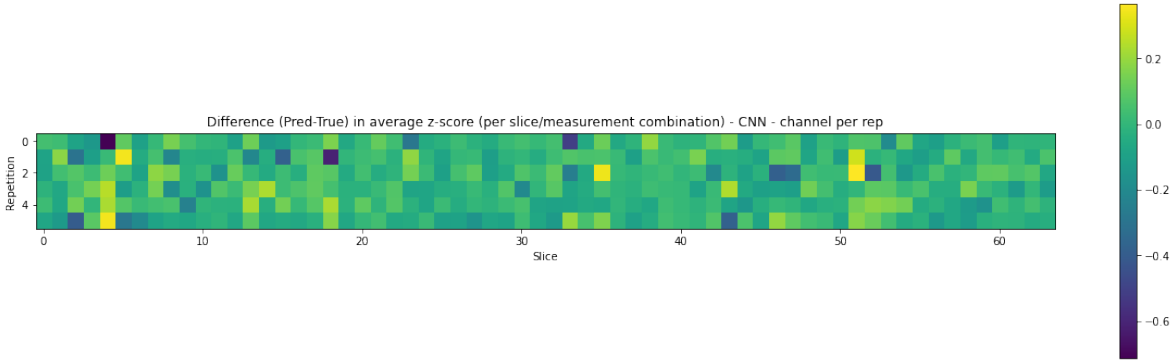
Figure 143: Predicted, true and difference in mean Z-score per slice/timepoint pair for test batch 3 - CNN_{reps} model - with regularization (MSE loss)



(a) Predicted mean Z-scores - with regularization - test batch 4



(b) True mean Z-scores - with regularization - test batch 4



(c) Difference (Pred - True) in mean Z-scores - with regularization - test batch 4

Figure 144: Predicted, true and difference in mean Z-score per slice/timepoint pair for test batch 4 - CNN$_{reps}$ model - with regularization (MSE loss)