Christian Tinauer

Explainable Deep Learning of Multi-parametric MRI Data for Alzheimer's Disease Classification

Master's Thesis

to achieve the university degree of

Master of Science

submitted to

Graz University of Technology

Supervisors

Assoc.-Prof. Dipl.-Ing. Dr. techn. Christian Langkammer

Dipl.-Ing. Dr. techn. Stefan Heber

Evaluator

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

Institute of Medical Engineering

Faculty of Computer Science and Biomedical Engineering

Graz, May 2020

Affidavit

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

Date

Signature

Abstract

Alzheimer's disease is the most common form of dementia and a major challenge for healthcare systems. Alongside clinical tests, magnetic resonance imaging shows promise to aid in the diagnostic process. Recent advances in computing power made processing of MR images using deep learning models feasible. The analysis of contemporary classification approaches showed that they mostly rely on structural MRI data acquired with MPRAGE sequences. In this thesis the applicability of additional image contrasts (FLAIR, R2, R2*, MTR) for Alzheimer's disease classification with deep convolutional neural networks is considered and compared. Furthermore, the explanation method "deep Taylor decomposition" indicated that those networks might learn features introduced by the applied image preprocessing. Therefore, this thesis introduces a new method to mitigate these problems by using the generated heatmaps during training for regularization. The proposed method leads to identification of features in anatomically more plausible regions, while maintaining a similar classification accuracy compared to contemporary approaches.

Key words: Alzheimer's disease, deep learning, explainability, relevance-guided training

Zusammenfassung

Die Alzheimer-Krankheit ist die häufigste Form der Demenz und eine wesentliche Herausforderung für Gesundheitssysteme. Neben klinischen Tests wird die Magnetresonanztomographie für den Diagnoseprozess genutzt. Mit den jüngsten Fortschritten in Rechenleistung wurde die Verarbeitung von MR-Bildern mit Deep Learning Modellen ermöglicht. Zeitgemäße Klassifikationsansätzen verwenden hauptsächlich mit MPRAGE-Sequenzen akquirierte, strukturelle MRT-Daten. In dieser Arbeit wird die Anwendbarkeit von zusätzlichen Bildkontrasten (FLAIR, R2, R2*, MTR) für die Klassifikation der Alzheimer-Krankheit mit tiefen neuronalen Faltungsnetzwerken betrachtet und verglichen. Darüber hinaus hat die Methode "deep Taylor decomposition" gezeigt, dass die Bildvorverarbeitungsschritte möglicherweise ungewollte Merkmale erzeugen, die von diesen Netzwerken für die Klassifikation genutzt werden. Um die Problem abzuschwächen führt diese Arbeit eine neue Methode ein, um die erzeugten Heatmaps für die Regularisierung während des Trainingsprozesses zu ermöglichen. Die vorgeschlagene Methode identifiziert anatomisch plausiblere Regionen, während eine ähnliche Klassifizierungsgenauigkeit verglichen mit zeitgemäßen Ansätzen sichergestellt wird.

Schlüsselwörter: Alzheimer-Krankheit, Deep Learning, Erklärbarkeit, relevanz-gesteuertes Training

Acknowledgements

This page is dedicated to all people who supported and encouraged me during the time I have been working on this thesis.

First of all I want to give my gratitudes to Assoc.-Prof. Dipl.-Ing. Dr. techn. Christian Langkammer, my supervisor at the Medical University of Graz, who made the thesis environment available to me. Thank you for motivating me in difficult times and engaging in many constructive discussions.

Secondly, I want to thank Dipl.-Ing. Dr. techn. Stefan Heber for discussions and support on deep learning topics and for challenging my ideas.

Furthermore, my gratitudes go to Univ.-Prof. Dipl.-Ing. Dr. techn. Rudolf Stollberger, my supervisor at the Graz University of Technology, for his support and feedbacks.

Special thanks go to my colleagues from the Neuroimaging Research Unit, Medical University of Graz for interesting discussions, which enlightened my working days. Thank you for the fun and the solidarity.

Furthermore, I want to thank my friends. Foremost, my deepest gratitudes to you Astrid.

To my wonderful family, who always believed in me and encouraged me to go on.

Contents

List of Figures

Acronyms and Abbreviations

AD Alzheimer's disease

AI artificial intelligence

API application programming interface

AUC area under curve

CNN convolutional neural network

CSF cerebrospinal fluid

DNN deep neural network

DTI diffusion tensor imaging

FLAIR fluid-attenuated inversion recovery

FLASH fast low-angle shot

GPU graphics computing unit

LRP layer-wise relevance propagation

MCI mild cognitive impairment

MPRAGE magnetization-prepared rapid gradient-echo

MRI magnetic resonance imaging

MR magnetic resonance

MTR magnetization transfer ratio

MT magnetization transfer

NC normal control

PET positron emission tomography

RNN recurrent neural network

ROC receiver operating characteristic

ReLU rectified linear unit

SAE stacked auto-encoder

SA sensitivity analysis

SVM support vector machine

1 Introduction

Alzheimer's disease (AD) is the most common form of dementia [1] and a major challenge for healthcare in the twenty-first century [2]. Worldwide, estimated 50 million people are living with AD or a related form of dementia in 2019 [3, 4]. The disease is most common in Western Europe and North America. In later life, it is among the top causes for disabilities [4]. The most accurate diagnosis is obtained from the histological examination of tissue samples from affected anatomical regions. AD-related staging can for example be done with the Braak system [5, 6]. An overview of the pathological evolution of AD is shown in figure 1. Unfortunatly, biopsy during life is impossible for AD because of the high risk/benefit ratio.

Alongside mental status and neuropsychological tests, magnetic resonance imaging (MRI) is used for diagnosis. Different MRI contrast mechanisms are considered to obtain information about the brain. Structural imaging is among the most utilizied techniques as AD is associated with brain atrophy, the loss of brain tissue [8, 9, 10]. In clinical MRI spin-lattice relaxation time (T1) and spin-spin relaxation time (T2) are the most important biophysical parameters contributing to image contrast, or other quantitative parameters such as T2* describing tissue properties like iron load [11]. Some of these properties reflect the presence or even the severity of AD [12].

In the recent years deep learning techniques have become increasingly utilized in

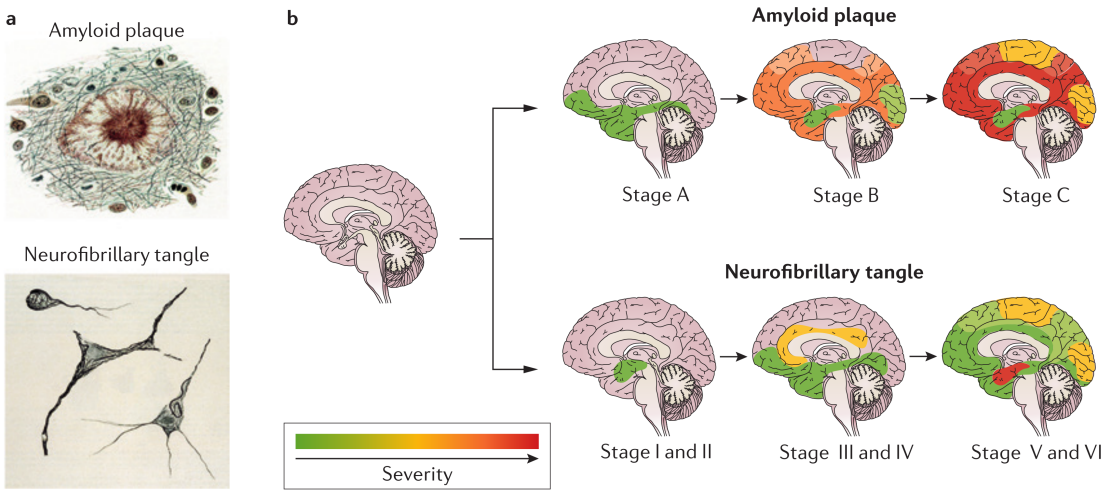


Figure 1: The pathological evolution of Alzheimer's disease. Amyloid plaques and neurofibrillary tangles spread throughout the brain as the disease progresses. Images in (**a**) are using the Bielschowsky method of silver impregnation to visualize the aggregated proteins that constitute the extracellular plaques and intracellular neurofibrillary tangles. In typical cases of Alzheimer's disease (AD), amyloid-β (Aβ) deposition precedes neurofibrillary and neuritic changes with an apparent origin in the frontal and temporal lobes, hippocampus and limbic system (**b**, top row). Less commonly, the disease seems to emerge from other regions of the cerebral neocortex (parietal and occipital lobes) with relative sparing of the hippocampus. The neurofibrillary tangles and neuritic degeneration start in the medial temporal lobes and hippocampus, and progressively spread to other areas of the neocortex (**b**, bottom row). With the advent of molecular imaging techniques for Aβ and tau, the longitudinal dispersal of pathological changes will become amenable to real-time in vivo study and will not be reliant on post-mortem reconstructions as depicted here. Aβ deposition (stages A, B and C) and neurofibrillary tangles (stages I–VI) are adapted from Braak and Braak [6]. Figure 1 and caption from [7].

medical applications, including image reconstruction [13], segmentation [14], and classification [15]. Deep learning methods achieve a high predictive accuracy particularly for classification tasks. In many cases, they are on par with human level performance. For the development and training of deep neural networks (DNNs) data sets with at least a few hundred entries are necessary. Many AD classification models have been developed using only T1-weighted structural images. These structural images are available but may not be sufficient to obtain a reliable classification. Therefore it is necessary to include different MRI contrasts and parameters into the classification model [16]. Contrasts which have proven to contribute insight into AD diagnosis can be explored for applicability with deep learning classification models.

Deep learning image classification models reduce high dimensional inputs (2-D, 3-D or 4-D) to a one dimensional vector. Each entry in the vector represents an output class. Their value relates to the probability of the input belonging to this class. The reduction of the dimensionality is achieved through so called hidden layers in the network. The input image data is put through the weighted connections of these layers until the output layer is reached. A trained model is capable of attributing a classification result to a given input. However, the question how the DNN obtained this output arises?

The good performance of these models comes at the price of explainability. To interpret the connections in a DNNs is beyond human comprehension. This is an issue for medical applications as which parts of the input image contributed to the classification is essential to establish trust in the prediction.

Recently, methods have been developed to explain a classification result in terms of input, forward pass of the input through the network and output [17]. The explanation is obtained as a heatmap with the same dimensionality as the original input. Regions in the input image that contributed most to the classification result are represented by high values in the heatmap. These methods visualize what the DNN focuses on.

2 Related Work

The application of automated image classification for AD has recently gained considerable attention as large-scale multi-modal neuroimaging data became available. Supported by the significant increase in computing power and the general availability of massive parallel computing capabilities coming from graphics computing units (GPUs), it became feasible to train deep and wide neural networks.

Studies focusing on the utilization of neuroimaging for AD and mild cognitive impairment (MCI) classification, [18] used these methods on the modalities structural MRI, functional MRI, diffusion tensor imaging (DTI) and fluorodeoxyglucose positron emission tomography (PET). They reviewed publications from January 1985 until June 2016 and found high accuracies discriminating between AD and normal control (NC). Performance was higher when multiple input modalities were combined but generalization and reproducability remain unsolved problems. Reviewed methods in [18] were also sub-categorized in terms of extracted features as a post-processing step of the modalities. However, an explanation on a per input basis for the reviewed methods is not given. Therefore it remains unclear if the high classification accuracies result from clinical useful features learned from the used data. Most of the reviewed methods in [18] utilize support vector machines (SVMs). More recent methods are based on deep learning such as stacked auto-encoders (SAEs), convolutional neural networks (CNNs) or combinations of them.

In [16] the authors reviewed diagnostic AD classification approaches between 2013 and 2018 using deep learning and reported highly accurate results on their test sets. They reported the finding that combining multimodal inputs can improve performance but reproducability of the results outside of the development environment remains an issue. One of the reasons is the limited size of the data sets, which stays below the hundreds for all reviewed publications. To account for that, the review suggests that the architecture of the used neural network should be adapted to enable transfer learning [19] and explainability [20]. All of the reviewed approaches use (deep) CNNs, which are widely used in computer vision. These methods do not require human intervention like handcrafted feature engineering but instead extract features directly from the given training data. However, this aspect of deep learning necessarily brings uncertainty over which features would be learned. Transparency in this sense is a remaining challenge.

Explanations can take many forms and because of the high complexity and dimensionsality of the CNNs, evaluating the quality of explanations or the interpretability of a model is difficult. Within the machine learning community different models for explanations have been proposed. Perturbation-based methods change the input and check if the output changes accordingly [21, 22]. For example if the classifier shall tell us whether there is a cat in an image with a cat, the cat is occluded from the image and the output should change. When the output stays the same the model may has picked up features not suitable to find cats. Still these methods do not tell anything about the learned features. Additionally, they are slow and may introduce artefacts coming from the perturbations. The second group is function-based [23] and gives explanations by using the derivative (gradient) of the function defined by the machine learning model

$$\nabla_x f(\boldsymbol{x}), \tag{1}$$

where ∇_x is the gradient with respect to the input image. Resulting sensitivity maps tend to be noisy because the linearization may be a oversimplification and does not

capture the non-linear nature of the function f well enough. The third group tries to overcome these weaknesses by adding noise and averaging results [24] or by training and presenting local sparse models of how predictions change when inputs are perturbed [25]. The idea of decomposing the function f into subfunctions, which can be more easily explained, forms the fourth group, layer-wise relevance propagation (LRP) [26]. Deep Taylor decomposition [27] describes the theoretical groundwork for LRP and can be implemented in a fast way.

3 Motivation

The goal of this work was to develop deep learning classification networks using structural T1-weighted magnetization-prepared rapid gradient-echo (MPRAGE) images, fluid-attenuated inversion recovery (FLAIR) images, R2 images, R2* images and magnetization transfer ratio (MTR) images. Furthermore, the MRI contrasts/parameters which are best suited for this task have been investigated. With the explainability method deep Taylor decomposition [27] the learned features of the different inputs are determined. Another goal of this work was to explore the effects of skull stripping (brain extraction).

The experiments on AD classification based on MRI data showed that DNNs might learn irrelevant features outside the brain or only learn to evaluate the quality of the used brain extraction algorithm. Concerned by this finding, the MRI features relevant for the classification of patients with AD when compared to NC have been examined.

Additionally, a method that allows to add prior knowledge to the training process is propossed. It enables focusing the training on relevant features. This helps the trained model to get invariant to certain data preprocessing steps.

4 Background

For the classification tasks executed for this work supervised learning methods have been used. Supervised learning is the task of finding a function that maps an input to an output based on example input-output pairs. A labeled data set is necessary for the learning task. The most popular supervised learning methods are deep neural networks. The first part of this chapter summarizes the theoretical background of what deep learning is and how the used networks are designed and trained. The second part focuses on the explainability methods implemented to get insight into the obtained classification outputs.

4.1 Deep Learning

Formal tasks, that is a set of rules and instructions, are amongst the easiest problems to solve for a computer but are among the most difficult undertakings for human beings. In an environment where such rules can be easily inferred, an algorithm is available, computers have a long history of being superior. In contrast, tasks which are easy to perform for people but are hard to describe formally, like recognizing words or faces in images, need a different approach. The idea is to allow computers to learn from experience and understand the task in terms of a hierarchy of concepts. Each concept is defined through its relation to simpler concepts. This solution avoids the need for

formally specifying all the knowledge that the computer needs by rather gathering the knowledge from experience. In deep learning, complicated concepts are learned by building them out of simpler ones built on top of each other. The concepts are represented by layers. A deep network is created by multiple connected layers.

These systems need the ability to acquire their own knowledge by extracting patterns from raw data. Corresponding methods are are referred to as machine learning based methods. Raw data may also be processed to support the learning task.

The inner workings of deep learning networks are described in the subsequent sections based on [28].

4.1.1 Single-layer Networks

Deep learning networks are constructed from layers. The layers describe their behavior and ability to capture the concepts. A layer has an input and an output, whereas the output of a layer becomes the input of the following layer. The first layer of a network is the data itself and the last layer is the result. All layers in between are called hidden layers, simply because they are inside the network and not directly accessible.

The input and output of a layer can be described as vectors, where the length of the output vector depends on the type of the layer. For fully connected layers the transformation from input to output is done with two linear operations followed by a non-linear function. The linear operations are

$$\boldsymbol{z} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}, \tag{2}$$

where $\boldsymbol{z}$ is the intermediate result, $\boldsymbol{x}$ is the input vector of the layer, $\boldsymbol{W}$ is the weighting matrix connecting input to output and $\boldsymbol{b}$ is the bias. The non-linear function a used

point-wise on the intermediate result $\boldsymbol{z}$ gives the output of the layer:

$$\boldsymbol{y} = a(\boldsymbol{z}) = a(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}). \tag{3}$$

Breaking down the matrix operations of equation 2 to the level of a single unit, the neuron (therefore the name neural network), yields

$$z_j = \sum_i w_{ij} x_i + b_j, \tag{4}$$

where the weighted sum of all input entries x_i contributing to the intermediate output z_j is calculated. The output bias b_j can be viewed as the threshold that must be reached by the sum to get the neuron to fire, depending on the used activation function

$$\sum_i w_{ij} x_i \geq t_j, \tag{5}$$

where t_j is accounting for the threshold. The applied activation function is another important aspect. Different non-linear activation functions have been proposed in the literature. Most famous the sigmoid activation function

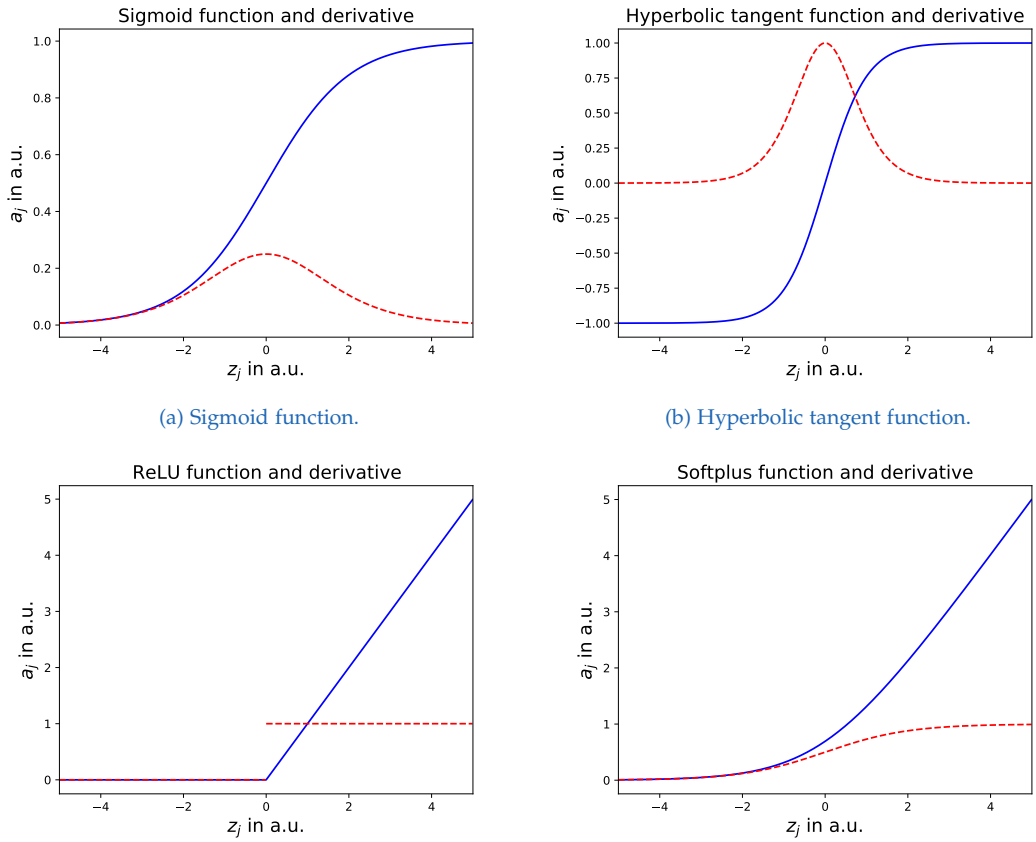
$$a(z_j) = \frac{1}{1 + e^{-z_j}}, \tag{6}$$

shown in figure 2a and its more practical counterpart the hyperbolic tangent function in figure 2b. Both functions have the drawback of function values not changing much when changes in z_j are big but z_j is far away from 0, which is visualized by the values of the derivates in the figures. This phenomena is called activation function saturation and recovering from it can significantly influence the training process.

A simple and now widely used alternative is the rectified linear unit (ReLU) activation function [29]

$$a(z_j) = \begin{cases} z_j & z_j > 0 \\ 0 & z_j \leq 0 \end{cases}, \tag{7}$$

shown in figure 2c. A smooth approximation is the softplus function [30] in figure 2d.



(a) Sigmoid function.

(b) Hyperbolic tangent function.

(c) Rectified linear unit function.

(d) Softplus function.

Figure 2: Overview of different activation functions (blue) and their corresponding derivates (dashed red).

Although the ReLU function is non-linear altogether, for positive z_j a linear response is given. This function does not suffer from the saturation problem and the linear part can be helpful for explainability. During the training process the weights and biases are adjusted to obtain outputs closer to the wanted outputs. Activation functions keep the output of a layer in a well defined range and therefore play a stabilizing role as even small changes to weights or biases could completly change the output otherwise.

To update the weights and biases in a meaningful way, a criterion is needed. This criterion is called objective function and should be suitable for the task at hand. It measures the distance between the output for a given input and the desired output. The goal is to minimize this distance, which makes it an optimization problem of the form

$$\mathbf{\Phi}^* = \text{argmin } c(\mathbf{\Phi}), \tag{8}$$

where $\mathbf{\Phi}^*$ denotes the parameters that minimize the objective function c. The function c is also referred to as cost function, loss function or error function. In this equation $\mathbf{\Phi}$ summarizes all weights and biases being optimized. As the count of variabels in $\mathbf{\Phi}$ is usually very high (up to millions in some network architectures), no analytical solution to the problem can be given. Instead a gradient descent based method [31] is utilized to calculate a numerical solution. In case of simple gradient descent the parameter update formula is

$$\Delta\mathbf{\Phi} = -\eta\nabla_{\mathbf{\Phi}}c, \tag{9}$$

where $\eta > 0$ is the step size, called learning rate in the context of deep learning, and $\nabla_{\mathbf{\Phi}}$ is the gradient of function c with respect to the parameters $\mathbf{\Phi}$. The updates

$$\mathbf{\Phi}^{(i+1)} = \mathbf{\Phi}^{(i)} - \eta\nabla_{\mathbf{\Phi}}c \tag{10}$$

are repeatingly calculated until a local minimum of the function c is reached. This will improve the network output based on the given objective function. On one hand with

this update rule only one example would be considered at a time causing non-smooth convergence, but on the other hand considering all data in the training set at once could lead to getting stuck in a local minimum. To deal with this circumstances, only a part of the training set, a mini-batch with a defined size m, is used at once, yielding the stochastic gradient descent update

$$\mathbf{\Phi}^{(i+1)} = \mathbf{\Phi}^{(i)} - \frac{\eta}{m} \sum_{j}^{m} \nabla_{\mathbf{\Phi}} c(\boldsymbol{x}_j), \tag{11}$$

where $\boldsymbol{x}_j$ is a input in the mini-batch. To make these ideas more precise, stochastic gradient descent works by randomly picking out a small number m of randomly chosen training inputs. Provided the sample size m is large enough it is expected that the average value of the $\nabla c(\boldsymbol{x}_j)$ will be roughly equal to the average over all $\nabla c(\boldsymbol{x})$. The method error back-propagation, published in 1986 [32], allows for an efficient way to calculate the stated partial derivates and makes training deep networks possible. The algorithm can be considered the working horse of deep learning and defines the error $\boldsymbol{\delta}^L$ in the last layer L of the network as

$$\delta_j^L = \frac{\partial c}{\partial z_j^L}. \tag{12}$$

Applying the chain rule yields

$$\delta_j^L = \sum_k \frac{\partial c}{\partial a_k^L} \frac{\partial a_k^L}{\partial z_j^L}, \tag{13}$$

where the sum is over all neurons k in the output layer. As the activation a_k^L depends only on the weighted input z_j^L when $k = j$, $\frac{\partial a_k^L}{\partial z_j^L}$ vanishes when $k \neq j$. The previous equation can be simplyfied to

$$\delta_j^L = \frac{\partial c}{\partial a_j^L} \frac{\partial a_j^L}{\partial z_j^L} \tag{14}$$

and with $a_j^L = a(z_j^L)$ the term on the right can be written as $a'(z_j^L)$, the equation becomes

$$\delta_j^L = \frac{\partial c}{\partial a_j^L} a'(z_j^L) \tag{15}$$

in component form and

$$\boldsymbol{\delta}^L = \nabla_{\boldsymbol{a}} c \odot a'(\boldsymbol{z}^L) \tag{16}$$

in matrix form, with $\odot$ noting the Hadamard product, being the first essential equation of error back-propagation. The second essential equation yields the error in higher layer l

$$\delta_j^l = \frac{\partial c}{\partial z_j^l}, \tag{17}$$

expressed in terms of the error in the next layer $l+1$ and by applying the chain rule the equation becomes

$$\begin{aligned} \delta_j^l &= \sum_k \frac{\partial c}{\partial z_k^{l+1}} \frac{\partial z_k^{l+1}}{\partial z_j^l} \\ &= \sum_k \frac{\partial z_k^{l+1}}{\partial z_j^l} \delta_k^{l+1}, \end{aligned} \tag{18}$$

where in the last line the two terms on the right-hand side have been interchanged and the definition of δ_k^{l+1} has been substituted. Differentiating

$$z_k^{l+1} = \sum_j w_{kj}^{l+1} a_j^l + b_k^{l+1} = \sum_j w_{kj}^{l+1} a(z_j^l) + b_k^{l+1} \tag{19}$$

yields

$$\frac{z_k^{l+1}}{\partial z_j^l} = w_{kj}^{l+1} a'(z_j^l) \tag{20}$$

and substituting back into equation 18 gives

$$\delta_j^l = \sum_k w_{kj}^{l+1} \delta_k^{l+1} a'(z_j^l) \tag{21}$$

in component form and

$$\boldsymbol{\delta}^l = ((\boldsymbol{w}^{l+1})^\top \boldsymbol{\delta}^{l+1}) \odot a'(\boldsymbol{z}^l) \tag{22}$$

in matrix form. The updates for the parameters of the layer are calculated by again using the chain rule and substituting δ_j^l

$$\frac{\partial c}{\partial b_j^l} = \underbrace{\frac{\partial c}{\partial z_j^l}}_{\delta_j^l} \underbrace{\frac{\partial z_j^l}{\partial b_j^l}}_{1} = \delta_j^l \tag{23}$$

$$\frac{\partial c}{\partial w_{jk}^l} = \underbrace{\frac{\partial c}{\partial z_j^l}}_{\delta_j^l} \underbrace{\frac{\partial z_j^l}{\partial w_{jk}^l}}_{a_k^{l-1}} = a_k^{l-1} \delta_j^l \tag{24}$$

The training process involves creating random mini-batches from the training set and calculating the parameter updates from the cost function. It is called an epoch, when a network has seen the whole training set for updates. The training goes on for multiple epochs until a stopping criterion is reached. Such a stopping criterion might be a acceptable small remaining error.

4.1.2 Multi-layer Networks

Multi-layer feedforward networks are the most essential deep learning models. They have universal approximation capabilities [33] and therefore can be used to approximate some function f^*. For classification the function would map an input $\boldsymbol{x}$ to categories $\boldsymbol{y}$. A feedforward network defines a mapping

$$\boldsymbol{y} = f(\boldsymbol{x}, \boldsymbol{\Phi}) \tag{25}$$

and learns the value of the parameters $\boldsymbol{\Phi}$ that result in the best function approximation.

Feedforward means that there are no feedback connections in which outputs of the model are given back into itself. Information flows through the function being evaluated from $\boldsymbol{x}$ through the intermediate layers and finally to the output $\boldsymbol{y}$. When networks include feedback connections, they are called recurrent neural networks (RNNs) [34].

Feedforward networks are called networks because they are typically represented by composing together many different functions. The model is associated with a directed

acyclic graph describing how the functions are composed together. A function $f(\boldsymbol{x})$ might be formed with

$$f(\boldsymbol{x}) = f^3(f^2(f^1(\boldsymbol{x}))), \tag{26}$$

creating a chain structure. The overall length of the chain gives the depth of the model.

4.1.3 Convolutional Neural Networks

A CNN [35] is a specialized neural network for processing data that has a grid-like topology. Examples are 2-D images or 3-D magnetic resonance (MR) images. It is called "convolutional neural network" because the network employs a mathematical operation called convolution, which is a specialized kind of linear operation. A convolution is defined as an integral over two functions of real valued argument

$$s(t) = \int x(a)w(t-a)da. \tag{27}$$

The convolution operation is typically denoted with an asterisk

$$s(t) = (x * w)(t), \tag{28}$$

where the first argument x is referred to as input and the second argument w as the kernel. The output of the operation is sometimes called feature map. For working with data in deep learning models the discretized version of the convolution is necessary:

$$s(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a). \tag{29}$$

MR images are usually a multidimensional array of data. For deep learning applications the kernel is also a multidimensional array of parameters that is adapted by the learning algorithm. The data arrays for the input and the kernel have finite size and it is assumed

that these functions are zero everywhere else. Convolutions are used over more than one axis at the time. For a 3-D MRI input convolved with a kernel the equation becomes:

$$S(i,j,k) = (X * W)(i,j,k) = \sum_m \sum_n \sum_o X(m,n,o)W(i-m, j-n, k-o). \tag{30}$$

The motivation behind convolutional networks cames from three ideas that can help improve a machine learning system:

- **Sparse interactions:** Fully connected layers use a matrix of parameters with a separate parameter describing the connection between every input unit and each output unit. Every output unit interacts with every input unit. However, convolutional networks have sparse interactions by making the kernel smaller than the input. Fewer parameters need to be stored, reducing the memory requirement for the model and improving its statistical efficiency. Also the number of operations for computing the output is reduced.
- **Parameter sharing:** For fully connected layers each weight of the matrix is used exactly once when computing the output of a layer. In contrast in convolutional layers the kernel is moved over the input using the same weights at different positions. It can be seen as tied weights because the value of the weight applied to one input unit is tied to the value of a weight applied elsewhere. Each member of the kernel is used at every position of the input. Only one set of parameters is learned instead of learning a separate set of parameters for every location.
- **Equivariant representations:** Because of the particular form of parameter sharing the layer has a property called equivariance to translation. Equivariance of a function means that if the input changes, the output changes in the same way. When processing MRI inputs, convolution creates a 3-D map of where certain features appear in the input. If the feature is moved in the input, its representation will be moved the same distance in the output. Naturally for convolution, equivariance is not given for other forms of translations, such as changes in the scale or rotation

of the input.

A convolutional layer is typically followed by a pooling layer. The purpose of the pooling layer is to replace the output of the convolutional layer with a summary statistic of the nearby output units. Widely used is the max pooling operation [36], which takes the maximum output within a rectangular neighborhood. Pooling helps to make the representation approximately invariant to small translations of the input.

Key design principles of convolutional neural networks have been drawn from neuroscience. Neurophysiologists David Hubel and Torsten Wiesel collaborated to determine many of the most basic facts about how the mammalian vision system works [37, 38, 39]. They observed how neurons in the cat's brain responded to images projected in precise locations on a screen in front of the cat. They found that neurons in the early visual system responded most strongly to very specific patterns of light, such as precisely oriented bars, but responded hardly at all to other patterns. A convolutional network is designed to capture properties attributed to the primal visual cortex of the brain such as the two-dimensional structure, simple cells and their activity characterized by a linear function of the image in a small, spatially localized receptive field, and complex cells inspiring pooling units. Convolutional neural networks have achieved high accuracies on data sets like ImageNet [40], resulting in their broad application.

4.2 Explainable AI

The ability to explain the reasons behind decisions is an important aspect, not only for human interaction but also for deep learning models. Explainability helps to identify appropriate models for the given task. Better understanding what models are doing and why they sometimes fail makes it easier to improve them. Following the arguments

in [41], the most important arguments for explainability in artificial intelligence (AI) are:

- **Verification of the system:** Black-box systems may not be trusted by default. In healthcare the use of models has to be interpretable and verifiable by medical experts.
- **Improvement of the system:** Understanding the weaknesses of an AI system is key to its improvement. It is easier to do such analysis on interpretable models than on black-box models by trial and error. Getting insight into a model's prediction can help identify biases in model or data sets. For example, preprocessing of the input may introduce unwanted features. Also the comparison of different models for the same task can be enhanced because comparing classification performance, without taking learned features into account, may lead to wrong models. Therefore identifying an appropriate model benefits from explainability.
- **Learning from the system:** Training models with millions of examples may reveal features which are not accessible by humans. Extracting these features can lead to new insights and help find hidden laws of nature.
- **Compliance to legislation:** As more AI systems find their way into areas of daily life, related legal aspects such as responsibility for made decisions receive increased attention. Black-box models may not be able to comply to these legal questions and therefore it is necessary that AI systems become more explainable. In the European Union users have a right for explanation when algorithms make decisions about them.

Following this arguments it is obvious that methods enabling explainability are necessary. In recent years heatmapping, a general idea for machine learning model prediction explanation, gained more attention and will be outlined in the next section.

4.3 Heatmapping

Explanation of a model prediction can be done with different approaches. In this two-step process the system first classifies the input and then an explanation method is applied to explain the prediction in terms of the input. The resulting heatmap has the same dimensionality as the input (in case of a volume it is also a volume), where the voxels represent the importance for the classification result. Figure 3 summarizes this process for the classification of an R2* map. The explanation methods sensitivity analysis (SA) and LRP are used to generate different kinds of heatmaps.

4.3.1 Sensitivity Analysis

This method uses the locally evaluated gradient to explain the predictions. The SA assigns the importance of each input variable i (for example image pixels or volume voxels) as

$$R_i = \left\| \frac{\partial}{\partial x_i} f(\boldsymbol{x}) \right\| . \tag{31}$$

As the name states this method attributes the highest scores to features which the output is most sensitive to. The partial derivative does not explain the function value $f(\boldsymbol{x})$ itself, but rather its variation. The variation may be a suboptimal measure for explaining the predictions of models. Important heatmap pixels or voxels calculated with this method indicate which input areas need to be changed to make the input look more or less like the predicted class. This measure assumes that the most relevant input features are those to which the output is most sensitive rather than explaining the prediction $f(\boldsymbol{x})$ itself. Hence, such a heatmap would not point to input features which are actually crucial for the prediction.

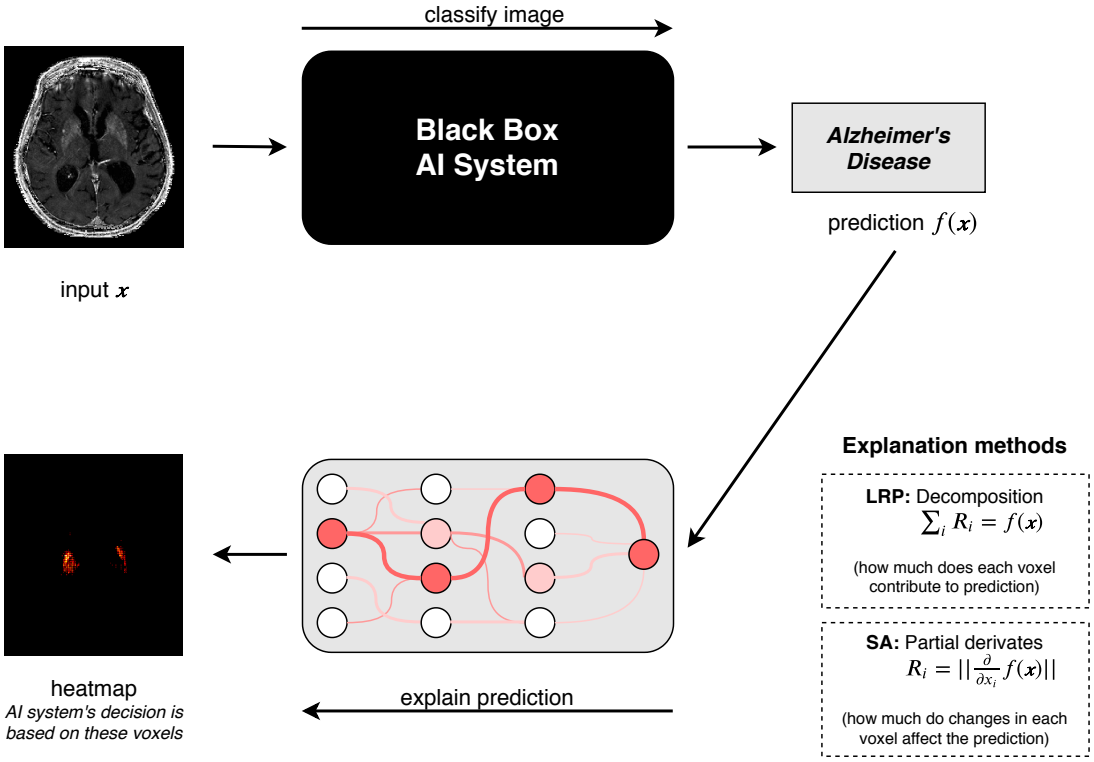


Figure 3: Explaining predictions of an artificial intelligence (AI) system based on [41]. The input image x is correctly classified as Alheimer's disease. In order to understand why the system has arrived at this decision, explanation methods such as sensitivity analysis (SA) or layer-wise relevance propagation (LRP) are applied. The result of this explanation is the heatmap, which visualizes the importance of each voxel. In this example the red-yellow voxels contribute most to the AI system's decision. With the heatmap it is possible to verify that the AI system works as intended.

4.3.2 Layer-wise Relevance Propagation

In contrast to SA, LRP is a framework for decomposing predictions of AI systems like feedforward neural networks in terms of input variables. It identifies pixels or voxels which are essential for the prediction. Therefore, this method redistributes the prediction $f(\boldsymbol{x})$ backwards using local redistribution rules until it assigns a relevance score R_i to each input variable (pixel or voxel). The process enforces relevance conservation which means no relevance is added or removed during the propagation through the layers and is formulated as

$$\sum_i R_i = ... = \sum_j R_j = \sum_k R_k = ... = f(\boldsymbol{x}). \tag{32}$$

Each relevance sum corresponds to a network layer and has to be equal to the prediction. The final relevance score R_i of each input variable determines how much this variable has contributed to the prediction.

For feedforward neural networks the redistribution itself can be done with the simple LRP rule:

$$R_j = \sum_k \frac{x_j w_{jk}}{\sum_j x_j w_{jk} + \epsilon} R_k, \tag{33}$$

where x_j is the neuron activation at layer l, R_k are the relevance scores associated to the neurons at layer $l+1$ and w_{jk} is the weight connecting neuron j to neuron k. To avoid divisons by zero a small stabilization term ϵ, which is positive when $\sum_j x_j w_{jk} \geq 0$ and negative else, is added. The value relevance conversation holds for $\epsilon = 0$. This rule redistributes relevance proportional to neuron activation x_j and the weight of the connection w_{jk}. More relevance is propagated through connections with large weights and more activated neurons x_j get larger shares of relevance.

As an advanced alternative the $\alpha\beta$-rule was introduced in [26], treating negative and

positive pre-activations separately:

$$R_j = \sum_k \left(\alpha \cdot \frac{(x_j w_{jk})^+}{\sum_j (x_j w_{jk})^+} - \beta \cdot \frac{(x_j w_{jk})^-}{\sum_j (x_j w_{jk})^-} \right) R_k, \tag{34}$$

where $(x_j w_{jk})^+ = x_j w_{jk} 1_{x_j w_{jk} > 0}$ and $(x_j w_{jk})^- = x_j w_{jk} 1_{x_j w_{jk} < 0}$ denote the positive and the negative part of $x_j w_{jk}$. To enforce conservation of relevance for this rule an additional constraint

$$\alpha - \beta = 1 \tag{35}$$

is necessary. For the special case $\alpha = 1$ the redistribution rule coincides with the deep Taylor decomposition of the neural network function when the ReLU activation is used throughout the neural network.

4.3.3 Deep Taylor Decomposition

It is desirable that heatmapping methods satisfy certain properties. The first definition is that heatmapping $R(\boldsymbol{x})$ is conservative, which was already introduced in equation 32. The second definition is $R(\boldsymbol{x})$ is positive if all values forming the heatmap are greater than or equal to zero:

$$\forall \boldsymbol{x}, p : R_p(\boldsymbol{x}) \geq 0. \tag{36}$$

This second property forces the heatmapping to assume that the model is lacking contradictory evidence. So no pixels or voxels are in contradiction with the classification result. These two properties combined form the third definition, which is heatmapping $R(\boldsymbol{x})$ is consistent if it is conservative and positive. There are multiple heatmapping methods which satisfy definition three. A simple example is the uniform redistribution

$$\forall p : R_p(\boldsymbol{x}) = \frac{1}{d} \cdot f(\boldsymbol{x}), \tag{37}$$

where d is the number of input dimensions.

It is not possible in practice to state all properties explicitly which make a heatmapping method meaningful. Instead, these definitions can be given implicitly by the choice of a particular algorithm derived from a mathematical model. The choice of the algorithm is constrained by definition three.

With arbitrary differentiable functions $f(\boldsymbol{x})$ a decomposition method based on Taylor expansion of the function at some well-chosen root $\tilde{\boldsymbol{x}}$ can be applied. The first-order Taylor expansion of $f(\boldsymbol{x})$ is given by

$$\begin{aligned} f(\boldsymbol{x}) &= f(\tilde{\boldsymbol{x}}) + \left(\frac{\partial f}{\partial \boldsymbol{x}}\Big|_{x=\tilde{x}}\right)^{\top} \cdot (\boldsymbol{x} - \tilde{\boldsymbol{x}}) + \epsilon \\ &= 0 + \sum_p \underbrace{\frac{\partial f}{\partial x_p}\Big|_{x=\tilde{x}} \cdot (x_p - \tilde{x}_p)}_{R_p(\boldsymbol{x})} + \epsilon, \end{aligned} \tag{38}$$

where the sum $\sum_p$ runs over all pixels or voxels in the input, $\{\tilde{x}_p\}$ are the pixel values of the root point $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}$ is chosen so that $f(\tilde{\boldsymbol{x}}) = 0$. The summed elements are identified as the relevances $R_p(\boldsymbol{x})$ assigned to the input units. Second-order and higher-order terms are denoted as ϵ. Most of them involve several pixels or voxels and are therefore more difficult to redistribute. Therefore, for simplicity only first-order terms are considered. Using the Hadamard product the heatmap can be calculated with

$$\boldsymbol{R}(\boldsymbol{x}) = \frac{\partial f}{\partial \boldsymbol{x}}\Big|_{x=\tilde{x}} \odot (\boldsymbol{x} - \tilde{\boldsymbol{x}}). \tag{39}$$

Choosing the free variable, the root point $\tilde{\boldsymbol{x}}$ is a challenge. A good root is one that removes what in the data point $\boldsymbol{x}$ causes the function $f(\boldsymbol{x})$ to be positive, but that minimally deviates from the original point $\boldsymbol{x}$ for the Taylor expansion to be still valid. In the general case the root point has to be found in iterative minimization procedure making Taylor decomposition not a good choice for explainability.

As a deep neural network consists of multiple layers, the resulting trained function $f(\boldsymbol{x})$ has a particular structure. Each neuron in the first layer may react to a particular

activation pattern that is localized in the input space. In higher layers the resulting activations patterns from the previous layer may be used to form more complex non-linearities that involve larger numbers of input pixels or voxels. The deep Taylor decomposition makes use of the property that a deep network can be structurally decomposed into a set of simpler subfunctions that relate quantities in adjacent layers. Therefore, instead of considering the whole neural network function f, the mapping of a set of neurons $\{x_i\}$ at a given layer to the relevance R_j assigned to a neuron x_j in the next layer is applied. To redistribute relevance R_j onto lower-layer relevances $\{R_i\}$ Taylor decomposition is applied on the local function that relates $\{x_i\}$ and R_j. Taylor decomposition of these simpler functions is easier than for the whole function f. In particular finding root points is a lot easier. A full backward pass through the network with this technique leads to the pixel- or voxel-wise heatmap.

Given the existence of a function that maps neuron activities $\{x_i\}$ to the upper-layer relevance R_j, and of a neighboring point $\{\tilde{x}_i\}$ such that $R_j(\{\tilde{x}_i\}) = 0$, the Taylor decomposition of

$$\sum_j R_j \text{ at } \{x_i\} \tag{40}$$

can be written as

$$\begin{aligned} \sum_j R_j &= \left(\frac{\partial(\sum_j R_j)}{\partial\{x_i\}}\Big|_{\{\tilde{x}_i\}} \right)^\top \cdot (\{x_i\} - \{\tilde{x}_i\}) + \epsilon \\ &= \sum_i \underbrace{\sum_j \frac{\partial R_j}{\partial x_i}\Big|_{\{\tilde{x}_i\}} \cdot (x_i - \tilde{x}_i)}_{R_i} + \epsilon, \end{aligned} \tag{41}$$

that redistributes relevance from one layer to the layer below, where ϵ denotes the Taylor residual, $|_{\{\tilde{x}_i\}}$ indicates that the derivative has been evaluated at the root point $\{\tilde{x}_i\}$, $\sum_j$ runs over neurons at the given layer, and $\sum_i$ runs over neurons in the lower layer. With equation 41 it is possible to identify the relevance of individual neurons in the lower layer in order to apply the same Taylor decomposition method one layer below.

Layer-wise relevance conservation, in the sense of the first definition, is given if each local Taylor decomposition in the network is conservative and the chain of equalities holds true. Similarly, if the second definition holds for each local Taylor decomposition, the positivity of relevance values at each layer is also ensured. The whole deep Taylor decomposition is consistent in the sense of definition three, if all Taylor decompositions of local subfunctions are consistent in the same sense.

The identification of the redistributed total relevance $\sum_j R_j$ onto the preceding layer was identified in equation 41 as:

$$R_i = \sum_j \frac{\partial R_j}{\partial x_i}\Big|_{\{\tilde{x}_i\}^{(j)}} \cdot (x_i - \tilde{x}_i^{(j)}). \tag{42}$$

Relevances R_i can therefore be obtained by performing as many Taylor decompositions as there are neurons in the hidden layer. A superscript $^{(j)}$ has been added to the root point $\{\tilde{x}_i\}$ in order to show that a different root point is chosen for decomposing each relevance R_j. Still there are various methods for choosing the root point $\{x_j\}$ that consider the diversity of possible input domains $\mathcal{X} \subset \mathbb{R}^d$. Each choice of input domain and associated method to find a root will lead to a different rule for propagating relevance $\{R_j\}$ to $\{R_i\}$. To make the search for root points feasible, the input domain is restricted to $\mathcal{X} = \mathbb{R}^d_+$. This restriction arises in feature spaces that follow the application of ReLUs. In that case the search domain is restricted to the segment

$$(\{x_i 1_{w_{ij} \leq 0}\}, \{x_i\}) \subset \mathbb{R}^d_+ \tag{43}$$

that contains at least one root. Injecting the nearest root on that segment into equation 42, leads to the z^+ relevance propagation rule

$$R_i = \sum_j \frac{z_{ij}^+}{\sum_i z_{ij}^+} R_j, \tag{44}$$

where $z_{ij}^+ = x_i w_{ij}^+$, and $w_{ij}^+ = w_{ij} 1_{w_{ij} > 0}$ denotes the positive part of w_{ij}. This rule corresponds to the $\alpha\beta$-rule shown in equation 34 with $\alpha = 1$ and $\beta = 0$. An additional

constraint is set on the bias of the layers, where $b_j \leq 0$ is forced for all j. Using this constraint guarantees the existence of a root point $\{\tilde{x}_i\}$ of the mapping function and therefore ensures the applicability of standard Taylor decomposition of the local subfunctions.

5 Methods

In this chapter the applied methods are presented. The first part describes the subjects and the database, followed by an overview of the MRI protocol set up for the data acquisition. Furthermore, the images used as input for the development of the network and their preprocessing are described. Finally, the networks and their trainings are summarized.

5.1 Subjects

For the development of the classifier networks two cohorts that represent the classes AD and NC were created. The MRI data sets from patients with probable AD were retrospectively selected from the outpatient clinic "Department of Neurology, Medical University of Graz, Austria". 259 data sets from 121 patients have been included (mean age 72 ± 8.8). 245 data sets from 183 age-matched healthy controls (mean age 69 ± 9.8) were selected from an ongoing community dwelling study. Each subject must at least contain a T1-weighted structural MPRAGE scan, which is necessary for registration steps. The selected MRI data sets were acquired with the protocol described in the next section. The rationale underlying about why each contrast was included is also given.

5.2 MRI Protocol

Patients and controls were scanned using a consistent quantitative MRI protocol at 3 Tesla (Siemens TimTrio) taking approximately 45 minutes including a T1-weighted MPRAGE sequence, a FLAIR sequence, a proton-density-/T2-weighted turbo-spin-echo sequence, a spoiled fast low-angle shot (FLASH) sequence and a magnetization transfer (MT) contrast sequence. Obtained images, subsequently calculated images (also called maps) and their diagnostic value are described in this section.

5.2.1 MPRAGE Images

The 3-D MPRAGE sequence [42] is one of the most widely applied sequences for structural brain imaging. It is used for high-resolution whole brain T1-weighted imaging, applied in clinical daily routine and also in research settings [43]. Images acquired with the sequence have been widely used for classifying brain tissues in voxel-based morphometry [44], detecting pathological changes of the brain [45] and increasing diagnostic accuracy for AD with whole-brain volumetry [46]. Hence, these images recently have been used with machine learning classifiers to assess structural changes in the brain due to disease effects [15].

Motivated by these results, MPRAGE images are included in this work. An example can be seen in figure 4. Image was acquired with resolution = 1×1×1 mm^3, matrix = 176×224×256, echo time = 2.6 ms, repetition time = 1900 ms, inversion time = 900 ms.

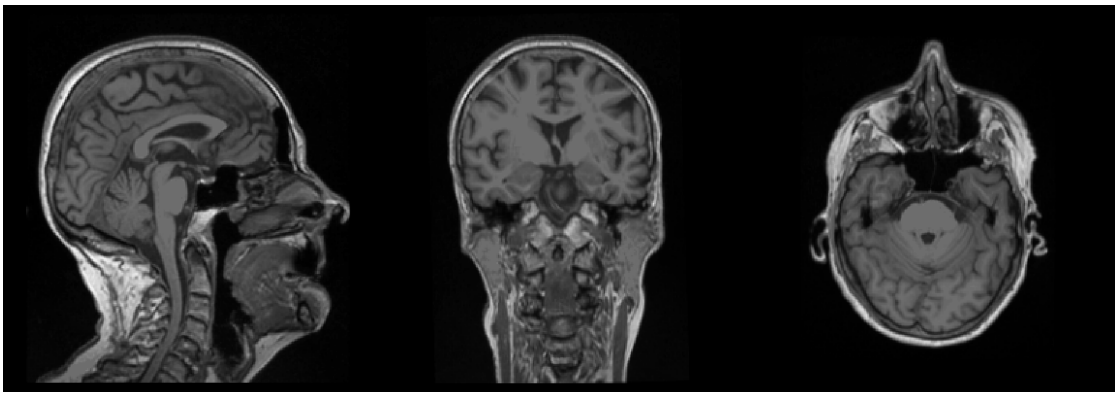


Figure 4: Normal control (NC) example of magnetization-prepared rapid gradient-echo (MPRAGE) image with 1 mm isotropic resolution. Sequence is used for high-resolution whole brain T1-weighted imaging, applied in clinical daily routine and also in research settings. MPRAGE images contain structural information.

5.2.2 FLAIR Images

FLAIR sequences have inversion recovery time set to null fluids. It is used in brain imaging to suppress cerebrospinal fluid (CSF) effects on the image, which enhances the visibility of periventricular hyperintense lesions, such als multiple sclerosis plaques [47], which are of great relevance for diagnoses. Aside from that, FLAIR images capture structural information, useful for the AD classification task and therefore are included in this work. An example image is given in figure 5 obtained with resolution = 0.9×0.9×3 mm^3, matrix = 192×256×44, slices = 44, echo time = 70 ms, repetition time = 10 s, inversion time = 2500 ms.

5.2.3 R2 Maps

R2 maps were estimated from a proton-density-/T2-weighted turbo-spin-echo sequence with resolution = 0.9×0.9×3 mm^3, matrix = 192×256, slices = 40, echo time = 10 ms,

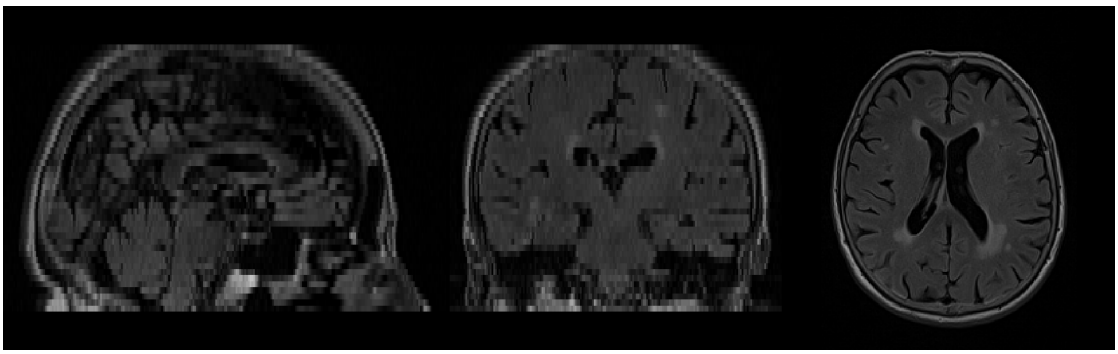


Figure 5: Alzheimer's disease (AD) example of fluid-attenuated inversion recovery (FLAIR) image. Sequence is used in brain imaging to suppress cerebrospinal fluid effects on the image and to capture structural information.

echoes = 2, echo spacing = 63 ms, repetition time = 5260 ms. Because there are only two echoes available voxel-wise computation was done by linearizing the inverse problem

$$M_{xy}[i] = M_{xy}[0]e^{-t[i]R_2}, \tag{45}$$

where $M_{xy}[i]$ is the measured voxel intensity at echo i, $M_{xy}[0]$ is the not measured start signal and $t[i]$ is the elapsed time at echo i. The linearized equation is solved by minimizing the quadratic errors. Solving the inverse problem was not part of this work but had been done previously during working on the Bachelor's Thesis [48].

Correlation between changes in R2 maps and the presence of AD was shown in [49, 50]. Hence, R2 maps are investigated in this work. An example is presented in figure 6.

5.2.4 R2* Maps

The estimation of R2* maps from a spoiled FLASH sequence with resolution = 0.9×0.9×2 mm^3, matrix = 208×256, slices = 64, echo time = 4.92 ms, echoes = 6, echo spacing = 4.92 ms, repetition time = 35 ms, was done in a similiar way as described in section 5.2.3.

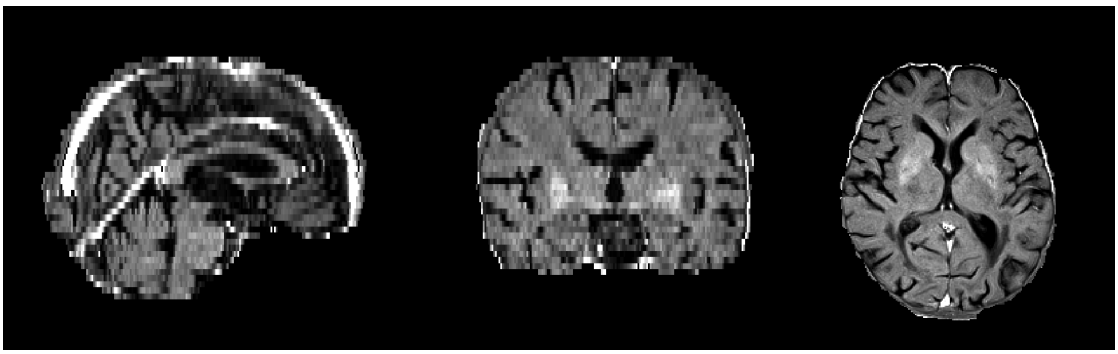


Figure 6: Normal control (NC) example of skull stripped R2 map. Changes in R2 maps may reflect on the presence of Alzheimer's disease (AD).

For the theoretical signal decay of the transversal magnetization with R2* an overview is shown in figure 7.

R2* is considered as a measure of iron content in gray matter and histological and in-vivo studies have shown that brain iron strongly accumulates, in particular in AD patients when compared to healthy controls [51, 11, 52]. Therefore, the computed R2* maps are included in the classifier trainings. A representative R2* map is shown in figure 8.

5.2.5 MTR Maps

Standard MRI directly detects only signals from mobile water protons with adequately long T2 relaxation times. In human brain tissue, this signal comes from free intra- and extracellular tissue water, called the "free water pool". Conversely, protons bound to macromolecules have too short T2 relaxation times (about 10 µs) to be detected directly. Macromolecules that bind protons in brain tissue are for example myelin proteins and lipids. This bound proton fraction, the "bound water pool", can be imaged indirectly by exploiting the transfer of magnetization between both proton pools, which is caused by

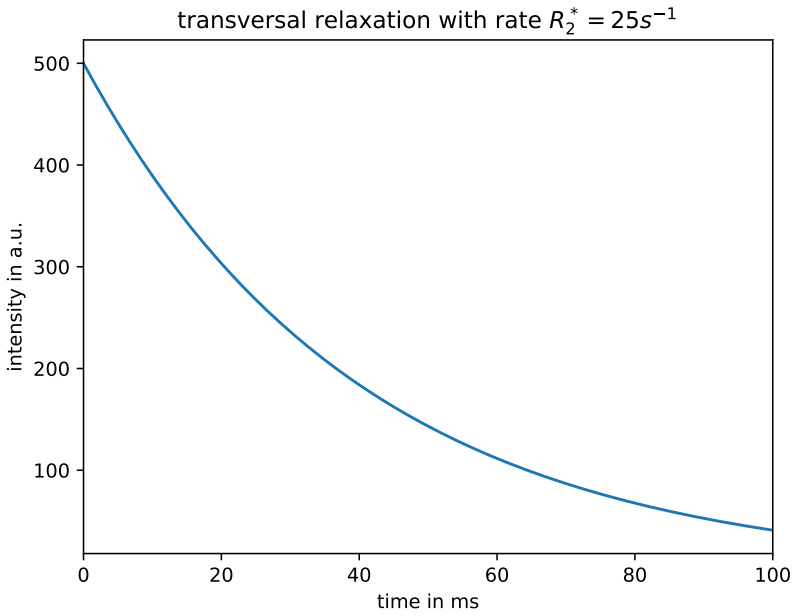


Figure 7: Theoretical transversal intensity relaxation. Intensity is given in arbitrary unit (a.u.).

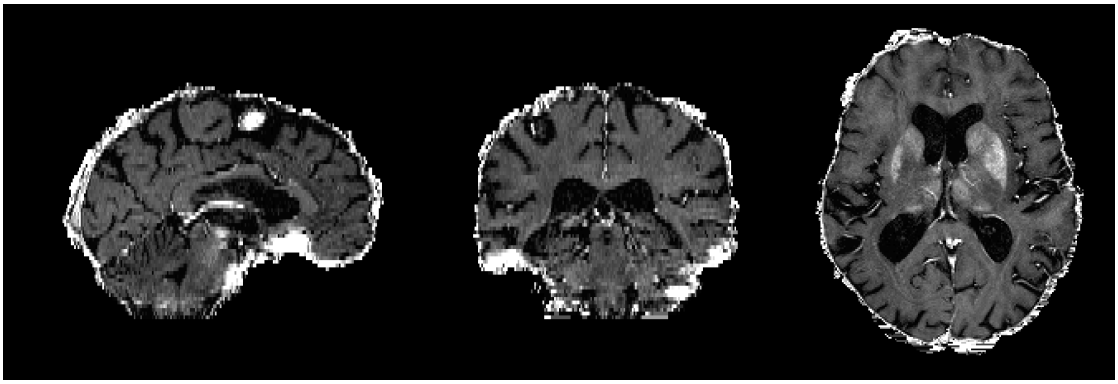


Figure 8: Alzheimer's disease (AD) example of skull stripped R2* map. R2* is considered as a measure of iron content in gray matter and histological and in-vivo studies have shown that brain iron strongly accumulates, in particular in Alzheimer's disease (AD) patients when compared to healthy controls.

dipolar coupling and chemical exchange mechanisms. In MT imaging, an off-resonance radio frequency pulse saturates the bound pool magnetization. Subsequent magnetization transfer shifts this magnetization to the magnetic resonance visible "free water pool". As a consequence, longitudinal magnetization decreases and causes a reduction of signal intensity. The degree of the MT induced signal decrease is usually assessed by the MTR [53, 54]. The MTR scales with the amount of magnetization exchange and with the extent of the "bound water pool" and is expressed through equation

$$\mathrm{MTR} = \frac{M_0 - M_{\mathrm{SAT}}}{M_0}, \tag{46}$$

where M_0 describes the signal intensity of a voxel without radio frequency saturation. M_{SAT} is the signal intensity of the identical voxel, acquired with the radio frequency saturation pulse [55, 56]. A common approach to assess global MTR changes is histogram analysis.

Differences in measured MT between AD patients and NC have been found in [54]. They conducted a voxel-based study by using non-linear registration and inclusion of a volumetric map to minimize partial volume effects resulting from atrophy and subsequent CSF contamination. The authors identified reduced MTR values in AD patients mainly in the hippocampus, temporal lobe, posterior cingulate, and parietal cortex. They have shown that MTR abnormalities in AD occur in a disease-specific pattern, independent of cortical atrophy. A review for MT imaging is given in [57].

Hence, for this work MTR maps are included for training deep learning classification networks. The inputs with saturation pulse and without saturation pulse for the calculation of MTR maps have been acquired with resolution = 0.9×0.9×3 mm^3, matrix = 192×256, slices = 40, echo time = 7.38 ms, repetition time = 40 ms. An example of a computed MTR map is presented in figure 9.

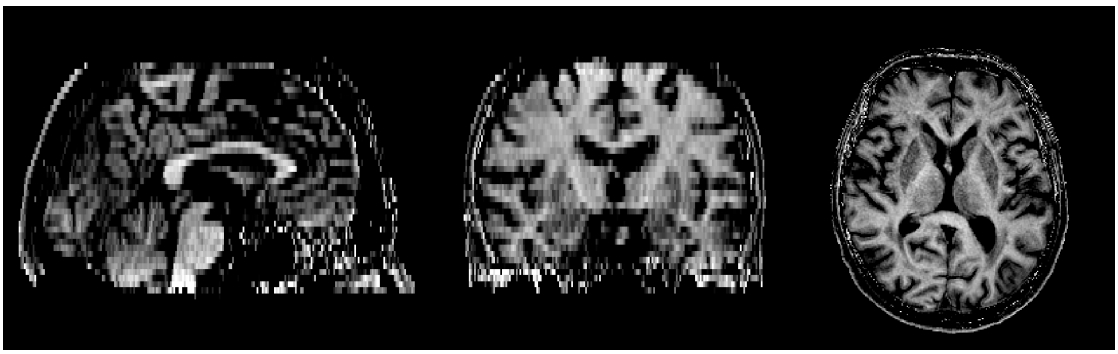


Figure 9: Normal control example of magnetization transfer ratio (MTR) map. MTR abnormalities in AD occur in a disease-specific pattern, independent of cortical atrophy.

5.3 Image Preprocessing

Skull stripping (brain extraction) is an often applied preprocessing step when brain MR images are used in deep learning classification [15, 58]. The term "preprocessing" is used to describe image manipulations done before handing the image to the learning task.

The brain masks necessary for brain extraction were obtained for the MPRAGE images by executing SIENAX [59] from FSL [60]

```
sienax <input> -B "-f 0.35 -B",
```

where the parameter B declares the parameter list encompased in "-f 0.35 -B" for the underlying FSL BET2 program, f defines the fractional intensity threshold (values between 0 and 1) and the second B attempts to reduce image bias and residual neck voxels. Resulting brain mask is used to create the skull stripped MPRAGE image by application of FSLMATHS [60]. An example is shown in figure 11.

All acquired images for each patient are stored in their native image spaces. Thus, the brain mask obtained from the MPRAGE image cannot be used directly on the other contrasts. It needs to be transformed to the respective image space. Therefore, the affine transformation matrices, transforming from each subject's image space to the subject's MPRAGE space, were obtained using FSL FLIRT [60] with mutual-info cost function. The transformation matrices were inverted and used to create the brain masks from the MPRAGE brain mask. Because of the different field of view position and resolution between translation source and target, the translated mask needs to be interpolated on the target grid. As the mask image only consists of zeros and ones and the transformed mask should also, the nearest-neighbor interpolation was used.

Image intensity normalization can help speed up the training process. It becomes even more important when multiple inputs with different scales are feed to a deep network. Different approaches for normalization have been proposed. Images are often normalized to their maximum value, giving intensity values between 0 and 1. However, images are normalized after skull stripping which leaves the normalization result highly influenced by the quality of the skull stripping process. High-valued voxels can often be found at the border between brain tissue and background. Moreover, quantitative image values like in R2* maps should be normalized to the same value to preserve the inter-subject differences. Hence, all images of the same contrast were normalized to the same constant value. For R2 maps, R2* maps and MTR maps the constant values were derived from physiological meaningful brain tissue values in the literature [61, 62]. In contrast, for MPRAGE and FLAIR images the constants are estimated by inspection of the cohorts. This normalization method does not guarantee for each image intensity to be in the range between 0 and 1, but it does for voxels being part of the brain tissue. Therefore, during training the network should pickup brain features more easily.

5.4 Networks

The final utilized network is the combination of a classification network and a corresponding heatmap generator. Heatmap generating is not only done after training for inference, but already during training. This allows for guiding the training process towards image features inside predefined regions (masks). After training of the combined network, the heatmap generator is also used for computing the final heatmaps alongside the classifier's prediction.

5.4.1 Classifier

The classification network uses the combination of a single convolutional layer followed by a down-convolutional layer as the main building block. The overall classification network stacks three of those main building blocks before passing the data through two fully connected layers. Each layer is followed by a ReLU non-linearity, except for the output layer where a Softmax activation is applied.

5.4.2 Heatmap Generator

Based on the definition of the classification network, the structure of the heatmap generator is derived. Thus, for every layer in the classifier an explanation layer, implementing deep Taylor decomposition [27], is added to the generator. The input, weights and output of the forward layer are shared with the backward layer for efficient memory usage. Using the shared properties, the explanation layer computes the redistribution of relevances according to z^+-rule in equation 44. For efficient computation it is necessary

to implement the redistribution as matrix operations rather than as the summation of the unit-wise calculation. Therefore, the equation is rearranged to

$$R_i = \sum_j a_i w_{ij}^+ \frac{R_j}{\sum_i a_i w_{ij}^+}, \tag{47}$$

where z_{ij} has been replaced by its definition $a_i w_{ij}^+$ and a_i is the i^{th} input to the forward layer. With $\sum_i a_i w_{ij}^+$ reducing to a_j, the outgoing activation when ReLU is applied. Hence, the first tensor operation necessary for implementation is

$$\boldsymbol{S} = \boldsymbol{R}^{l+1} \oslash \boldsymbol{A}^l, \tag{48}$$

where $\boldsymbol{S}$ is the intermediate result in the l^{th} layer, $\boldsymbol{R}^{l+1}$ the relevance from the higher layer, $\boldsymbol{A}^l$ the activation in the forward layer and $\oslash$ denoting the point-wise division. Going back to equation 47 and inserting the unit s_j from $\boldsymbol{S}$ yields

$$R_i = \sum_j a_i w_{ij}^+ s_j. \tag{49}$$

The summation runs over the j^{th} column of the layer's weight matrix. Given $\boldsymbol{S}$ is a column vector it is necessary to transpose the column from the weight matrix to compute R_i from them. For all R_i this combines to matrix equation

$$\boldsymbol{R}^l = \boldsymbol{W}^\top \boldsymbol{S} = \boldsymbol{W}^\top (\boldsymbol{R}^{l+1} \oslash \boldsymbol{A}^l), \tag{50}$$

where $\boldsymbol{W}^\top$ is the transposed weight matrix of the l^{th} layer. The transposed operation is very important for deep networks because they are also used during error back-propagation. In most of the tensor libraries, like TensorFlow [63] or PyTorch [64], this pairs are implemented for the available operations. The summarized case shows the use of this pairs for fully connected layers, which are basically matrix multiplications. For the implementation of this work the second important operation is the convolution. Hence, for the relevance distribution of a convolutional layer the "transposed convolution" is applied. The operation is called "transposed convolution" because it transposes

the given kernel before calculating the convolution. With the matrix equations for relevance redistribution defined, the heatmap generator layers can be build for all types of layers used in the classifier.

5.4.3 Relevance-guided Network

By combining the classification and the heatmap generator network, the "relevance-guided network" is obtained. In overview-figure 10 it can be seen that the network has two outputs, the classification result and the generated heatmap. The binary cross entropy loss function is utilizied to train the classifier. It measures how far away the current prediction is from the correct result. Using only this measure for training would stear the network to learn any features that helps discriminating between the two classes AD and NC. This features might not be useful in a diagnostic sense. To account for that this work proposes a new method to focus the network on relevant features within masks without directly manipulating the input images. Thus, to guide the training process, the output of the heatmap generator is used to extend the loss function of the classifier by

$$\text{loss}_{\text{relevance}}(\boldsymbol{R}, \boldsymbol{M}) = -\mathbf{1}^{\top} \text{vec}(\boldsymbol{R} \odot \boldsymbol{M}), \tag{51}$$

where $\boldsymbol{R}$ denotes the relevance, $\boldsymbol{M}$ is a predefined mask, $\text{vec}(\boldsymbol{A})$ denotes the row major vector representation of $\boldsymbol{A}$, and $\mathbf{1}$ is a vector where all elements are set to one. Note, that the negative sign accounts for the maximization of the relevance, and $\odot$ denotes the Hadamard product.

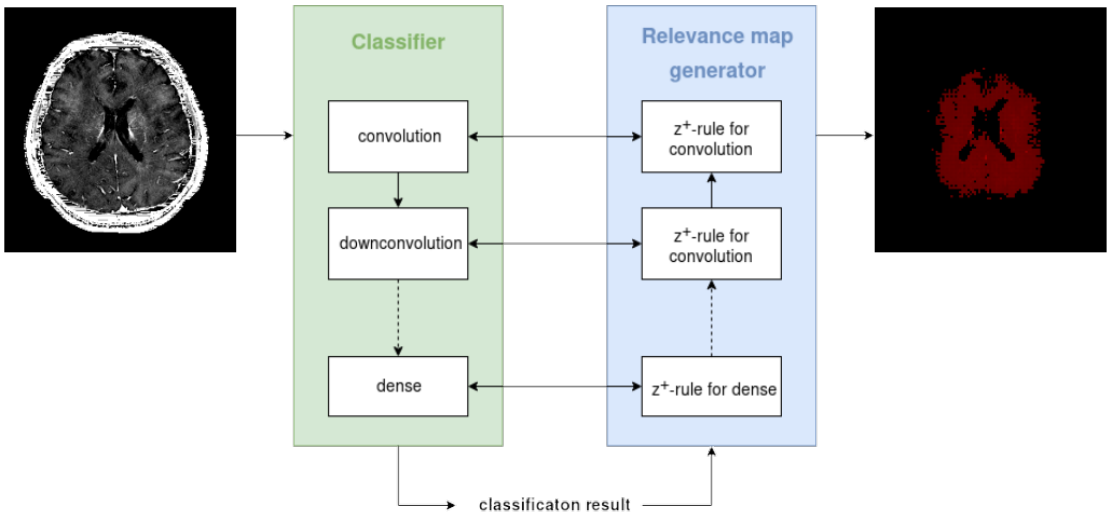


Figure 10: Overview of the used relevance-guided classification network. A classification network (green) has been extended with a relevance map generator (blue). For every layer of the classification network a corresponding relevance redistribution layer is added to the generator network. The output relevance map of the generator has the same resolution as the R2* map used as input for the classifier and allows to guide the training of the classification network by adding a term that sums relevance values inside a given brain mask to the categorical cross entropy loss.

5.4.4 Training

Development and training was done in three stages. The explainability method LRP [26] was implemented to get more insight into which regions of the input image are important for the classifier. The first stage was established because FLAIR images showed good classification performance. However, obtained heatmaps revealed preprocessing problems, such as highly "relevant" voxels in the skull, leading to the proposed method "relevance-guided training". Thus, in the first stage models were trained with FLAIR images downsampled to 2 mm isotropic in three configurations:

- FLAIR images in their native subject space without any preprocessing except for intensity normalization.
- FLAIR images in their native subject space with skull stripping and intensity normalization.
- FLAIR images in their native subject space without skull stripping, intensity normalization and the relevance-guided network.

Resulting performance and heatmaps are shown in chapter 6 and encouraged the setup of the second stage. For the second stage R2* images downsampled to 2 mm with the following configurations were applied to test if volumetric information contained in the input images influences the training of the classifier:

- R2* images in their native subject space with intensity normalization.
- R2* images linearly registered to individual subject's T1 space with intensity normalization.
- R2* images non-linearly registered to MNI152 with intensity normalization.

For all cases the skull stripping version, that is the standard classification network with masked input, against the proposed relevance-guided method was trained and

tested. For the third stage, models for MPRAGE, FLAIR, R2, R2* and MTR images with their native resolutions were trained with the relevance-guided method only.

5.4.5 Implementation

Developing the networks was done with the Python programming language (Python Software Foundation, https://www.python.org/) using Keras [65] with the TensorFlow [63] backend. The implementation consists of the following major parts:

- **Input and output:** To avoid loading all data sets for training and validation into the computer's memory the Keras Sequence class is inherited and implemented. The Keras framework uses the Sequence implementation to only load data sets which are currently necessary for processing. Thus, it becomes feasible to train the networks with the amount of MRI data used in this work. For the Sequence to work it needs to know where to find the data on the hard disk, which is given as base path. In the base path are the folders which make up for the classes (AD and NC). Inside each class are the subject folders, which in turn contain the MRI data and masks. The MRI data is stored in the NIfTI format (https://nifti.nimh.nih.gov/) and is read and written with the Nibabel Python package [66].
- **Classifier network:** Keras provides high level abstractions to the TensorFlow variables and operations. Networks can be composed with the different available layer classes. The classifier is implemented with Conv3D, Dense (fully connected) and Activation layers. To support the applicability of the heatmap generator, a new Bias layer for the classifier is introduced. The Bias layer is intended to add the bias parameter to the incoming tensor. To make sense of it, the bias operation is suppressed in the preceding layers (Conv3D or Dense), making their output reusable for sharing with the heatmap generator network.

- **Heatmap generator builder:** Creating the heatmap generator network is achieved through a method reversly traversing through the tensor graph of the classifier network and adding deep Taylor decomposition layers suitable for the visited layers. The deep Taylor decomposition layers implement the formulation in equation 50.
- **Training:** The code responsible for training makes use of the previous three parts, manages the hyperparameters and the optimizer, saves the parameters after each training epoch and makes overseeing the training process through TensorBoard [63], a progress visualizaton tool, possible.
- **Inference:** The final main part is obliged to load the selected, previously saved parameters and use them for inference on the test set. The outputs are the class probabilities and the corresponding heatmap for a test input image.

6 Results

Classification accuracies on the test set in table 2 show similiar performance for all three development stages. However, comparing obtained heatmaps reveals significant differences in the learned features. The application of skull stripping and image registration may introduce artifacts and biases into the images, which are picked up by network during training. The relevance-guided method improved the learned features and lead to explanations with specific brain regions, which have previously been identified to significantly differ between AD and NC.

6.1 Preprocessing

In figure 11 a skull stripping result obtained with FSL SIENAX [59, 60] is presented. The corresponding intensity probability histogram in figure 12 shows that single voxels with high intensity, compared to the value range of up to 400 with higher probabilities, remain after applying the brain mask. For comparison the intensity probability of the same MPRAGE image without skull stripping is shown. These histogram distributions are similar for all skull stripped MPRAGE images.

Image intensity normalization was done with constants listed in table 1. Constants for MPRAGE and FLAIR images were defined through inspection of the data cohorts. For

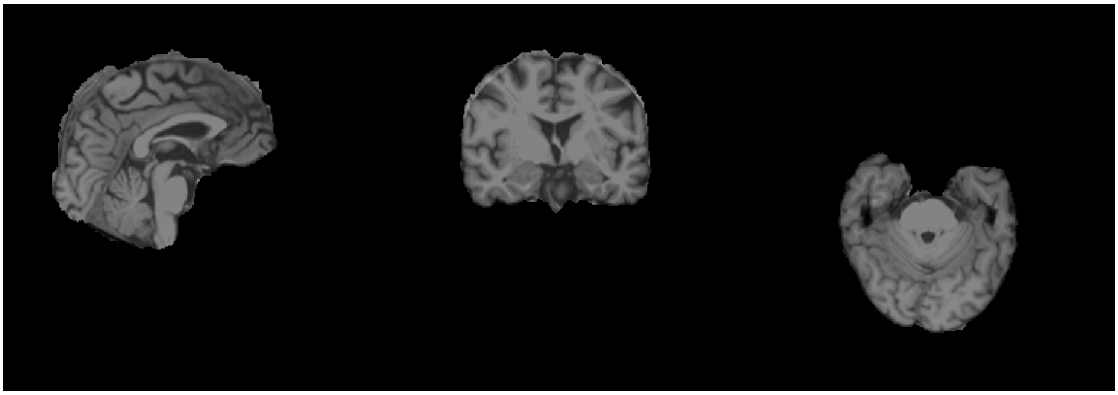


Figure 11: Skull stripped magnetization-prepared rapid gradient-echo (MPRAGE) image. For skull stripping FSL SIENAX was used. High intensity voxels remain anterior the brain stem.

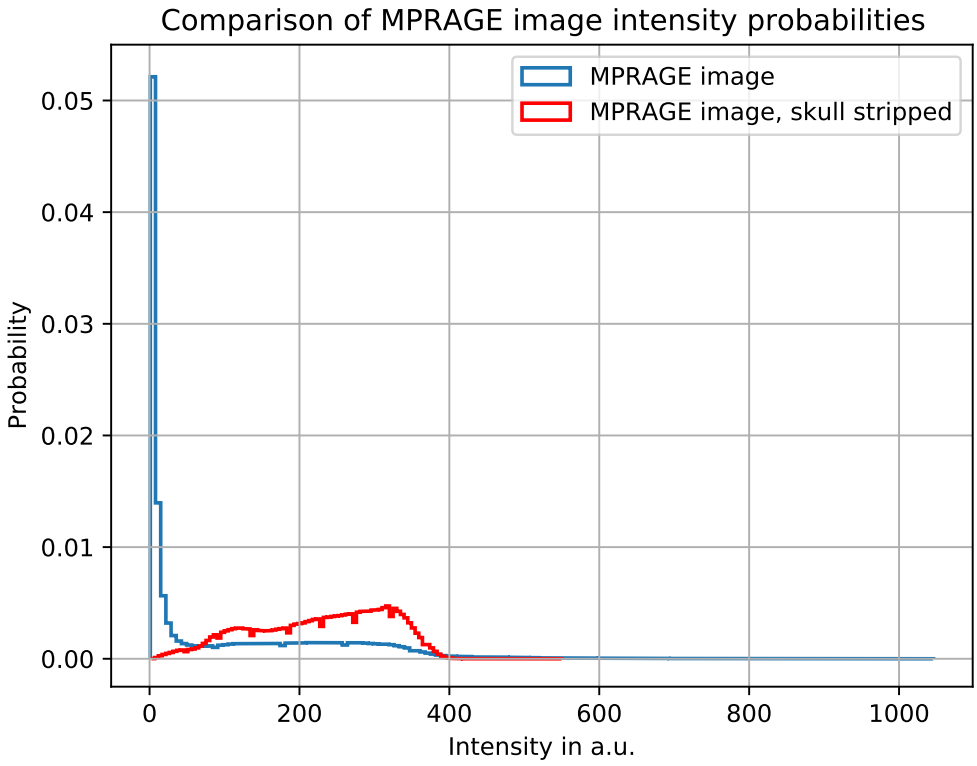


Figure 12: Comparison of intensity propability histograms of magnetization-prepared rapid gradient-echo (MPRAGE) image with skull stripped MPRAGE image.

R2, R2* and MTR constants were derived from literature [61, 62] and were verified by inspection of the data cohort.

Image contrast	Normalization value
MPRAGE	450.00 a.u.
FLAIR	1000.00 a.u.
R2	0.02 1/s
R2*	40.00 1/s
MTR	0.50 a.u.

Table 1: Intensity normalization values for used image contrasts.

6.2 Networks

In the resulting classifier's network architecture all convolutional layers are defined to have eight kernels with size 3×3×3 each. Additionally the down-convolution layers have the striding parameter set to 2×2×2. The down-convolution layers replace and serve as pooling layers, reducing the dimensionality as the data flows through the network. First fully connected layer contains 16 units, followed by the final fully connected layer with two units. Weights of all layers were initialized with values from the Glorot uniform distribution [67], whereas all biases were initialized with zeros. All biases were constrained to be non-positive to fulfill the requirements for deep Taylor decomposition.

Adam optimizer [68] with learning rate 0.001 and default parameters was used to update the parameters during training. Training went on for 300 epochs for all configurations. The data was split up into 354 training, 75 validation and 75 test subjects,

keeping the class distribution. Current parameters were saved at the end of each training epoch. Optimal parameters were selected retrospectivly from the list of saved parameters.

6.3 Classification Performance

The classification performances achieved in the three development stages are presented in table 2. For stage three the receiver operating characteristic (ROC) curves and the corresponding area under curve (AUC) are shown in figure 13. To speed up development the input images used in stages one and two were downsampled to 2 mm isotropic resolution. The same classifier architecture was used for all configurations. For configurations marked with "relevance-guided", the classifier was extended with the heatmap generator. Furthermore, the heatmap generator's output was used to regularize the learning task by adding the relevance loss function to the classifier's loss.

6.4 Heatmaps

Figure 14 shows the computed heatmaps of a patient with AD for all three configurations of stage one. Presented is an subject, which was correctly classified by all three trained models. The overlaid brain mask (purple) was obtained from subject's MPRAGE image and linearly transformed to subject's FLAIR space. FLAIR image and brain mask were downsampled to 2 mm isotropic resolution. The highlighted regions for the model without skull stripping concentrate outside of the brain (green), whereas the focused regions of the model with skull stripping (blue) are at the border of the brain mask. The map from relevance-guided training (red) on the other hand shows relevant regions inside the brain, in the presented case it might be interpreted visually as a combination

Stage	Configuration	Accuracy
1	FLAIR, native space	87 %
	FLAIR, native space, skull stripped	85 %
	FLAIR, native space, relevance-guided	85 %
2	R2*, native space, intensity norm., skull stripped	80 %
	R2*, native space, intensity norm., relevance-guided	82 %
	R2*, lin. reg. MPRAGE, intensity norm., skull stripped	82 %
	R2*, lin. reg. MPRAGE, intensity norm., relevance-guided	86 %
	R2*, nonlin. reg. MNI152, intensity norm., skull stripped	82 %
	R2*, nonlin. reg. MNI152, intensity norm., relevance-guided	82 %
3	MPRAGE, native space, intensity norm., relevance-guided	–
	FLAIR, native space, intensity norm., relevance-guided	94 %
	R2, native space, intensity norm., relevance-guided	90 %
	R2*, native space, intensity norm., relevance-guided	83 %
	MTR, native space, intensity norm., relevance-guided	88 %
	MPRAGE, native space, intensity norm.	77 %
	MPRAGE, native space, intensity norm., skull stripped	74 %

Table 2: Comparison of classification performances on the test set for all three stages. For stage one and two the images were downsampled to 2 mm isotropic and accuracy was calculated for the class threshold of 0.5. Accuracy for stage three was calculated for the class threshold obtained with Youden's index [69]. Training with magnetization-prepared rapid gradient-echo (MPRAGE) in stage three was not successful.

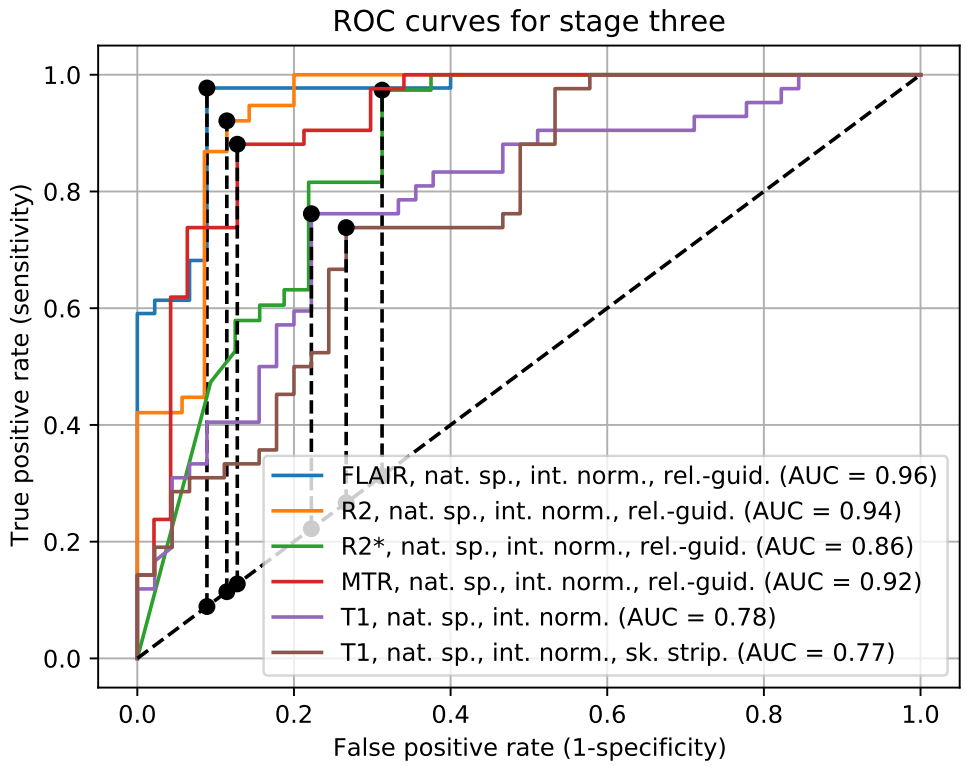


Figure 13: The receiver operating characteristic (ROC) curves for training stage three. Vertical dashed black lines between black points mark Youden's index [69].

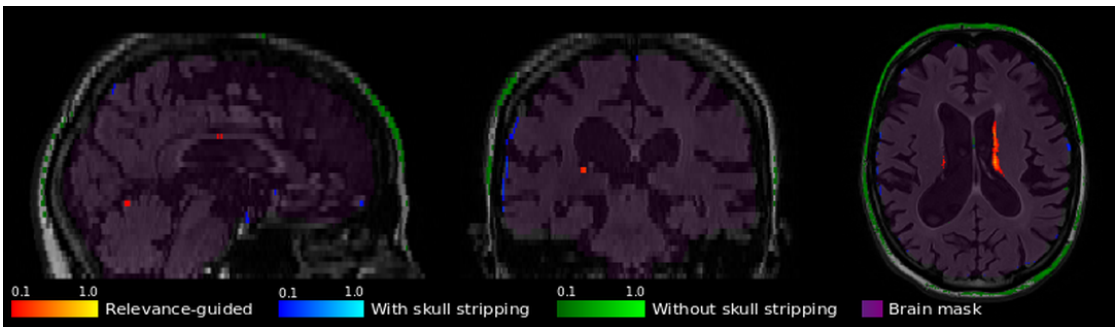


Figure 14: FLAIR image of a patient with Alzheimer's disease overlaid with the brain mask (purple). Additionally shown are the relevance maps obtained by the model without skull stripping (green), the model with skull stripping (blue) and the relevance guided model (red). Relevance maps are presented normalized to the maximum value.

of ventricle atrophy and white matter hyperintensities. Relevance maps are presented normalized to the maximum value and normalized relevances above a threshold of 0.1 are shown.

The second stage was setup to test for volumetric information contained in all images. Models were trained on native R2* maps, R2* maps affinely registered to subject's MPRAGE image and R2* maps non-linearly registered to MNI152 template. For all three configurations the version with skull stripping, that is the standard classification network with masked input, against the proposed relevance-guided method is compared. All obtained mean heatmaps in figure 15 are presented in MNI152 template space. Most important features for skull stripped version are found at the brain-skull interface region for all three configurations (yellow overlays). In contrast, mean heatmaps for relevance-guided method show relevant regions within brain tissue, with most identified features concentrated in the basal ganglia region (red-orange overlays).

For the final stage MPRAGE, FLAIR, R2, R2* and MTR images were used for training with the relevance-guided method. As stated in table 2, training with MPRAGE images did not converge and therefore no classification performance is reported. Training the

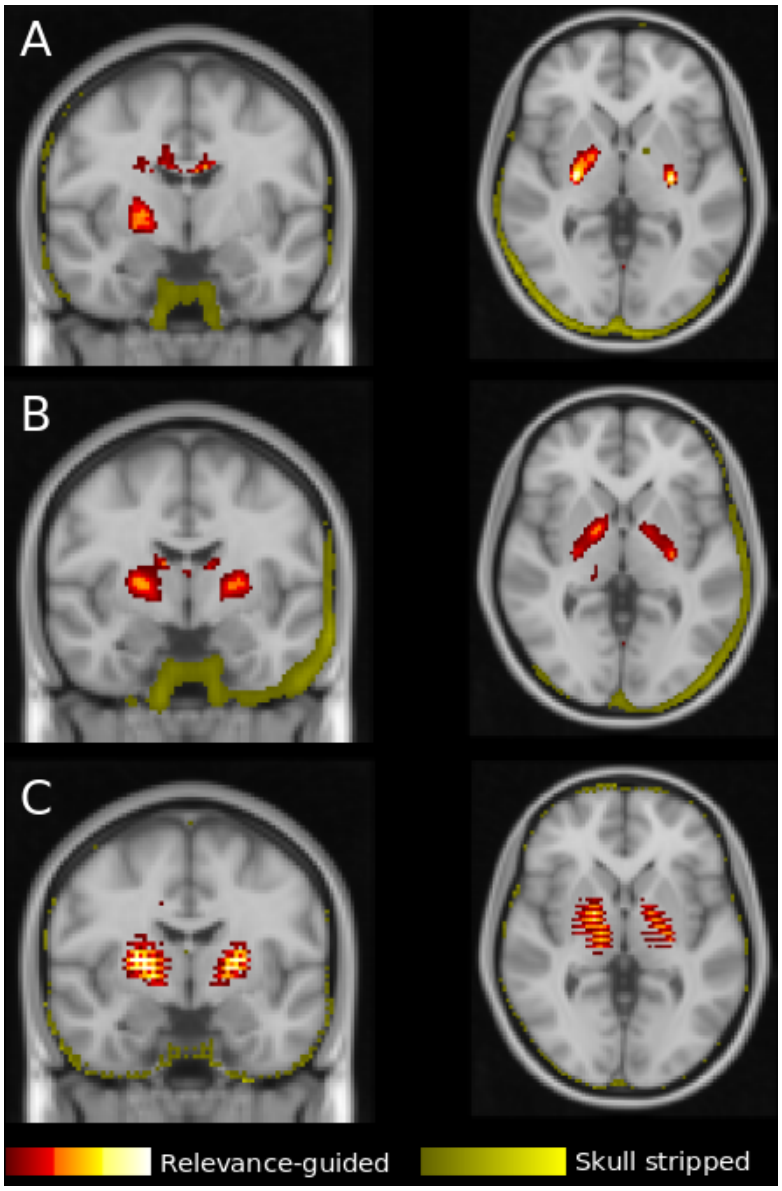


Figure 15: Mean of relevance maps from all patients and controls in the test set obtained by the relevance-guided model (red-orange) and the model with skull stripping (yellow) trained on (**A**) native R2* maps, (**B**) R2* maps affinely registered to MPRAGE sequence, (**C**) R2* maps non-linearly registered to MNI152 template. Mean maps are overlaid on MNI152 template. To speed up the training process all data was downsampled to 2mm isotropic resolution.

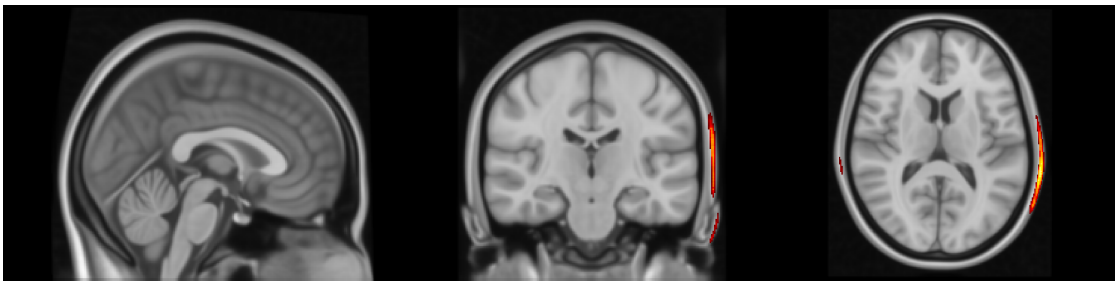


Figure 16: Mean of relevance maps obtained by the classifier model trained with magnetization-prepared rapid gradient-echo (MPRAGE) images (red-orange) overlaid on MNI152 template. Identified features concentrate at the skull.

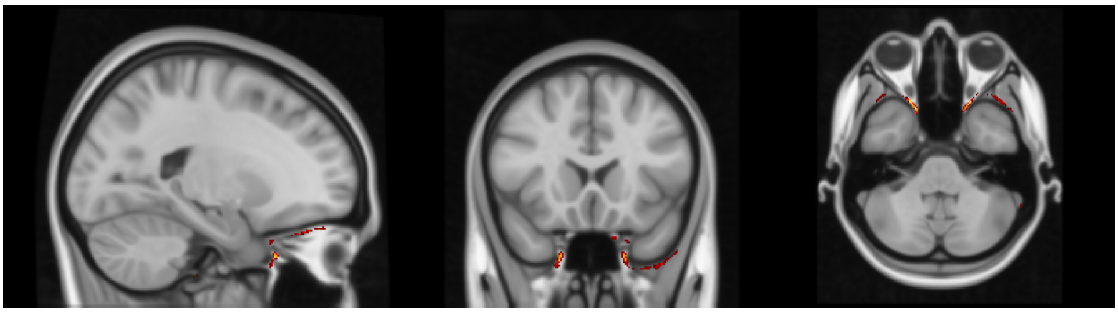


Figure 17: Mean of relevance maps obtained by the classifier model trained with skull stripped magnetization-prepared rapid gradient-echo (MPRAGE) images (red-orange) overlaid on MNI152 template. Identified features concentrate on skull stripped border regions.

classifier without relevance-guiding on MPRAGE images in native space yields mean heatmaps overlaid on MNI152 template in figure 16 for the configuration without skull stripping and figure 17 for the configuration with skull stripping. Features concentrate in brain border regions, which is even more obvious for the AD subject in figure 18, created with the model with skull stripping. Indicated regions are concentrated at the brain mask border.

In contrast, relevance-guided models trained with FLAIR, R2, R2* and MTR achieve high accuracies and the computed mean heatmaps in figure 19 indicate regions with high importance inside the brain.

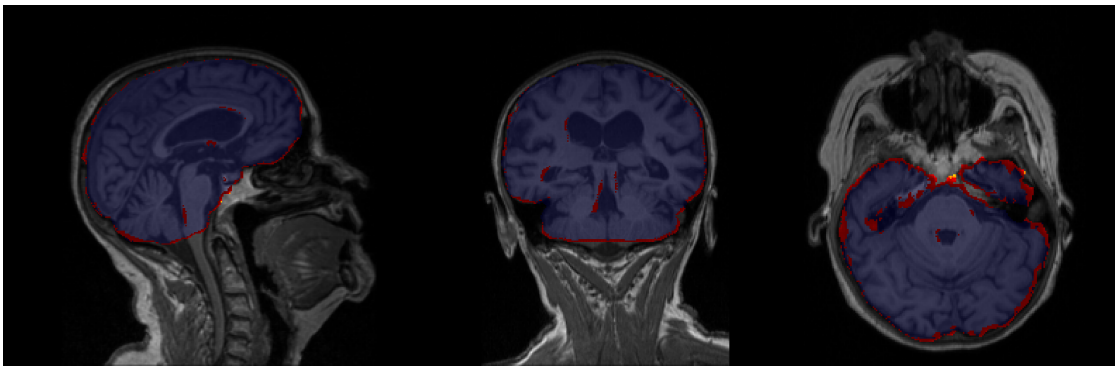


Figure 18: Relevance map obtained by the classifier model trained with skull stripped magnetization-prepared rapid gradient-echo (MPRAGE) images (red-orange) overlaid on subject's MPRAGE image with brain mask (blue). Identified features concentrate at skull stripped border regions.

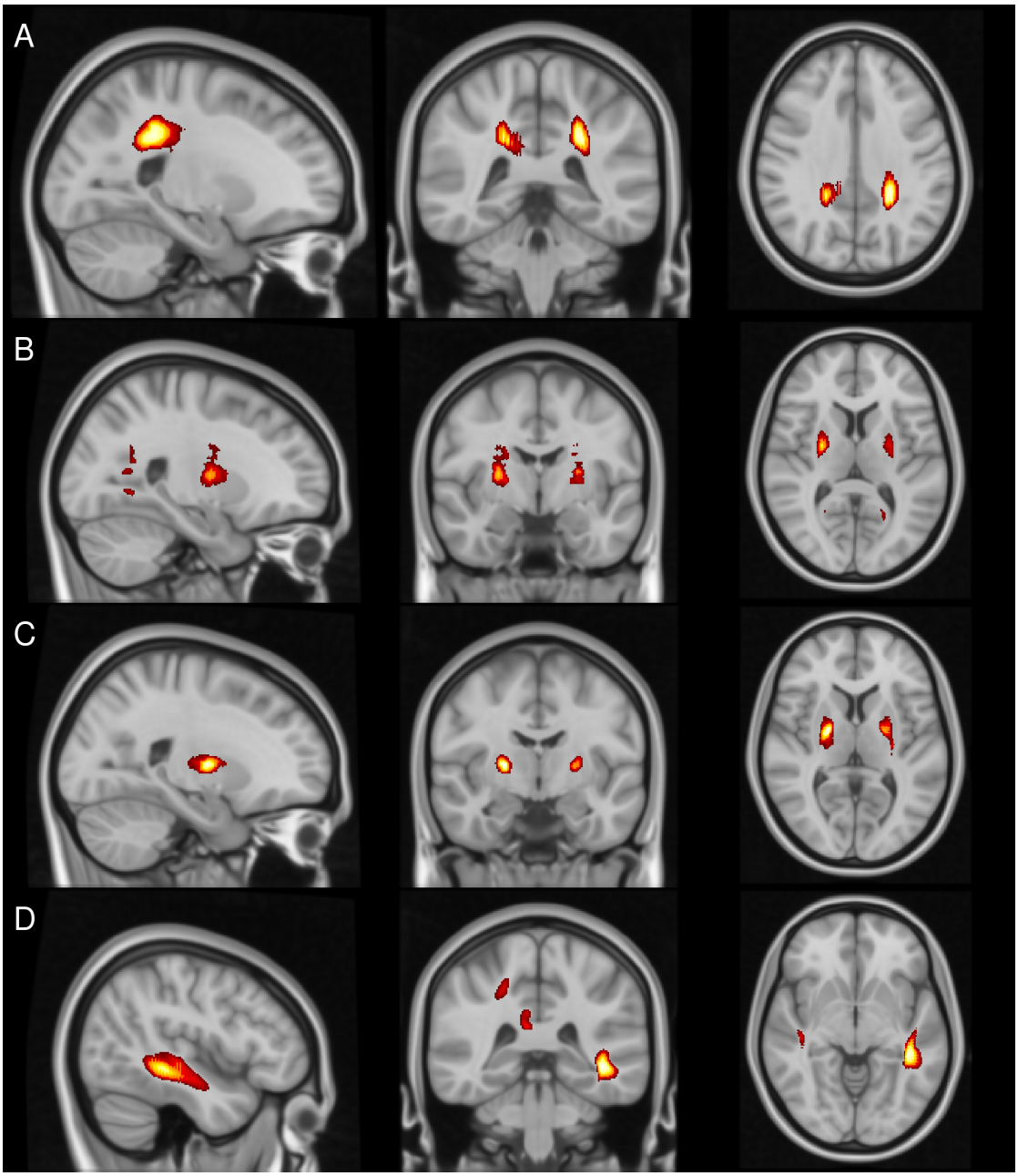


Figure 19: Mean of relevance maps (red-orange) obtained by the relevance-guided model trained with (**A**) fluid-attenuated inversion recovery (FLAIR) images, (**B**) R2 maps, (**C**) R2* maps and (**D**) magnetization transfer ratio (MTR) maps overlaid on MNI152 template. The regions of highest relevance are found in the brain parenchyma.

7 Discussion

This chapter analyzes and discusses the chosen network design, the achieved performances and the obtained heatmaps. Furthermore, the proposed method "relevance-guided training" is reviewed. Thereby the development framework and occured issues are examined. Finally, the chapter closes with prospects and recommendations for further work.

7.1 Networks

Designing deep learning networks is restricted by the memory available for training. The development was done on a single GPU with 12 GB of memory. Extending the classifier with the heatmap generator nearly doubles the necessary memory amount for the overall network, because for every layer in the classifier a corresponding redistribution layer in the heatmap generator is added. The heatmap generator has no trainable parameters on its own but shares the parameters with the classifier. Thus, the additional memory need is caused by intermediate computation results in the network graph and the gradient information. Furthermore, updating the network's parameters is done with a stochastic algorithm only taking a batch of input images into account at a training step. The consumed memory amount linearly increases with the batch size. Nevertheless, the batch size was set to eight to get good estimations of

the gradient and a smooth minimization progress of the loss function. Another design restriction is imposed by the 3-D nature of the used images, which inherently have a higher memory demand compared to the more common case of 2-D data. This becomes even more evident with the applied convolutional layers and their outputs. Each layer defines the count of output channel it produces, where each channel is again a volume. The volumes have the same dimensionality as the input volumes unless striding is defined. Striding is used to reduce the output dimensionality by shifting the kernel of the convolution operation not by only one unit but by the striding size. A striding size of 2×2×2 reduces the output size by half in each dimension, resulting in $1/8$ of the size of the input volume. For the classifier convolutional layers with this striding setting were used instead of pooling layers because relevance redistribution with a defined gradient can be more easily implemented for convolutional layers. Given all these criterions, the kernel size for the convolutional layers were chosen to be 3×3×3. Also the channel count was fixed to eight for these layers, which should be high enough to capture image features but helps to reduce the memory burden. Subsequent fully connected layer was designed to have 16 units in order to reduce memory need but also to help avoid overfitting during training with fewer trainable parameters.

The heatmap generator network is created dynamically by inspection of the classification network. In order to do so, the classifier's output tensor is used to identify the network layer it descended from. As the identified layer knows about its input tensor, through repeating this procedure the classifier graph can be reconstructed in reverse. From the gathered information the combination of the redistribution layers are derived. Each redistribution layers receives access to the corresponding forward layer, allowing for parameter sharing and an optimized implementation of the formulation. Tensors computed in the classifier's layers are reused in the redistribution layers whereever possible. The deep Taylor decomposition method was chosen because of the computationally fast way it can be implemented. The TensorFlow framework [63]

provides tensor operations and the corresponding gradient operations. These gradient operations are used making it feasible to create the heatmaps during training in reasonable time. On the other hand, gradient operations for these applied gradient operations need to be defined too, to allow for usage during training. Meaning the second derivative of the central tensor operation in each layer in the classifier needs to be defined. Furthermore the second derivative has to be unequal to zero because otherwise the error back-propagation would become zero too and divisions by zero would occur. The theorectical framework of deep Taylor decomposition implies the usage of the ReLU activation function throughout the network and constraints biases to be negative. Implementations for this work comply with these recommendations.

7.2 Classification Performance

The classification performances for all configurations are summarized in table 2, showing most resulting accuracies between 80 % and 88 %. Achieved accuracies do not compete with the high accuracies of up to nearly 100 % reported in [16, 15, 70]. In their review [16] Jo et al. pointed out issues with generalizability and reproducability. Results are highly dependant on the used data sets, where subjects might be preselected. In contrast, a data set acquired consecutively from patients with AD at the outpatient clinic of "Department of Neurology, Medical University of Graz, Austria" was used for this work.

Nevertheless, configurations with relevance-guiding show comparable or even better classification performance as configurations without it. Highest classification accuracy and highest ROC-AUC value (figure 13) were accomplished in stage three with relevance-guiding and FLAIR images. With the high contrast between the ventrical system and surrounding brain tissue in FLAIR images, brain atrophy is the most probably feature

picked up by the network. The mean heatmap for this configuration in subfigure **A** of figure 19 highlights regions near the occipital horn of the lateral ventricels. On the other hand, no classification performance is reported for the configuration with relevance-guiding and MPRAGE images in stage three because trainings were not successful. This might be because no discriminative features inside the brain could be learned by the network and suitable features are outside the brain. Hence, for comparison training sessions using only the classification network and MPRAGE images without and with skull stripping were done and yielded 77 % and 74 %. Trainings did converge with classification accuracies comparable to the other MR imaging modalities but heatmaps in figures 16, 17 and 18 reveal that learned features concentrate at the skull or the border of the brain mask. The learning process seems to be highly influenced by the skull stripping algorithm.

7.3 Heatmaps

Comparing the mean heatmaps for all configurations of stage one in figure 14 depicts how skull stripping influences the learned features of the network. Avoiding skull stripping renders learned features outside of the brain, at the skull, whereas skull stripping introduces new features used by the network. Introduced features concentrate at the border of the brain mask. This might be due to the exploition of the volumetric information incorporated by skull stripping. In contrast the relevance-guided method extracts features inside the brain, most likely a combination of brain atrophy and white matter hyperintensities. In order to test for the volumetric influence, the configurations in stage two were designed with different registration strategies and R2* maps. Again, for all registrations in figure 15 the relevant regions for classification with skull stripping are at the border between brain and background. For the non-linear registration configuration

the same MNI152 brain mask was applied on all subjects for skull stripping. Hence, all registered brains have the same volume and other features should have been learned by the network. Since this is not the case, the source of this effect remains unclear, however it was demonstrated that brain extraction algorithms can be biased by the patient cohort [71]. In contrast, when using the proposed relevance-guided approach and independently of preprocessing, the regions of highest relevance were found in the basal ganglia and the dentate nucleus. R2* is considered as a measure of iron content and histological and in-vivo studies have shown that brain iron strongly accumulates in these regions, in particular in AD patients when compared to NC [52]. Mean heatmaps in figure 19 obtained in the third stage show highly relevant features to be inside the brain. Positions of the features are in good agreement with AD studies [49, 50, 11, 54]. The reported accuracies paired with the per subject explanations make FLAIR, R2, R2* and MTR images suitable for the classification task. However, relevance-guiding with MPRAGE was not successful, an issue to be investigated in ongoing research. Avoiding skull stripping yielded important regions for classification at the skull shown in figure 16. Also using skull stripping for MPRAGE images again resulted in focusing on border regions between brain and background shown in figures 17 and 18.

7.4 Relevance-guided Training

In order to guide the training process the generated heatmaps are incorporated into the overall networks loss function. In equation 51 the heatmap generator output is multiplied with a predefined mask. The sum over the masked output is a measure for how much information inside the mask was relevant to the current prediction. With a softmax activation at the last layer of the classification network and the relevance conservation rule defined by the LRP framework the total relevance in the output is

always one. Hence, the amount of relevance inside and outside of the mask can be viewed as percentages. In this work the relevance loss function is designed to maximize the amount of relevance inside the mask. It is also possible to minimize the relevance outside of the mask. In fact, experiments trying to minimize the relevance outside of the mask did not come up with results as good as with the maximization approach. This might be due to the objections of the two loss functions. Whereas the binary cross entropy loss function encourages to find features in all regions of the input image, the minimization of relevance outside the mask hinders finding features. On the other hand maximizing relevance inside the mask encourages the search for features instead. However, the reasons for this behavior remain unclear and are up to further investigations.

7.5 Development Framework

With the introduction of Keras [65] creating deep learning networks became faster as the framework deploys a well structured application programming interface (API). New custom network layers can easily be developed and integrated. The training process itself is provided by the Keras engine. Keras was developed as a frontend and abstraction to multiple tensor libraries such as TensorFlow [63], Theano [72], CNTK [73] and NumPy [74]. As of now Keras is fully integrated in TensorFlow, superseding their initial approach of abstractions. Further development only happens within the TensorFlow project. The TensorFlow 1 backend used for this work has a big drawback. It enforces a "define and run" paradigm, where the whole graph has to be defined before it is run. Errors are therefore hard to debug as they mostly occur during runtime. A dedicated TensorFlow debug session can be used which enables defining break rules. If a rule criterion is met, the process is interrupted and analysis can be done with

special instructions on the command line. To overcome this problems the tensor library PyTorch [64] provides an imperative and a graph approach. Also eager execution, an imperative environment, was added to TensorFlow with the release of version 2. To make use of the imperative environment, the code for this work needs to be ported and adapted to the changes between TensorFlow version 1 and 2.

7.6 Conclusion

This work introduces a novel relevance-guided classification network for the task of AD classification. The network allows to focus on predefined image regions, presently brain tissue. The method incorporates brain masks into a term of the loss function during training, which eliminates the dependency of the prediction on the quality of those masks and consequently also offers the possibility to identify anatomical regions at a subject-level relevant for the classification decision.

The results of this explorative work demonstrate that the preprocessing of MR images is crucial for the feature identification by DNNs. Additionally, it shows that skull stripping is necessary to avoid identification of features outside the brain. However, this may introduce new features at the edge of the brain mask, which are subsequently used by the DNN for classification. The source of the newly introduced features remains unclear, but volumetric information might play a role in this. In contrast, when using the proposed relevance-guided approach and independently of preprocessing, the regions of highest relevance were found in the brain parenchyma and in anatomical more plausible regions, which were also found in histological and neuroimaging studies in patients with AD.

FLAIR, R2, R2* and MTR images are identified to be suitable for AD classification with

deep learning models. In conclusion, the results are in good agreement with findings from iron mapping studies and strongly support the hypothesis that the relevance-guided approach is minimizing the impact of preprocessing steps such as skull stripping and registration.

7.7 Summary and Further Work

In conclusion, this work demonstrated that classification in neurological disorders by using deep learning approaches heavily depends on the preprocessing of the MR images. Even more importantly, background features that are unrelated to the underlying pathology such as the brain extraction algorithm or voxels in the skull are not only contributing but highly relevant for the disease classification decision.

While most of the present work did not investigate and consider these aspects, the aforementioned issues are of high importance for deep learning based AI systems, to potentially become part of automated clinical diagnostic processes in future.

The results of this work raised a series of follow-up questions and more investigations about the heatmaps have to be conducted for a better understanding of deep learning based decision making. While the positions of the classification-relevant regions in these maps were histologically plausible, an interpretation about shape or texture is lacking and remains an open research issue. Also the proposed relevance loss function needs further systematic analysis. Additionally, as relevance-guiding is memory-heavy, only one image contrast was used per configuration. However, investigating multiple contrasts is promising because of the biophysically different and complimentary contrast generation mechanisms in MPRAGE, FLAIR, R2, R2*, and MTR. In this context,

it is not unexpected that combining these quantitative MRI parameters could (i) substantially improve the classification accuracy, (ii) will allow the identification of novel pathology-induced features beyond the current knowledge and (iii) also might reveal which MRI parameters are valuable for disease classification in a clinical setup with limited MRI scanning time.

Bibliography

[1] Dening T, Sandilyan M B. Dementia: definitions and types. *Nursing standard (2014+)*, 29(37): 37–42 (2015).

[2] World Health Organisation. Global action plan on the public health response to dementia 2017 - 2025. Technical report, World Health Organisation (2017).

[3] Alzheimer's Association. 2019 Alzheimer's disease facts and figures. *Alzheimer's & Dementia*, 15(3): 321–387 (2019).

[4] Alzheimer's Disease International. World Alzheimer Report 2019: Attitudes to dementia. Technical report, Alzheimer's Disease International, London (2019).

[5] Braak H, Braak E. Frequency of stages of Alzheimer-related lesions in different age categories. *Neurobiol Aging*, 18(4): 351–357 (1997).

[6] Braak H, Alafuzoff I, et al. Staging of Alzheimer disease-associated neurofibrillary pathology using paraffin sections and immunocytochemistry. *Acta Neuropathol*, 112(4): 389–404 (2006).

[7] Masters C L, Bateman R, et al. Alzheimer's disease. *Nat Rev Dis Prim*, 1: 1–18 (2015).

[8] Sluimer J D, Vrenken H, et al. Whole-brain atrophy rate in Alzheimer disease: Identifying fast progressors. *Neurology*, 70(19 PART 2): 1836–1841 (2008).

[9] Leung K K, Bartlett J W, et al. Cerebral atrophy in mild cognitive impairment and Alzheimer disease: Rates and acceleration. *Neurology*, 80(7): 648–654 (2013).

[10] Henneman W J, Sluimer J D, et al. Hippocampal atrophy rates in Alzheimer disease: Added value over whole brain volume measures. *Neurology*, 72(11): 999–1007 (2009).

[11] Ghadery C, Pirpamer L, et al. R2* mapping for brain iron: Associations with cognition in normal aging. *Neurobiol Aging*, 36(2): 925–932 (2015).

[12] Bulk M, Kenkhuis B, et al. Postmortem T2*-Weighted MRI Imaging of Cortical Iron Reflects Severity of Alzheimer's Disease. *J Alzheimer's Dis*, 65(4): 1125–1137 (2018).

[13] Hammernik K, Klatzer T, et al. Learning a variational network for reconstruction of accelerated MRI data. *Magn Reson Med*, 79(6): 3055–3071 (2018).

[14] Shelhamer E, Long J, et al. Fully Convolutional Networks for Semantic Segmentation. *IEEE Trans Pattern Anal Mach Intell*, 39(4): 640–651 (2017).

[15] Hosseini-Asl E, Keynton R, et al. Alzheimer's Disease Diagnostics by Adaptation of 3D Convolutional Network. In *Proc. - Int. Conf. Image Process. ICIP*, volume 2016-Augus, pages 126–130. IEEE Computer Society (2016).

[16] Jo T, Nho K, et al. Deep Learning in Alzheimer's Disease: Diagnostic Classification and Prognostic Prediction Using Neuroimaging Data. *Front Aging Neurosci*, 11 (2019).

[17] Montavon G, Samek W, et al. Methods for interpreting and understanding deep neural networks. *Digit Signal Process A Rev J*, 73: 1–15 (2018).

[18] Rathore S, Habes M, et al. A review on neuroimaging-based classification studies and associated feature extraction methods for Alzheimer's disease and its prodromal stages. *Neuroimage*, 155: 530–548 (2017).

[19] Pan S J, Yang Q. A survey on transfer learning. *IEEE Trans Knowl Data Eng*, 22(10): 1345–1359 (2010).

[20] Gilpin L H, Bau D, et al. Explaining explanations: An overview of interpretability of machine learning. In *Proc. - 2018 IEEE 5th Int. Conf. Data Sci. Adv. Anal. DSAA 2018*, pages 80–89. Institute of Electrical and Electronics Engineers Inc. (2019).

[21] Zeiler M D, Fergus R. Visualizing and understanding convolutional networks. In Fleet D, Pajdla T, et al., editors, *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, volume 8689 LNCS of *Lecture Notes in Computer Science*, pages 818–833. Springer Verlag (2014).

[22] Fong R C, Vedaldi A. Interpretable Explanations of Black Boxes by Meaningful Perturbation. In *Proc. IEEE Int. Conf. Comput. Vis.*, volume 2017-Octob, pages 3449–3457. Institute of Electrical and Electronics Engineers Inc. (2017).

[23] Simonyan K, Vedaldi A, et al. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *2nd Int Conf Learn Represent ICLR 2014 - Work Track Proc* (2013).

[24] Smilkov D, Thorat N, et al. SmoothGrad: removing noise by adding noise. *arXiv e-prints*, abs/1706.03825 (2017).

[25] Ribeiro M T, Singh S, et al. "Why should i trust you?" Explaining the predictions of any classifier. In *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, volume 13-17-Augu, pages 1135–1144. New York, New York, USA: Association for Computing Machinery (2016).

[26] Bach S, Binder A, et al. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS One*, 10(7): e0130140 (2015).

[27] Montavon G, Lapuschkin S, et al. Explaining nonlinear classification decisions with deep Taylor decomposition. *Pattern Recognit*, 65: 211–222 (2017).

[28] Goodfellow I, Bengio Y, et al. *Deep Learning*. MIT Press (2016). http://www.deeplearningbook.org.

[29] Nair V, Hinton G. Rectified Linear Units Improve Restricted Boltzmann Machines Vinod Nair. In *Proc. ICML*, volume 27, pages 807–814 (2010).

[30] Glorot X, Bordes A, et al. Deep Sparse Rectifier Neural Networks. *Proc 14th Int Conf Artif Intell Statisitics 2011*, 15: 315–323 (2011).

[31] Kiefer J, Wolfowitz J. Stochastic Estimation of the Maximum of a Regression Function. *Ann Math Stat*, 23(3): 462–466 (1952).

[32] Rumelhart D E, Hinton G E, et al. Learning representations by back-propagating errors. *Nature*, 323(6088): 533–536 (1986).

[33] Hornik K. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2): 251–257 (1991).

[34] Hochreiter S, Schmidhuber J. Long Short-Term Memory. *Neural Comput*, 9(8): 1735–1780 (1997).

[35] LeCun Y, Boser B, et al. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Comput*, 1(4): 541–551 (1989).

[36] Zhou Y T, Chellappa R, et al. Image Restoration Using a Neural Network. *IEEE Trans Acoust*, 36(7): 1141–1151 (1988).

[37] Hubel D H, Wiesel T N. Receptive fields of single neurones in the cat's striate cortex. *J Physiol*, 148(3): 574–591 (1959).

[38] Hubel D H, Wiesel T N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *J Physiol*, 160(1): 106–154 (1962).

[39] Hubel D H, Wiesel T N. Receptive fields and functional architecture of monkey striate cortex. *J Physiol*, 195(1): 215–243 (1968).

[40] Krizhevsky A, Sutskever I, et al. ImageNet classification with deep convolutional neural networks. *Commun ACM*, 60(6): 84–90 (2017).

[41] Samek, Wojciech and Wiegand, Thomas and Müller K R. Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models. *ITU J ICT Discov - Spec Issue 1 - Impact Artif Intell Commun Networks Serv*, 1(1): 1–10 (2017).

[42] Brant-Zawadzki M, Gillan G D, et al. MP RAGE: A three-dimensional, T1-weighted, gradient-echo sequence - Initial experience in the brain. *Radiology*, 182(3): 769–775 (1992).

[43] Wang J, He L, et al. Optimizing the Magnetization-Prepared Rapid Gradient-Echo (MP-RAGE) sequence. *PLoS One*, 9(5) (2014).

[44] Ashburner J, Friston K J. Voxel-based morphometry - The methods. *Neuroimage*, 11(6 I): 805–821 (2000).

[45] Sastre-Garriga J, Arévalo M J, et al. Brain volumetry counterparts of cognitive impairment in patients with multiple sclerosis. *J Neurol Sci*, 282(1-2): 120–124 (2009).

[46] Hedderich D M, Spiro J E, et al. Increasing Diagnostic Accuracy of Mild Cognitive Impairment due to Alzheimer's Disease by User-Independent, Web-Based Whole-Brain Volumetry. *J Alzheimer's Dis*, 65(4): 1459–1467 (2018).

[47] Bakshi R, Ariyaratana S, et al. Fluid-attenuated inversion recovery magnetic resonance imaging detects cortical and juxtacortical multiple sclerosis lesions. *Arch Neurol*, 58(5): 742–748 (2001).

[48] Tinauer C. *Parallelisierung monoexponentieller Fitroutinen mittels CUDA zur Messung von Relaxationszeiten klinischer Datensätze*. Bachelor's thesis, Graz University of Technology (2015).

[49] Knight M J, McCann B, et al. Quantitative T2 mapping of white matter: Applications for ageing and cognitive decline. *Phys Med Biol*, 61(15): 5587–5605 (2016).

[50] Luo Z, Zhuang X, et al. The Correlation of Hippocampal T2-Mapping with Neuropsychology Test in Patients with Alzheimer's Disease. *PLoS One*, 8(9): e76203 (2013).

[51] Langkammer C, Ropele S, et al. MRI for iron mapping in Alzheimer's disease. *Neurodegener Dis*, 13(2-3): 189–191 (2014).

[52] Ropele S, Langkammer C. Iron quantification with susceptibility. *NMR Biomed*, 30(4) (2017).

[53] Draganski B, Ashburner J, et al. Regional specificity of MRI contrast parameter changes in normal ageing revealed by voxel-based quantification (VBQ). *Neuroimage*, 55(4): 1423–1434 (2011).

[54] Ridha B H, Tozer D J, et al. Quantitative magnetization transfer imaging in Alzheimer disease. *Radiology*, 244(3): 832–837 (2007).

[55] Henkelman R M, Stanisz G J, et al. Magnetization transfer in MRI: A review (2001).

[56] Graham S J, Henkelman R M. Understanding pulsed magnetization transfer. *J Magn Reson Imaging*, 7(5): 903–912 (1997).

[57] Seiler S, Ropele S, et al. Magnetization transfer imaging for in vivo detection of microstructural tissue changes in aging and dementia: A short literature review. In *J. Alzheimer's Dis.*, volume 42, pages S229–S237. IOS Press (2014).

[58] Li F, Tran L, et al. A Robust Deep Model for Improved Classification of AD/MCI Patients. *IEEE J Biomed Heal Informatics*, 19(5): 1610–1616 (2015).

[59] Smith S M, Zhang Y, et al. Accurate, robust, and automated longitudinal and cross-sectional brain change analysis. *Neuroimage*, 17(1): 479–489 (2002).

[60] Smith S M, Jenkinson M, et al. Advances in functional and structural MR image analysis and implementation as FSL. In *Neuroimage*, volume 23 (2004).

[61] Siemonsen S, Finsterbusch J, et al. Age-dependent normal values of T2* and T2' in brain parenchyma. In *Am. J. Neuroradiol.*, volume 29, pages 950–955. American Society of Neuroradiology (2008).

[62] Liu Z, Pardini M, et al. Magnetization transfer ratio measures in normal-appearing white matter show periventricular gradient abnormalities in multiple sclerosis. *Brain*, 138(5): 1239–1246 (2015).

[63] Abadi M, Agarwal A, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems (2015). Software available from tensorflow.org.

[64] Paszke A, Gross S, et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc. (2019).

[65] Chollet F, et al. Keras. https://keras.io (2015).

[66] Brett M, Markiewicz C J, et al. nipy/nibabel: 2.5.0 (2019).

[67] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks. *J Mach Learn Res*, 9: 249–256 (2010).

[68] Kingma D P, Ba J L. Adam: A method for stochastic optimization. In *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.* International Conference on Learning Representations, ICLR (2015).

[69] Ruopp M D, Perkins N J, et al. Youden Index and optimal cut-point estimated from observations affected by a lower limit of detection. *Biometrical J*, 50(3): 419–430 (2008).

[70] Ehsan H A, Ghazal M, et al. Alzheimer's disease diagnostics by a 3D deeply supervised adaptable convolutional network. *Front Biosci - Landmark*, 23(3): 584–596 (2018).

[71] Fennema-Notestine C, Ozyurt I B, et al. Quantitative evaluation of automated skull-stripping methods applied to contemporary and legacy images: Effects of diagnosis, bias correction, and slice location. *Hum Brain Mapp*, 27(2): 99–113 (2006).

[72] Al-Rfou R, Alain G, et al. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688 (2016).

[73] Seide F, Agarwal A. CNTK: Microsoft's Open-Source Deep-Learning Toolkit. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 2135. New York, NY, USA: Association for Computing Machinery (2016).

[74] Van Der Walt S, Colbert S C, et al. The NumPy array: A structure for efficient numerical computation. *Comput Sci Eng*, 13(2): 22–30 (2011).