Ischemic Stroke Segmentation on CT-Perfusion Data using Deep Learning Methods

Master Thesis in Computer Science

submitted by

Florian Thamm

born May 21, 1994 in Munich, Germany

Department of Computer Science

Faculty of Engineering

Friedrich-Alexander-University of Erlangen-Nuremberg

Supervisor: Prof. Dr.-Ing. habil. Andreas Maier

Co-Supervisor: Leonid Mill, M.Sc.

Co-Supervisor: Dr. rer. nat. Markus Jürgens

Eidesstattliche Erklärung

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Die Richtlinien des Lehrstuhls für Masterarbeiten habe ich gelesen und anerkannt, insbesondere die Regelung des Nutzungsrechts.

Florian Thamm

Erlangen, den 22.10.2019

Übersicht Die moderne Computer Tomographie (CT) bietet mit der CT Perfusions Aufnahme die Möglichkeit, die Durchblutung von Hirngewebe quantitativ zu messen. Unter anderem kann dadurch die Kernregion eines ischämischen Infarktes segmentiert werden. Dies erfolgte bisher meist durch Schwellwertsegementierungen, wobei in Veröffentlichungen verschiedene Schwellwerte und Parameter empfohlen werden. Deep Learning (DL) stellt eine Alternative zur klassischen Bildverarbeitung dar und findet nun auch Anwendung in der Segmentierung des Kerninfarktes. Die vorliegende Arbeit stellt eine Erweiterung der U-Net Architektur vor, die die Rohdaten aus der Perfusionsaufnahme komprimiert und mit den Parameterkarten zur Segmentierung nutzt. Darüber hinaus beschäftigt sich die Arbeit auch mit der Vorverarbeitung zur Normalisierung der Daten, sodass ein Training aus unterschiedlichen Datenquellen möglich ist. Ferner wird eine Methode vorgestellt, die mithilfe von Korrelationsanalysen auf die Auswirkung verschiedener achitektureller Entscheidungen der DL Modelle eingeht. Desweiteren wird ein Algorithmus vorgestellt, mit dessen Hilfe die relevantesten Perfusionsparameter bestimmt werden können. In einer experimentellen Auswertung konnte weder die Funktionalität des Algorithmus noch Vorteile durch die erweiterte Architektur bestätigt werden. Ein Auswertung zeigt zudem, dass die DL Segmentierung (mit einem Dice-Score von 0.425) im Vergleich zur Schwellwertsegmentierung (mit einem Dice-Score von 0.198) zu einer besseren Segmentierung führt.

Abstract Modern Computed Tomography (CT) offers with CT Perfusion the quantitative measurement of the blood circulation in brain tissue. Among other things, the core region of an ischemic infarct can be segmented. So far, this has mostly been done by threshold segmentations, whereby literature suggests various threshold values and parameters. Deep learning (DL) is an alternative to classic image processing and is now also used in the segmentation of the infarcted tissue. The thesis at hand presents an extended version of the U-Net architecture, which compresses the raw cine data from the perfusion acquisition and uses the compressed image alongside the perfusion parameter maps for the final segmentation. This work also presents a preprocessing pipeline for the normalization of the data which enables the training from various data sources. Furthermore, a method is proposed that uses correlation analysis to examine the impact of different architectural decisions on the considered DL models. Moreover, an algorithm is introduced that can be used to determine the most relevant perfusion parameters. In an experimental evaluation, the functionality of the algorithm as well as advantages of the extended architecture could not be confirmed. An evaluation has shown that the DL segmentation (with a Dice score of 0.425) leads to better results compared to the threshold segmentation (with a Dice score of 0.198).

Contents

Abbreviations

AIF	Arterial-Input Function
AUC	Area under the Receiver Operator Curve
AVG	Average Intensity Projection
BASE	Baseline Projection
BatchNorm	Batch Normalization
CBF	Cerebral Blood Flow
CBV	Cerebral Blood Volume
CE	Cross Entropy
CNN	Convolutional Neural Network
CRC	Channel-wise Relevance Counting
CSF	Cerebrospinal Fluid
CT	Computed Tomography
CTA	Computed Tomography Angiography
CTP	Computed Tomography Perfusion
DL	Deep Learning
DWI	Diffusion Weighted Imaging
FDA	Food and Drug Administration
FN	False Negative
FP	False Positive
GP	Gaussian Process
HU	Hounsfield Unit
ISLES	Ischemic Stroke Lesion Segmentation
Lin	Linear Interpolation
LRP	Layer-wise Relevance Propagation
MICCAI	International Conference on Medical Image Computing and Computer Assisted Intervention
MIoU	Mean Intersection over Union
MIP	Maximum Intensity Projection
MRI	Magnet Resonance Imaging
MTT	Mean Transit Time
NN	Nearest Neighbor
Pad	Padding at Image Borders
Prec	Precision

Rec	Recall
ReLU	Rectified Linear Unit
RGB	Red, Green and Blue Channel
TAC	Time-Attenuation Curve
TMAX	Time to Peak of the Residue Function
TN	True Negative
TP	True Positive
TTD	Time to Drain
TTP	Time to Peak of Time-Attentuation Curve
TTS	Time to Start
UpConv	Upsampling combined with Convolutional Layer
VS	Volumentric Similarity

Chapter 1

Introduction

1.1 Acute Strokes and Imaging

The global second leading cause of death and the third leading cause of disability are cerebrovascular accidents, also known as strokes [Joh16, Don08]. Strokes, in general are defined as an abrupt onset of a focal neurological deficit caused by a lack of cerebral blood supply. Either spontaneous ruptures or artery occlusions can be causes of an undersupply of the cerebrovascular parenchyma, which allows a classification of strokes into either the hemorrhagic or the ischemic type. Depending on the appearance of the occlusion a further sub-classification can be done, dividing the ischemic stroke into thrombotic, embolic or microartery kind [Wit12, Moz16].

The treatment of patients suffering acute ischemic strokes vastly differs to hemorrhagic strokes. Thrombolysis, which is a therapeutic dissolution of blood clots using medications (thrombolytics), is usual in ischemic cases but might be lethal in hemorrhagic ones [Pea02, Bam16, Psc11]. Therefore, the classification of the underlying type is a question of central interest for the resulting treatment. In ischemic cases of larger onset times ($\geq$ 4.5 h) thrombolysis remains limited, as this involves higher hemorrhagic risks. The onset time the time that elapses from the beginning of the stroke accident. Therefore in cases of large occluded arteries the mechanical/surgical thrombectomy, which is a mechanical disruption or retrieval of the clot, can be the superior and is the preferred method [Pea02, Tor13, ET17, Bam16]. As a consequence the ability of localizing the ischemic stroke lesion on artery-level is crucial for a successful mechanical thrombectomy and therefore for the whole treatment in terms of its clinical outcome [Tor13, Pal15].

Computed tomography (CT) and magnetic resonance imaging (MRI) represent the most important modalities addressing the diagnosis, management and in particular the exact localization of acute strokes. Image 1.1 shows examples of CT and MR images of a patient suffering a stroke.

MRI has gained high acceptance in the evaluation of acute stroke, specifically diffusion-weighted imaging (DWI), shown on the right of figure 1.1, which outperforms typical CT applications in terms of precision, but is rather difficult in clinical practice [Sch02, Lan00]. MR scanners are more expensive in either the purchasing and the maintenance, thus to a lesser extend available and need signficantly longer. In the field of CT several applications exist: 1) Unenhanced CT, shown on the left in figure 1.1 is widely used as a first-line imaging tool to identify hemorrhagic insults. 2) CT Angiography (CTA) is mainly used in the identification of intravascular thrombi that can be targeted for thrombolysis and/or mechanical thrombectomy [Tor13, Gon11]. 3) CT Perfusion (CTP), likewise DWI and in contradiction to unenhanced CT and CTA, is commonly used to localize stroke lesions, by characterizing the blood flow on tissue-level. This flow is described by a variety of parameters, including the cerebral blood flow (CBF), the cerebral blood volume (CBV), shown in the center of figure 1.1, the mean transit time (MTT) and other measures like time to peak (TTP) or time to drain (TTD). The parameters describing the blood perfusion on pixel-basis provide an insight into the delivery of blood to the brain parenchyma and enables the distinction between penumbra and the core of the critically infarcted tissue, which is of high importance for further revascularization procedures. The penumbra describes the hypoperfused (undersupplied) brain parenchyma, that can be rescued when reperfusion is established in a timely fashion, which gets irreversibly infarcted otherwise [Bam16]. The term core describes the parenchyma that is already irreversibly infarcted and cannot be recovered by reperfusion. CTP and DWI can be considered as methods to identify patients thought to be optimal candidates for reperfusion therapies, such as the mechanical thrombectomy [Gon11, Lin16b, Tor13].

In conclusion the imaging of acute strokes demands four critical questions which can be answered by choosing the right CT application [Tor13]:

- Is there hemorrhage?
  $\Rightarrow$ CT unenhanced

- Is there an intravascular thrombus that can be targeted for thrombolysis?
  $\Rightarrow$ CT Angiography

- Is there a “core” of critically ischemic irreversibly infarcted tissue?
  $\Rightarrow$ CT Perfusion

- Is there a “penumbra” of severely ischemic but potentially salvageable tissue?
  $\Rightarrow$ CT Perfusion

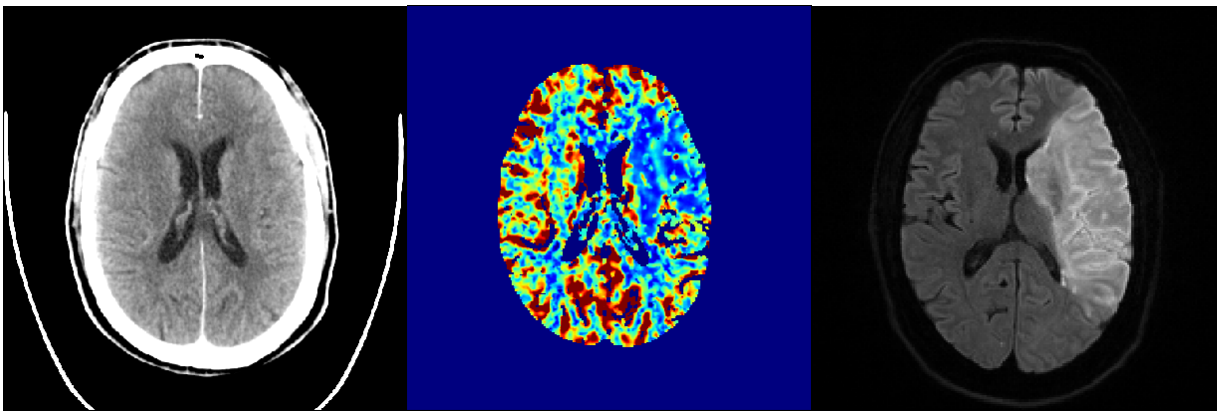

Figure 1.1: CT and MR scans of the same patient suffering an ischemic stroke. Left: Unenhanced CT Acquisition (Base); Center: CT Perfusion Acquisiton (Cerebral Blood Volume) shows the lack of blood supply in one brain hemisphere; Right: Follow-up Diffusion-weighted MR image showing the infarcted core as a bright region.

1.2 Problem Statement

Since the segmentation of the core and the penumbra on CTP image data is not standardized, different thresholds for either the core or the penumbra, have been proposed as state-of-the art methods, while most of them are compared with the DWI as the gold standard [Win07, Win06, Sch08, Soa10, Cam11]. These methods do not only differ in their threshold levels, but also on which parameters they are working on. However, these thresholding methods do neither consider anatomical structures nor spatial relations of the infarcted tissue itself, which may introduce artifacts in terms of small noisy clusters, causing a mismatch between CTP and DWI. In order to counteract that drawback, it is not unusual to extend the segmentation process by clustering methods [Biv11, Biv13].

Image segmentation in general is a well studied field in image processing. However, it is still a challenging task for many applications. Especially in medical applications like the segmentation of ischemic infarcted tissues, high precision results are required to provide the best possible patient care and treatment [Ras13, Sze10]. As a consequence of the high difficulty, many algorithms and methods were invented in the last years, but deep learning methods first and foremost gained higher attention for their superior performance over classic approach as thresholding, e.g [Bad17, Cir12, Lon15, Ron15]. In contrast to threshold approaches, deep learning methods incorporate spatial information and context into the segmentation.

Deep learning approaches and their success depends highly on the underlying data. CT Perfusion data with DWI follow up annotations as ground truths of patients suffering ischemic

strokes are rarely available due to the nature of such a specific case (ischemic, follow up etc.), the requirement of advanced equipment (CTP, MRI) and the data protection regulations nowadays. As a consequence, one single data source, for example a single clinic, usually can barely deliver enough data to enable the development of a reliable and robust deep learning model. The acquisition on different data sources allows a growth on suitable training datasets on one hand, but on the other hand increases the inter-subject variance as well, which results in different image appearances, since different clinics have different scanners, procedures and staff. Thus, deep learning models not only need to generalize over the segmentation task itself, but as an additional side task they must adapt to varying surrounding circumstances, which leads into a quality decrease. Furthermore the quality of the datasets, in terms of the general image properties (resolution, noise level, etc.) or the registration accuracy of the MR volume onto the CT volume are influencing factors that must be taken care of. Regarding the registration aspect in specific, recently published works on the MICCAI ILSES Challenge 2018 [Son18, Mai17, Kis13] showed weaknesses. Section 3.1 contains a detailed analysis of the data used in this challenge. Despite the ISLES Challenge, other recently published approaches still left space for further research, because they were either using non state-of-the-art segmentation architectures [Rob18] or were using a small number of different perfusion parameters [Luc18, Abu18].

The problems that currently appear with the core segmentation in practice can be summarized as follows:

1. Threshold Segmentation
   (a) Inconsistent threshold levels
   (b) Inconsistent use of parameter maps
   (c) Noisy clusters, due to missing consideration of anatomy
2. Deep Learning Segmentation
   (a) Lack of CTP datasets with DWI follow ups necessary for a robust training forces the use of different data sources
   (b) Inter data source variance requires the models to generalize to different dataset conditions (e.g. different scanners)

One aim of this thesis is to investigate the actual relevance of all different perfusion parameters and their subsets on the segmentation of the core, since many methods have been proposed with no consistent opinion about that, which addresses problem 1(a) and 1(b). By using Neural Networks

for the segmentation, anatomical structures are involved in the learning process, which addresses the problem 1(c) indirectly. The data pool used in this thesis was built from three different sources (in total 115 cases), each with their own characteristics. Focusing on problem 2(a) and 2(b), this thesis also aims to find a way of processing the data onto a common basis, such that the data can be used to train a deep learning model.

1.3 Contribution

The contributions of the thesis at hand are the following:

- Analysis of the CTP data and labels from various sources (Section 4.1.2)
- Image processing pipeline that allows the integration of different data sources into the training set (Section 4.1.3)
- Segmentation of the core of the critically infarcted tissue on CTP data using
  - the U-Net architecture (Section 4.2.1)
  - the U-Net architecture combined with a 4D-Extractor (Section 4.2.2)
- Systematic approach for model parametrizations based on statistics and Bayesian optimization (Section 5.1)
- Analysis regarding the relevance and the impact of various subsets of all perfusion parameters on the segmentation outcome (Section 5.2)
- Evaluation and Analysis of the models with a comparison of state-of-the-art thresholding methods (Chapter 6)

Chapter 2

Medical Principles

The aim of this chapter is to outline the fundamentals concerning the segmentation on CTP data. Section 2.1 covers the CTP acquisition process with a description of its perfusion parameters. Section 2.2 briefly summarizes the clinical assessment and the state-of-the-art segmentation of the critically infarcted tissue, also known as 'core'.

2.1 CT Perfusion Fundamentals

2.1.1 Acquisition

One of the most important technologies in medical imaging is the Computed Tomography (CT), which allows physicians to have high-resolution insights into the human body [Mai18]. Not only native CT represents a major role in diagnostics but also many derived methods were invented, such as cardiac imaging, angiography and dual-energy, each with a significant impact on the medical research area [Hsi09]. One of them, the CT Perfusion (CTP), provides in-depth information about the pathophysiology of acute strokes [Gon11]. The idea of CTP is to describe the blood-perfusion of a given volume (head), with characteristic parameters on pixel basis. The so called perfusion parameters are cerebral blood flow (CBF), cerebral blood volume (CBV), time to maximum of the residue function (TMAX), mean transit time (MTT), time to peak (TTP), time to drain (TTD), time to start (TTS), average intensity projection (AVG), maximum intensity projection (MIP) and base volume (BASE). Every parameter is described in section 2.1.2. The CTP yields volumes consisting of several channels, where each represents one perfusion parameter.

The acquisition of a CTP volume starts with the injection of a contrast bolus, during a continuous acquisition of the region of interest. The covered spatial volume depends on the

manufacturer and the specifications of the scanner. The complexity relies in the acquisition of large regions (whole head) with sufficient temporal sampling, to observe the perfusion on different slice levels while trying to keep the dosage as low as possible. Several techniques were proposed to cover the problem of covering enough volume. One way to cover large regions at once is to simply provide detectors that are large enough to acquire the whole region of interest. Another way is a techniques called “toggle table“, illustrated in figure 2.1, where the table repeatedly moves back and forth, alternating between two different cine volumes. The helical mode is working similarly but uses continuous spiral scans, where the table moves back to the starting position once the scanner has reached the regions end [Bam16]. Further, the application of two boluses and therefore two acquisitions, the so called “two slab“ method, is again, a way to enlarge the covered area. [Gon11, dL08].

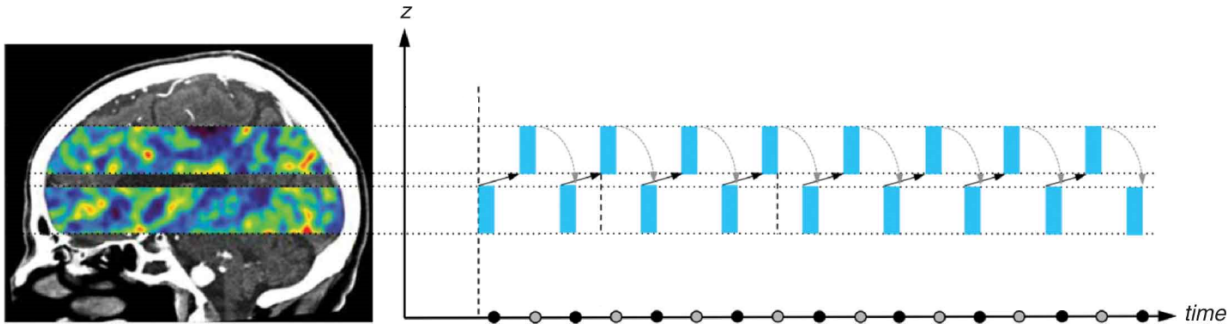

Figure 2.1: CT Perfusion acquisition with the toggle table mode. Left: A sagittal image with the covered regions, with a gap to better illustrate both positions. Right: A plot to show the detector coverage of both regions dependent to the time. The arrows are denoting the transition from one slab to next one. [Bam16].

Followed by the reconstruction of the volumes, several pre-processing steps are needed to be done in order to finally determine the perfusion parameters. Non-correctable motion artifacts are one of the main challenges that occur in the CTP procedure, and must be detected in advance to any other further proceedings, otherwise these artifacts can invalidate the study [dL08]. After acquiring the volume, the attenuation per voxel can be tracked over the time axis. This function is called the time attenuation curve (TAC) per voxel, illustrated in figure 2.2. Its shape and height depends on inflowing contrast medium as well as on local properties of the capillaries and the parenchyma. The inflow is described by the arterial input function (AIF) which represents the arterial contrast agent inlet of one of the arterial vessels supplying the region of interest [Hoe04, Ram14, dL08]. By involving the AIF in the subsequent computations, variations in the bolus injection protocols and patient physiology are allowed. The detection of one of these arteries is necessary as well, either automatically or manually.

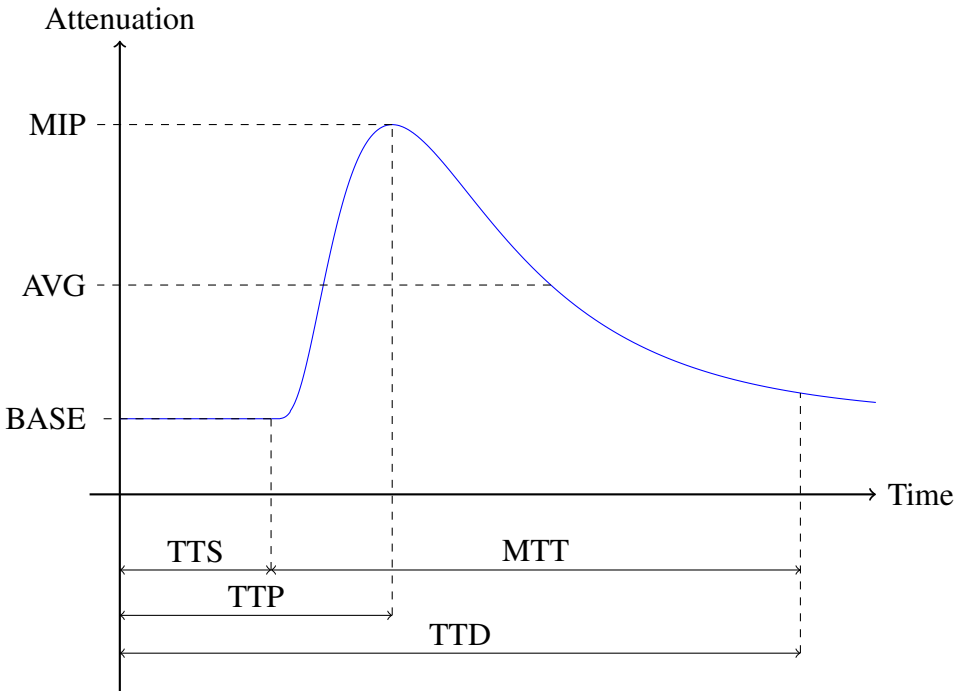

Figure 2.2: Attenuation-time curve per region with delineated perfusion parameters.

2.1.2 Perfusion Parameters

After the pre-processing the actual perfusion parameters can be computed. Some of them (BASE, MIP, AVG) can be computed directly on the cine data. In order to quantify the CBV, TMAX, TTP, TTD, TTS, MTT or the CBF maps, two approaches exist: (1) the direct measurement based on the Fick principle [Fic55] or, (2) the deconvolution method directly on the TAC per voxel. [Ram14, Hoe04, Hsi09, Abe10]. The TAC is denoted with $b(t)$ and can be expressed using the AIF $a(t)$ and the residue function $r(t)$, which describes the fraction of tracer still present in the capillaries at time t with

$$b(t) = a(t) * r(t) \tag{2.1}$$

where $*$ denotes the convolution operation. The deconvolution between the TAC and the AIF returns the residue function $r(t)$ [Bam16]. In this thesis a software prototype of syngo.via by Siemens Healthcare GmbH that utilizes the deconvolution method, was used in order to calculate every map except MIP, Base and AVG. Figure 2.3 displays the perfusion parameters used in this thesis on a patient suffering an ischemic stroke in the left hemisphere.

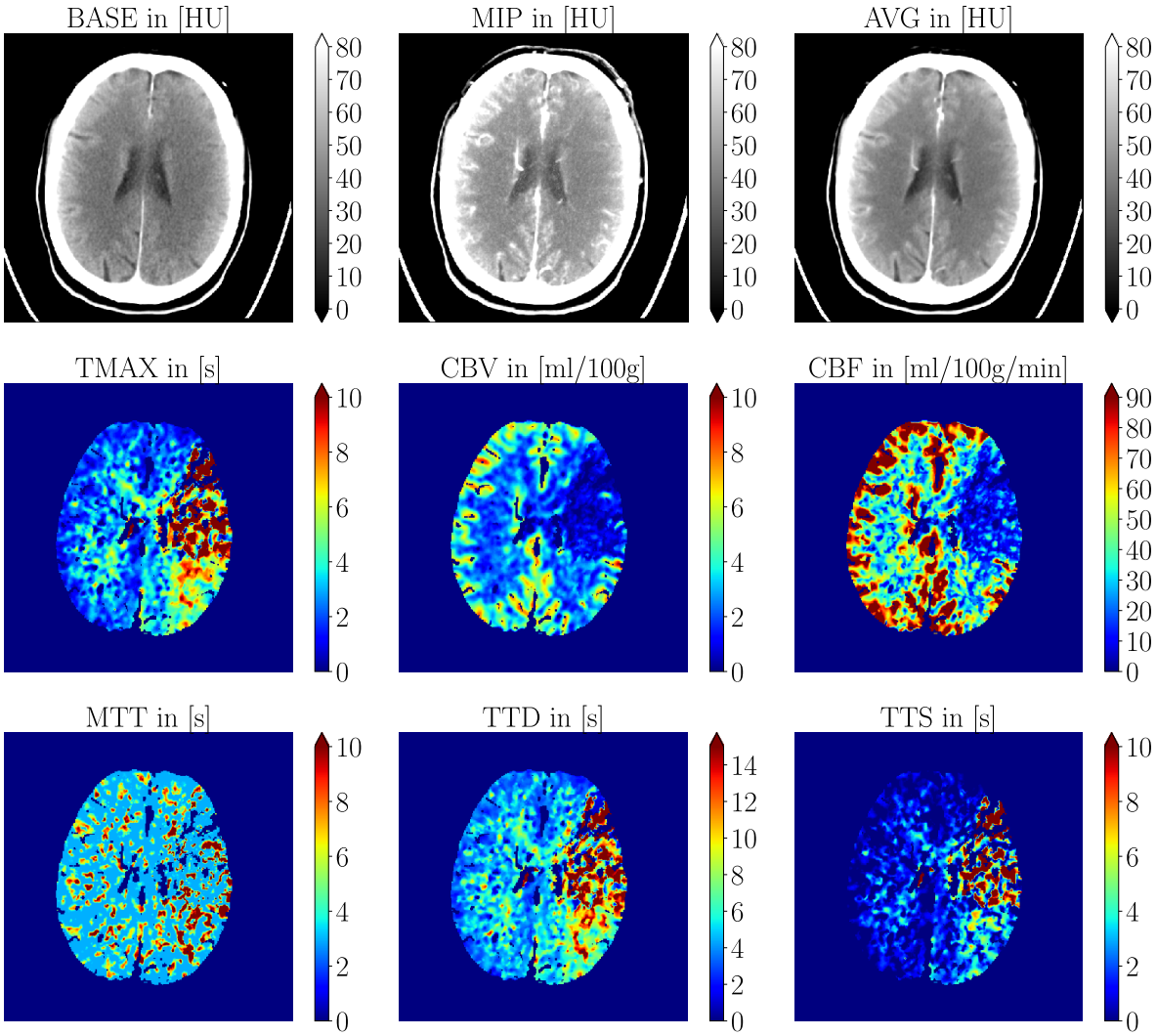

Figure 2.3: Example of different perfusion parameters on the same slice of one patient windowed in their typical range. The displayed range is shown next to the respective channel.

Cerebral Blood Flow (CBF)

CBF is defined as the amount of blood that was moving through a given volume of brain tissue in a specific time period. It is measured in ml blood / 100g brain tissue / min and is calculated based on the residue function $r(t)$ [Ram14, Bam16].

Cerebral Blood Volume (CBV)

CBV describes the total volume of flowing blood in a given volume of brain measured in ml blood / 100 g brain tissue [Ram14]. The CBV map is computed using the deconvolution method. Likewise CBF, CBV is computed with the residue function $r(t)$.

Mean Transit Time (MTT)

MTT, defined as the mean transit time of flowing blood from the arterial inlet to the venous outlet is measured in seconds. Staying close to reality, the blood traverses through different paths of the vascular structure, which means the actual transit time is distributed over that range. Therefore, the MTT value is simply the mean of this distribution [Hsi09, Ram14]. The MTT is computed with the integral over the residue function [Bam16]. CBV, CBF and MTT are connected by the central volume principle (image 2.4) [Mei54, Hsi09].

$$\text{CBV} = \text{CBF} \times \text{MTT} \tag{2.2}$$

Time to Start (TTS)

TTS indicates the time in seconds, from the beginning contrast agent injection to actual contrast enhancement.

Time to Drain (TTD)

TTD measured in seconds, is the time period beginning with the acquisition to the wash-out time point of the contrast agent. TTD is simply the sum of TTS and MTT [Hak19].

$$\text{TTD} = \text{MTT} + \text{TTS} \tag{2.3}$$

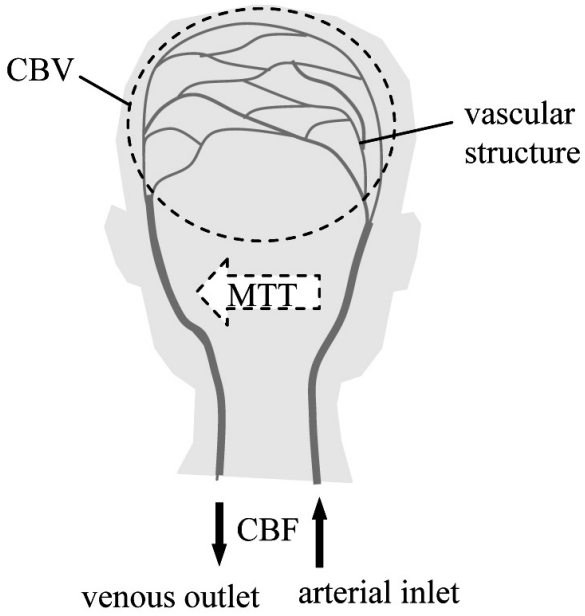

Figure 2.4: Illustration of the transit time due to different path lengths and the relation to CBV and CBF based on the central volume principle [Hsi09].

Time to Peak (TTP)

TTP represents the time it takes from the bolus arrival to the peak of the attenuation intensity. TTP is influenced by the shape of the AIF. The AIF is shaped, amongst others, by the flow/amount rate of the contrast injection and the vasculature and its hemodynamic response between the point of injection and the tissue of interest [Bam16]. Therefore, TTP can vary between different patients, just because of different AIF choices. As a consequence, TTP was excluded from any computations in the thesis at hand.

Time to Maximum (TMAX)

TMAX, shown in figure 2.3 refers to the time to the maximum value of the residue function $r(t)$, or in other words the maximum of tracer still present in the capillaries [Hak19, Bam16].

Base (BASE)

BASE describes the attenuation of the tissue before any contrast enhancements by the bolus and is measured in Hounsfield Units (HU). The base map can be considered as a native (non-contrasted plain) CT scan.

Average Intensity Projection (AVG)

AVG, which stands for the average intensity projection, is the average intensity value of the attenuation curve, measured in HU.

Maximum Intensity Projection (MIP)

MIP, the maximum intensity projection, displays the maximum intensity value over time of the attenuation curve measured in HU.

2.2 Penumbra and Core

2.2.1 Clinical Assessment

During the first hours after the ischemic onset, the affected brain tissue can be classified in three different regions [Gau16].

1. The ischemic core of the irreversibly infarcted tissue, that cannot be rescued and remains damaged even if reperfusion is established [Bam16].

2. The penumbra, which is the "tissue at risk" for infarction in the absence of reperfusion but can be saved otherwise [Sch08].

3. The oligemea region, which presents a reduction in blood flow that is not sufficiently critical to result in brain infarction [Gau16].

The elapsed time after stroke onset to the reperfusion is the leading influence factor of the dynamic relation of these three regions. As the time passes, the ischemic core grows and progressively arrogates the penumbra tissue, leading to the concept of "time is brain". Between 24 h and seven days after the stroke onset a lesion growth of the ischemic core region can be observed. Unlike the reperfusioned and rescued penumbra a "secondary lesion" occures by inflammatory reactions of the core [Gau16]. Follow-up images (24 h after the ischemic onset) of two patients with inflammatory reactions are shown in figure 2.5.

As listed in section 1.3, this thesis focuses on the segmentation of the core region. Thus, methods used to clinically assess the region are described in the following. Technically speaking, this section describes methods used to differentiate between the core region and the combination of the penumbra and the oligemia. One has typically two choices for the identification of the core region namely MR-Based protocols (DWI) and CT-Based protocols (CTP) [Bam16]. The

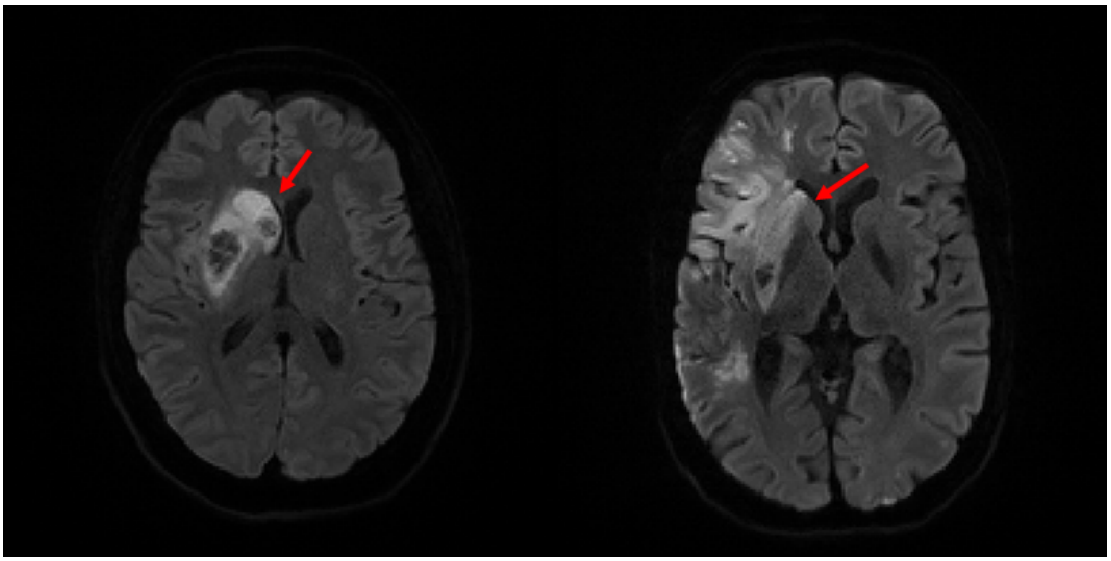

Figure 2.5: DWI follow-ups of two patients suffering ischemic strokes. The ischemic core appears bright on DWI images, including the inflammatory region, which cause tissue shifts that reveals in asymmetric deformations of the cerebrospinal fluid. The red arrows indicate the parenchyma shifts caused by the core inflammatories.

diffusion-weighted imaging (DWI) plays a central role in stroke therapy [Kra09]. In patients with acute ischemic strokes, visually tense lesions in the DWI image closely approximates the ischemic core, which appear as bright clusters as shown in figure 2.5 and 1.1 [Bam16]. The DWI essentially describes the diffuseness by microscopic motion of water protons, as an intrinsic property of tissue [Bam03]. An alternative to DWI is the CT Perfusion, a method to identify the ischemic core, driven by the hypothesis that cerebral vascular autoregulations are lost in infarcted regions. Consequently regions of infarcted tissue must have reduced CBF and CBV values [Bam16]. In clinical practice the actual workflow, a CT based workflow might happens as the following: An unhenhanced CT acquisition is performed to exclude a potential hemorrhagic case. In cases of ischemic strokes,a CTA is used to localize the clot and a CTP is used to determine the size of the penumbra and core. Based on the results, revascularization procedures, e.g. mechanical thrombectomy and/or a thrombolysis are set into operation. After 24 h of the stroke onset and revascularization, treatment outcome can be reviewed with a DWI acquisition, which reveals the final outcome of the therapy.

In many recent deep-learning and classic image processing publications ([Son18, Mai17, Kis13, Luc18, Abu18, Win07, Win06, Sch08, Soa10, Cam11]), CTP data were often compared with DWI as the gold standard. Typically the DWI data was acquired as follow up, to verify the treatment outcome. Considering the mentioned inflammatory shifts in the time period 24 h after

the onset, an error is being introduced. Thus, labels annotated on follow ups are not perfectly representative of the core within the first 24 h of onset. In addition to that, if the reperfusion therapy, say the mechanical thrombectomy failed (see image 1.1), penumbra tissue evolves to infarcted core, which again is one factor that causes a mismatch.

2.2.2 State-of-the-art Segmentation

In the past many works have been published addressing the segmentation of the core and the penumbra based on various parameter maps. In fact, state-of-the-art segmentation is still thresholding of parameters. The reasons are, firstly the transparency of the segmentation from the physicians perspective is much higher, than using black-box deep learning systems. Secondly, black-box systems for the automated segmentation of, for example the infarct core are harder to be approven e.g. by the Food and Drug Administration (FDA). However, the threshold values presented by Abels et al. in [Abe10] are the default threshold and hence recommended values in the CT Perfusion Software syngo.via (NeuroPerfusion) by SIEMENS Heathcare GmbH. Since this thesis focuses on the segmentation of the core only, no penumbra threshold are shown, but can be found in the respective references. Abels et al. suggests for the core segmentation a threshold of $CBV < 1.1\,ml/100g$ or a relative threshold of $rCBV < 0.37$. For comparison, Campbell et al. [Cam11] suggest in a more recent publication a threshold of $CBV < 2.0\,ml/100g$ but rather recommends to use $CBF < 15\,ml/100\,g/min$ instead. In another recent publication Lin et al. [Lin16a] suggests a relative threshold of $rCBV < 0.55$.

Chapter 3

Technical Principles

The aim of this chapter is to introduce the concept of neural networks (Section 3.1.1) including their optimization (Section 3.1.2) with its necessary constituents to enable image processing (Section 3.1.3). Furthermore this chapter presents neural networks with suitable architectures for image segmentation purposes (Section 3.2.1) as well as metrics and losses (Section 3.2.2). The principle chapter ends with section 3.3 about a detailed introduction to attention analysis of neural networks as a tool for further investigations.

3.1 Deep Learning

3.1.1 General Concept

A classification problem is defined as the process of mapping multi-dimensional features $\boldsymbol{x}$, e.g. vectors $\boldsymbol{x} \in \mathbb{R}^n$ to classes $y \in \mathbb{R}$ [Bis06]. The ideal classifier f^* is a function that maps the data to their corresponding labels $y = f^*(\boldsymbol{x})$. In a classification setting the goal is to find a classifier f returning predictions $\hat{y} = f(\boldsymbol{x})$, that approximates f^* given N correspondences of $\boldsymbol{x}$ and y, such that the error minimizes between the so called "label" y and the "prediction" $\hat{y}$, for some measure of error [Goo16]. There exist many methods to model f, e.g. Support Vector Machines [Cor95] or Random Forest Classifiers [Ho95], while deep neural networks especially, became quite impactful [Den14].

Multi-layered neural networks are the fundamental concept behind deep learning models [Goo16]. Artificial neural networks, inspired by the nature of biological neurons, consist of so called artificial neurons. Biological neurons receive their signal via dendrites, process the signal in the cell and output the processed signal via the axon [McC43]. This idea is transferred to

the artificial neuron, which receives some multidimensional input, e.g. a vector $\boldsymbol{x} \in \mathbb{R}^n$, where n denotes the number of features, and returns one single value $\hat{y} \in \mathbb{R}$. The signal processing is modeled by a weighted sum with $\boldsymbol{w} \in \mathbb{R}^n$ and a so called bias $b \in \mathbb{R}$ such that the neuron computes the following [Goo16]

$$\hat{y} = \sum_{i=1}^{n} w_i x_i + b, \tag{3.1}$$

where i represents the i-th component of the input vector $\boldsymbol{x}$ and $\boldsymbol{w}$. More elegantly with the scalar products treating the vectors $\boldsymbol{w}$ and $\boldsymbol{x}$ as one dimensional matrices [Goo16]

$$\hat{y} = \boldsymbol{w}^T \boldsymbol{x} + b \tag{3.2}$$

The bias can be considered as a weight acting independent of the input. This artificial neural cell is also known as a perceptron [Ros58]. A perceptron, as described in equation 3.1 or 3.2, yields only one output value. In order to provide multiple outputs, say a vector, m perceptrons are conducted in parallel, all receiving the same input $\boldsymbol{x}$, combined yielding the outputs $\hat{\boldsymbol{y}} \in \mathbb{R}^m$. This multi output perceptron is also called a fully-connected layer because when interpreted as a graph, this layer links every input to every output with a weight [Bis06]. Notice that n is given by the number of inputs, while m, the number of neurons is a parameter than can be set freely. Linear algebra provides with matrices an efficient way of expressing the fully connected layer. The fully connected layer is a matrix multiplication where $\boldsymbol{W} \in \mathbb{R}^{m \times n}$ denotes the neurons weights, with each row representing one neuron. Additionally, extending the dimensionality of $\boldsymbol{x}$ by one entry filled with a "1", the bias can be incorporated into the weights matrix [Goo16]. A fully connected layer expressed as a formula is

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{W}}(\boldsymbol{x}) = \boldsymbol{W}\boldsymbol{x} \tag{3.3}$$

Considering the biological cell again, the neuron processes the signal not only linearly, but also non-linearly [Goo16]. As an extension to the fully connected layer, these non-linear operations, also known as activation functions, play a major role in the field of deep learning. The activation function is a non-linear function $f_{activation} : \mathbb{R}^n \to \mathbb{R}^n$. A frequently and commonly used activation function is the rectified linear unit (ReLU) [Jar09]

$$\hat{y} = f_{ReLU}(\boldsymbol{x}) = \begin{cases} 0 & \boldsymbol{x} \leq 0 \\ \boldsymbol{x} & \boldsymbol{x} > 0 \end{cases} \tag{3.4}$$

or the Sigmoid unit as a second example for a widely used activation function [Goo16]

$$\hat{y} = f_\sigma(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{x}}} \tag{3.5}$$

both shown in figure 3.1.

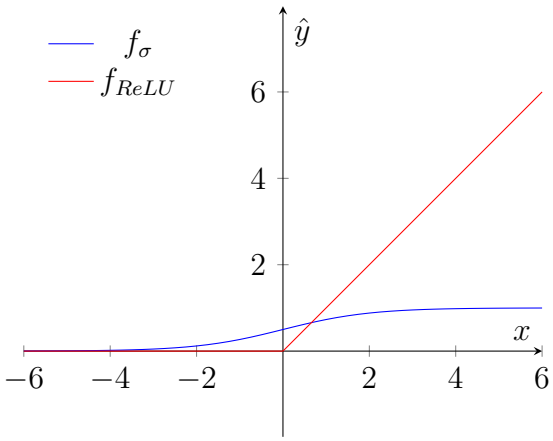

Figure 3.1: Activation functions ReLU (red) and Sigmoid (blue) are plotted.

The core idea of deep learning is the chaining of multiple layers, thus the phrase “deep“. Each of these layers, of a so called network, can generally be denoted by some function $f^{(l)} : \mathbb{R}^n \to \mathbb{R}^m$ where l represents the position of the respective layer inside the network and can either be a “weighted“ layer or an activation layer [Goo16]. $f^{(l)}$ receives $\boldsymbol{x}^{(l)} \in \mathbb{R}^n$ and returns $\hat{\boldsymbol{y}}^{(l)} \in \mathbb{R}^m$, such that

$$f^{(l)}(\boldsymbol{x}^{(l)}) = \hat{\boldsymbol{y}}^{(l)} \tag{3.6}$$

Finally, chaining d layers consecutively together defines a feed forward network [Goo16]

$$\hat{\boldsymbol{y}} = f(\boldsymbol{x}) = \hat{\boldsymbol{y}}^{(d)} = f^{(d)}(f^{(d-1)}(\ldots(f^{(2)}(f^{(1)}(\boldsymbol{x}))))) \tag{3.7}$$

The respective outputs and inputs for each layer relate as follows

$$\hat{\boldsymbol{y}}^{(l-1)} = \boldsymbol{x}^{(l)} \tag{3.8}$$

Notice that, d is again a parameter, that can be chosen freely. Typically, a deep learning model uses fully connected layers and activation layers alternatingly. For instance a network of three

layers and Sigmoid activations might defined as the following

$$\hat{\boldsymbol{y}} = f(\boldsymbol{x}) = \hat{\boldsymbol{y}}^{(d)} = f^{(3)}_{\boldsymbol{W}}(f^{(2)}_{\sigma}(f^{(1)}_{\boldsymbol{W}}(x))) = f^{(3)}(f^{(2)}(f^{(1)}(\boldsymbol{x}))) \tag{3.9}$$

The equations 3.7 and 3.9 are representing the "forward pass", which describes the forward propagation of the input $\boldsymbol{x}$ through the network f to the output $\hat{\boldsymbol{y}}$. The use of non-linear activations is crucial since otherwise, a d-layered network, without any non-linearities would mathematically collapse into one single layered network

$$\hat{\boldsymbol{y}} = f^{(d)}_{\boldsymbol{W}}(f^{(d-1)}_{\boldsymbol{W}}(\dots(f^{(2)}_{\boldsymbol{W}}(f^{(1)}_{\boldsymbol{W}}(\boldsymbol{x}))))) = \boldsymbol{W}^{(d)}\boldsymbol{W}^{(d-1)}\dots\boldsymbol{W}^{(2)}\boldsymbol{W}^{(1)}\boldsymbol{x} = \boldsymbol{W}'\boldsymbol{x} = f_{\boldsymbol{W}'}(\boldsymbol{x}) \tag{3.10}$$

As a result, non-linear activations prohibit the simplification into one layer and that gives the network the ability of solving complex problems (approximating more complex f^*).

3.1.2 Optimization

As presented in section 3.1.1 a network consists of weights. In order to solve the task at hand, these weights must be fine tuned accordingly, using training samples. The deviation between known labels $\boldsymbol{y}$ and the predictions $\hat{\boldsymbol{y}}$ is measured by the loss function $L(\boldsymbol{y}, \hat{\boldsymbol{y}})$ receiving N data correspondences of labels and predictions. For instance the L_2-loss based on the L_2-Norm is defined as

$$L_2(\boldsymbol{y}, f(\boldsymbol{x})) = \sum_{j=1}^{m}(\boldsymbol{y}_j - f(\boldsymbol{x}_j))^2 = \|\boldsymbol{y} - \hat{\boldsymbol{y}}\|_2^2 \tag{3.11}$$

with j enumerating with the correspondence. The loss function allows the reformulation of the given problem into an optimization problem, where the goal is to find the neuron weights $\boldsymbol{W}$ to approximate f^* as best as possible

$$\boldsymbol{W}^* = \underset{\boldsymbol{W}}{\operatorname{argmin}}\, L(\boldsymbol{y}, f(\boldsymbol{x})) \tag{3.12}$$

e.g. for the network in equation 3.9 the optimization problem can be formulated as:

$$\boldsymbol{W}^{(1)*}, \boldsymbol{W}^{(2)*} = \underset{\boldsymbol{W}^{(1)}, \boldsymbol{W}^{(2)*}}{\operatorname{argmin}}\; L(y, f^{(2)}_{\boldsymbol{W}}(f_{\sigma}(f^{(1)}_{\boldsymbol{W}}(\boldsymbol{x})))) \tag{3.13}$$

By definition of the losses, the weights have direct impact on the performance in terms of the error made as it illustrated in figure 3.2. For reasons of simplicity both $\boldsymbol{W}$ are simplified to be one scalar value and the dependencies to $\boldsymbol{x}$ and $\boldsymbol{y}$ were omitted. Due to the non-linearities of a neural

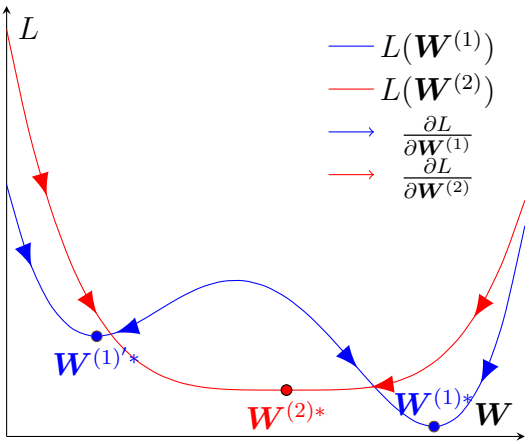

Figure 3.2: Simplified loss-weight curves, where $L(\boldsymbol{W}^{(1)})$ is non-convex, thus gradient descent would lead into two different solutions $\boldsymbol{W}^{(1)'*}$ and $\boldsymbol{W}^{(1)*}$. $\boldsymbol{W}^{(1)*}$ is a better choice as $\boldsymbol{W}^{(1)'*}$ but its not guaranteed to be the global minimum, since not the whole domain of $L(\boldsymbol{W}^{(1)})$ is plotted. Although $\boldsymbol{W}^{(2)}$ seems to be convex, it is not guaranteed as well, since only a sub-region is shown, with one optimum at $\boldsymbol{W}^{(2)*}$. The arrows represent the direction of the gradient, and thus the path the gradient descent algorithm would follow.

network most loss functions become non-convex. Because of the large number of adjustable weights the problem becomes high dimensional. Which means the networks are usually trained by iterative algorithms, ending up (due to the non-convexity) with a solution which cannot guaranteed to be the global optimum [Goo16, Bis06]. As visualized in figure 3.2, the goal is to approach $\boldsymbol{W}^*$, which is possible following the graphs by their steepest descent pointing into the closest local minimum. This algorithm is called gradient descent and is widely used in the deep learning community. The gradient iteratively leads to a solution of the minimization problem which is computed using the following update scheme for the layer $f^{(l)}$ with weights $\boldsymbol{W}^l$ [LeC98]

$$\boldsymbol{W}^{(l)} \leftarrow \boldsymbol{W}^{(l)} - \eta \frac{\partial L}{\partial \boldsymbol{W}^{(l)}} \tag{3.14}$$

where $\eta \in \mathbb{R}$ is the step size which is often referred as learning rate in the context of neural networks. The global learning rate is usually set to constant value. For each layer with weights, the gradient $\frac{\partial L}{\partial \boldsymbol{W}^{(l)}}$ must be determined. One established method to compute the necessary gradient $\frac{\partial L}{\partial \boldsymbol{W}^{(l)}}$ is the backpropagation algorithm published in [Rum88], which recursively uses the chain rule in order to derive layer by layer with respect to their weights. Mandatory component is the loss function L whose derivative $\frac{\partial L}{\partial \hat{\boldsymbol{y}}}$ with respect to the prediction $\hat{\boldsymbol{y}}$ is the starting point of the backpropagation algorithm.

$$\frac{\partial L}{\partial \hat{\boldsymbol{y}}} = \frac{\partial L}{\partial \hat{\boldsymbol{y}}^{(d)}} \tag{3.15}$$

For example the L2 loss from equation 3.11 is derived as follows:

$$\frac{\partial L_2}{\partial \hat{\boldsymbol{y}}} = 2\left(\boldsymbol{y} - \hat{\boldsymbol{y}}\right) \tag{3.16}$$

The algorithm iterates now from the most outer function $f^{(d)}$ to the most inner function $f^{(1)}$ of the network. With each iteration i a new shell of the chaining is being considered computing each of the following by using the chain rule for derivatives:

$$\frac{\partial L}{\partial \boldsymbol{W}^{(l)}} = \frac{\partial L}{\partial \hat{\boldsymbol{y}}^{(l)}} \frac{\partial \hat{\boldsymbol{y}}^{(l)}}{\partial \boldsymbol{W}^{(l)}} \tag{3.17}$$

$$\frac{\partial L}{\partial \boldsymbol{x}^{(l)}} = \frac{\partial L}{\partial \hat{\boldsymbol{y}}^{(l)}} \frac{\partial \hat{\boldsymbol{y}}^{(l)}}{\partial \boldsymbol{x}^{(l)}} \tag{3.18}$$

Since the output of one inner layer f_{l-1} is the input of the next outer layer f_l

$$\hat{\boldsymbol{y}}^{(l-1)} = \boldsymbol{x}^{(l)} \tag{3.19}$$

the gradient with respect to the input is being carried over to the next iteration by

$$\frac{\partial L}{\partial \hat{\boldsymbol{y}}^{(l-1)}} = \frac{\partial L}{\partial \boldsymbol{x}^{(l)}} \tag{3.20}$$

and the recursion continues by decrementing l until the earliest layer is reached [Rus16]. From equation 3.17 and 3.18, it follows, that only the partial derivatives $\frac{\partial \hat{\boldsymbol{y}}^{(l)}}{\partial \boldsymbol{W}^{(l)}}$ and $\frac{\partial \hat{\boldsymbol{y}}^{(l)}}{\partial \boldsymbol{x}^{(l)}}$ must be determined for each layer, all other components are given by the iterative nature of this algorithm. For example, the necessary partial derivatives for a fully connected layers are computed as follows

$$\hat{\boldsymbol{y}}^{(l)} = \boldsymbol{W}^{(l)}\boldsymbol{x}^{(l)} \tag{3.21}$$

$$\frac{\partial \hat{\boldsymbol{y}}^{(l)}}{\partial \boldsymbol{W}^{(l)}} = \boldsymbol{x}^{(l)} \tag{3.22}$$

$$\frac{\partial \hat{\boldsymbol{y}}^{(l)}}{\partial \boldsymbol{x}^{(l)}} = \boldsymbol{W}^{(l)} \tag{3.23}$$

For activation functions, no weights need to be updated, so computing equation 3.18 is sufficient to preserve the backpropagation.

As mentioned, it is common practice to use a gradient descent type algorithm for the optimization of neural networks. In general there exists three types which are described in the following, where N is the total number of training sets and the Batch Size BS is the number of samples

involved in the computation of one gradient step [Rud16]:

1. Gradient descent: The computation of the gradient is based on the entire data set, so the algorithm performs just one update. This type is computationally expensive because of high memory consumption but yields a stable update behavior.

$$BS = N \tag{3.24}$$

2. Stochastic gradient descent: The weights are updated with every sample of the data set. This method requires little memory but results in noisy update steps.

$$BS = 1 \tag{3.25}$$

3. Mini-batch gradient descent: The gradient is computed on mini-batches of the training data, so that with k updates all samples of the N data are covered. This method combines the best of the previous two methods. On one hand it is computationally manageable, while on the other hand the gradients are less noisy which leads into a more robust update.

$$BS = \frac{N}{k} \tag{3.26}$$

Due to the properties of each algorithm, the mini-batch variant is the one which is commonly used in deep learning practice. This thesis used one of most popular mini-batch gradient descent algorithms for updates, the Adaptive Moment Estimation algorithm (Adam) [Kin14]. Adam allows to assign individual learning rates to each weight parameter. Furthermore it provides momentum terms for the gradients and the individual learning rates, which allows Adam to incorporate the gradients and learning rates of the previous iterations for stabilizing the gradients and learning rates of the current iteration [Rud16].

3.1.3 Image Processing

Section 3.1.2 covered vectors as inputs only, but images can be processed by neural networks as well. An image can be treated as multidimensional data structure, with the resolution (number of pixels) in two dimensions (height h and width w) and the number of channels c (e.g. red, green and blue or greyscale) as a third dimension as shown in figure 3.3. In the following images are denotes as $\boldsymbol{X} \in \mathbb{R}^{h \times w \times c}$. In addition to the resolution of images, medical tomographic images also have a spatial resolution also known as pixel resolution, pixel spacing or the nominal voxel

distance of every voxel in the voxel grid. In other words, one pixel in image space represents a certain volume in spatial space. Characteristic parameters to describe the spatial resolution are the pixel spacing and the slice distance and/or the slice thickness.

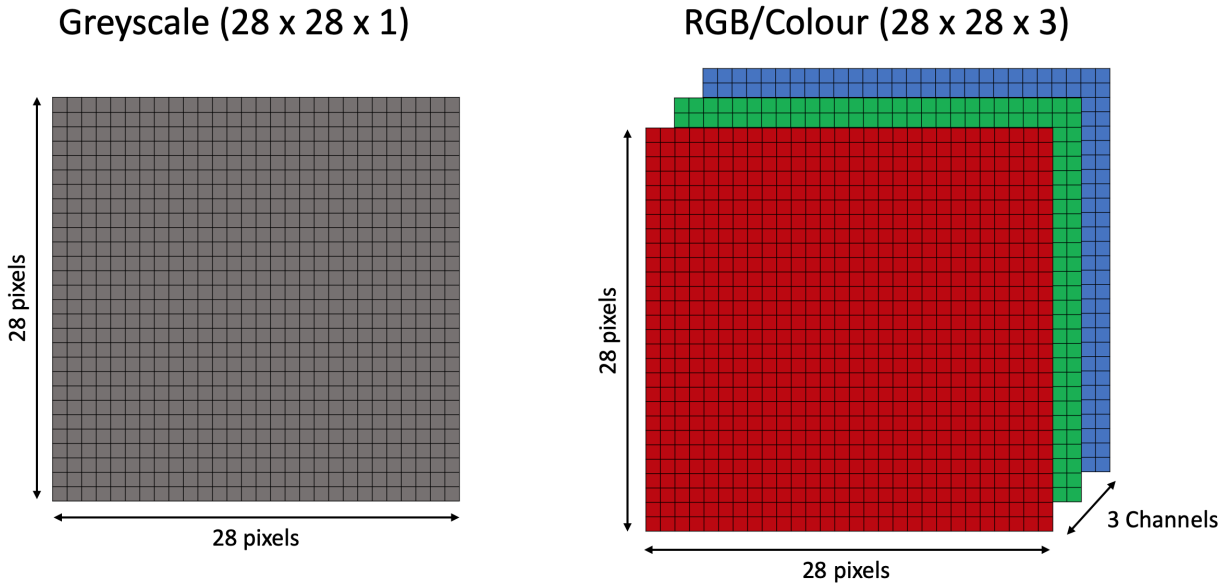

Figure 3.3: Left: A greyscale image (one channel) with a resolution of 28 x 28 pixels. Right: A colored image decomposed into its red, green and blue (RGB) channel again with a resolution of 28 x 28 pixels [Mar19].

Convolutional Layer

Convolutional layers allow the processing of images inside neural networks and became substantial in image processing. One convolution layer is represented by a set $\boldsymbol{K} = \{\boldsymbol{K}_i \in \mathbb{R}^{n\times m\times c} | i = 1, \ldots, k\}$ of k kernels with typically small spatial sizes in height n and width m, and a channel depth c which is equal to the channel depth of the input $\boldsymbol{X}$ [O'S15]. Just as the fully-connected layer, the convolution layer adds one bias for each of the k kernels with $\boldsymbol{b} \in \mathbb{R}^k$ to the convolution result. The convolution layer convolves the input $\boldsymbol{X} \in \mathbb{R}^{h\times w\times c}$ with its kernels where each convolution with one kernel results in a one-channeled feature map. To get k feature maps, this process is being repeated k times each with an individual kernel

$$\hat{\boldsymbol{Y}}_i = \boldsymbol{K}_i * \boldsymbol{X} + b_i, \quad i = 1, \ldots, k \tag{3.27}$$

with $\hat{\boldsymbol{Y}}_i \in \mathbb{R}^{h\times w\times 1}$ and $*$ denoting the convolution operation. After all feature maps have been merged, the entirety of the output is $\hat{\boldsymbol{Y}} \in \mathbb{R}^{h\times w\times k}$. With the use of Toeplitz matrices, the

convolution with kernels as in equation 3.27 can be transformed into a matrix multiplication [Gra06]. Thus, the convolution layer can be considered as a special type of fully connected layer, where the difference relies on the reuse of the kernel weights for multiple inputs instead of having individual weights for each input node. This concept is called “shared weights“ [LeC95]. As a result, compared to fully connected layer, the convolutional layer gets more training samples per weight, which has beneficial impact on the total performance.

Image 3.4 illustrates the convolution process for an image $\boldsymbol{X} \in \mathbb{R}^{32\times32\times3}$ with one kernel of the kernel set $\boldsymbol{K} = \{\boldsymbol{K}_i \in \mathbb{R}^{5\times5\times3} | i = 1, \ldots, 10\}$ yielding the output $\hat{\boldsymbol{Y}} \in \mathbb{R}^{32\times32\times10}$. In

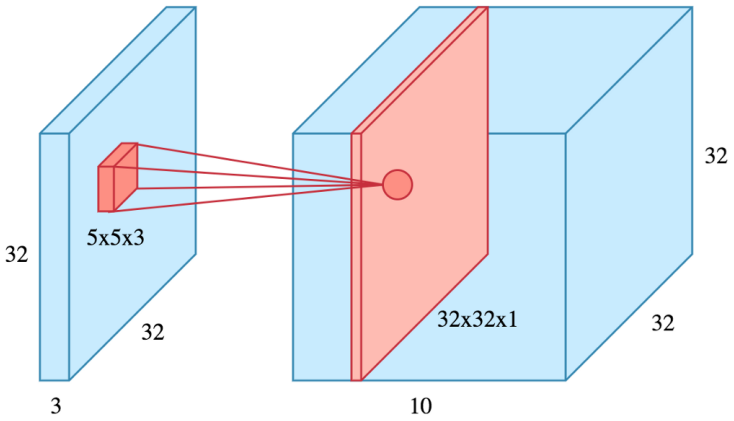

Figure 3.4: The convolution process of an image with a resolution of 32 x 32 pixels and 3 channels with 10 kernels each with 5 x 5 weights for all 3 channels, resulting in an image with a resolution of 32 x 32 and 10 channels [Ard19]

common deep learning frameworks (e.g. tensorflow [Aba15], theano [The16] and pytorch [Pas17]), the convolution layer can be configured, besides the mentioned kernel size (n and m) and number of filters (k) by certain parameters, summarized by Dumoulin and Visin [Dum16] as follows:

1. *Padding*, which defines the convolution on border cases. In this thesis “same“ mode was used, which extends the borders of the original image $\boldsymbol{X}$ with zeros, depending on the kernel size, so that the result $\hat{\boldsymbol{Y}}$ has the same resolution in height h and width w as the original image $\boldsymbol{X}$. This convolution mode is also shown in figure 3.4.

2. *Stride* defines the step size of one convolution slide. Image 3.4 shows the convolution with stride of 1 in height and width dimension. Image 3.5 illustrates the strided convolution of a 6×6 input with a 3×3 kernel using a 2×2 stride, which results in a reduction of dimensionality. Thus, the stride can be used in order to subsample images.

3. *Transposition* of a convolution, which swaps the forward and the backward pass of a

convolution, and can be considered as the opposite direction of an convolution. For instance a strided transposed convolution increases the resolution instead of subsampling. Whereas the convolution returns one single value per kernel slide, the transposed convolution returns per pixel new values for each kernel entry per slide as it is also shown in figure 3.5.

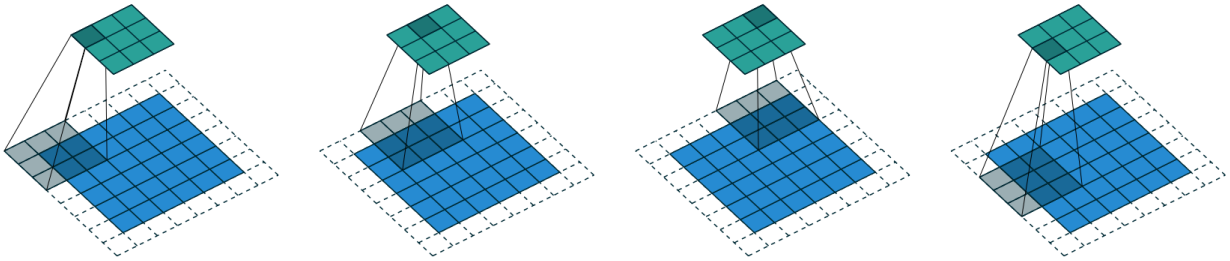

Figure 3.5: Strided Convolution: Blue marks the 6×6 input, which is convolved with a 3×3 kernel and a stride of 2×2 which results in the cyan marked output of a resolution of 3×3. The borders are padded by a 1×1 strip. In case of a stride of 1×1 the cyan output would have the same resolution as the input. Transposed Convolution: Cyan marks the 3×3 input, which is transposed convolved with a 3×3 kernel with no padding resulting in a 6×6 output marked as blue with an extended border of 1×1 which gets ignored in further proceedings [Dum16].

Max Pooling

Pooling layers provide an efficient way of subsampling images inside a neural network. Similar to convolutional layers, a kernel performing a non linear operation slides over the input. In most deep learning applications, the maximum operation is used, which returns the maximum value of the input patch. Like convolutional layers, pooling layers have the same parameters, but mostly they are used with strides higher or equal to 2, which allows the sub sampling [Dum16].

Batch Normalization

The batch normalization abbreviated with “BatchNorm“, published by Ioffe and Szegedy [Iof15] is a layer used for regularization purposes. The BatchNorm layer normalizes the feature maps along the batch axis regarding their mean and standard deviation. Originally, it was invented to counteract the so called “internal co-variate shift“ that says if a network utilizes ReLUs as activations, the activation values steadily drift away from the origin. Thus the network must constantly adapt to that dynamics such that a loss in prediction accuracy is observable. The BatchNorm layer compensates that by reshifting the activations back to the origin by normalization with their mean. However, San et al. [San18] rejected that theory, and showed experimentally, the batch

normalization layer smooths the loss-weights-landscape, which improves the stability of gradient descent and the networks sensitivity to hyperparameters and initialization.

3.2 Image Segmentation with Deep Learning

Image segmentation is defined as the process of sectioning an image into a set of regions containing pixels which relate to the same class. Because of the high degree of difficulty, algorithms of many kinds has been developed so far. Recent algorithms are e.g. thresholding approaches, region growing approaches, clustering methods, atlas guided approaches and neural networks [Pha00]. These algorithms vary as well as the data they can be applied on. Consequently, each algorithm has its own strengths and weaknesses, but convolutional neural networks (CNN) in combination with deep learning turned out to be very effective in addressing segmentation problems. Thus, deep learning gained more attention in the last few years while outperforming many of the other mentioned approaches.

3.2.1 Architectures

From a technical point of view, image segmentation is the same as the classification of every individual pixel. Once all pixels are classified, the sum of all classifications can be considered as the segmentation outcome. What all modern deep learning segmentation architectures have in common, is that they yield a map for every class containing probability values of how likely each pixel belongs to a certain class, whereas they differ in the way how to compute this map. This setup is often referred as end-to-end segmentation and was first proposed by Long et al. [Lon15]. The key component to enable end-to-end segmentations is the use of upsampling layers, which increase the resolution of feature maps after they have been downsampled by convolutional layers. The way of applying these upsampling layers is crucial. Long et al. further figured out in their work, the more often the input image gets convolved, hence compressed and consequently more often upsampled the better the segmentation gets, up to a certain degree. Ronnerberger et al. [Ron15] built upon that idea and modified the Long et al's architecture accordingly. Moreover, the network proposed by Ronneberger et al, also known as U-Net, consists of two main steps, the encoding process and the decoding process, similiar to Long et al's structure. In addition to that, after every upsampling, the upsampled result is concatenated with the corresponding feature maps of the encoding path. Due to this concatenation, information can be transferred and integrated into the decoding process. U-Net became very established in the field of image segmentation [Fal19],

therefore it was used in this thesis as well. A detailed description of U-Net can be found in section 4.2.1.

3.2.2 Metrics and Losses

The aim of this section is to present common and often used metrics in the field of segmentation and suitable loss functions designed for deep learning purposes. Image segmentation is the sum of the classification of individual pixels. Thus, the same measures for classification can be applied on the segmentation outcome, but on a pixel basis. This thesis focuses on the segmentation of the core only, therefore a binary segmentation is required which allows the use of the confusion matrix, whose components are explained in the following. The measurements for correctness is the *true positive* value, TP which is the number of correctly classified positive pixels (e.g. the class 'core'), and its negative counterpart is the *true negative* value TN, which is the number of correctly classified negative pixels (e.g. the class 'non-core'). The values describing the erroneous of a segmentation are the *false positive* value FP, which are the pixels falsely classified as positive (e.g. pixels that are of class 'non-core' but were predicted as 'core') and the negative counterpart the *false negative* value, which are the pixels that were falsely classified as negative (e.g. pixels that are of class 'core' but were segmented as 'non-core') [Faw06]. Figure 3.6 shows the confusion matrix, an illustration of how these values emerge in the context of the core segmentation.

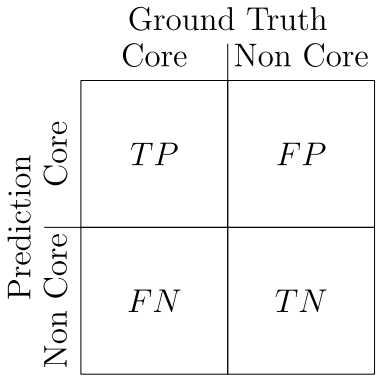

Figure 3.6: Confusion matrix in the context of the core segmentation.

Precision and Recall

Precision, defined in equation 3.28 is calculated as the proportion between the number of pixels predicted correctly and the total number of predicted labels and is often used in conjunction with recall defined in equation 3.29, which represents the percentage of correctly predicted pixels

among all true labels [Her16].

$$Precision = \frac{TP}{TP + FP} \tag{3.28}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.29}$$

Jaccard index and Mean Intersection over Union

The Jaccard index, defined in equation 3.30, also known as Intersection over Union, is a similarity measure to compare to segmented regions by their fraction of the region of overlap devided by the region of intersection [Jac12]

$$Jaccard = \frac{TP}{TP + FP + FN} \tag{3.30}$$

A good segmentation is achieved if the Jaccard index is close to 1, which means ground truth and prediction are perfectly overlapping each other. In contrast to a good segmentation, a bad segmentation that means ground truth and prediction do not overlap at all and the Jaccard index amounts to 0. The Jaccard index can be computed for both binary classes (Core and Non-Core), the average Jaccard index of both classes, defined in equation 3.31, is called

$$MIoU = \frac{1}{2}\left(\frac{TP}{TP + FP + FN} + \frac{TN}{TN + FP + FN}\right) \tag{3.31}$$

The MIoU behaves similar to the AUC measure. The perfect segmentation, reaches a MIoU value of 1, whereas its inversion (swapping both classes) has a MIoU of 0. Likewise the AUC, the MIoU measures the worst possible segmentation with 0.5.

Besides the usage of the Jaccard index as a measure, it can be used as a loss as well, by extending it into a distance function, thus the Jaccard loss is roughly the same as

$$L_{Jacc} \approx 1 - Jaccard \tag{3.32}$$

Optimization based on the Jaccard loss leads to a training that maximizes the overlap between the prediction (Core) and its ground truth. Losses that are directly computed on the TP, TN, FP and FN values, are not feasible for optimization purposes. The reason relies on the thresholding operation that is required to compute the confusion matrix. Once a thresholding is involved in the optimization, all gradients that depend on it (basically all gradients for weights), would collapse during backpropagation and its multiplications of gradients, because the derivative of a

step function (thresholding) is either 0 or undefined at the discontinuity. Thus, the Jaccard index must be approximated by a continuous measure behaving similarly. This approximation is denoted with d for "differentiable". The loss is being computed between a prediction $\hat{\boldsymbol{y}}$ and their ground truth correspondence $\boldsymbol{y}$

$$L_{Jacc} = 1 - Jaccard_d = 1 - \frac{|\boldsymbol{y} \cdot \hat{\boldsymbol{y}}|}{|\boldsymbol{y}| + |\hat{\boldsymbol{y}}| - |\boldsymbol{y} \cdot \hat{\boldsymbol{y}}|} \tag{3.33}$$

Sorensen-Dice coefficient

The Sorensen-Dice, also known as the F1-Score, defined in equation 3.34, is the harmonic mean between precision and recall, and therefore evaluates both measures as equally important. The Dice behaves similar to the Jaccard index, but weights the true positive predictions higher. A perfect segmentation reaches a Dice score of 1, and 0 otherwise [Tah15].

$$Dice = \frac{2TP}{2TP + FP + FN} \tag{3.34}$$

As mentioned above, the Jaccard index and the Dice coefficient are closely related to each other, and can be transformed into the other with

$$Jaccard = \frac{Dice}{2 - Dice} \tag{3.35}$$

Volumetric Similarity

Unlike the Jaccard index or the Dice coefficient, the volumetric similarity (VS) is not based on the overlap of ground truth and the prediction. Instead it measures the absolute difference of one region to the other, which means no overlaps are being considered [Tah15]. The VS measure is computed as follows:

$$VS = 1 - \frac{|FN - FP|}{2TP + FP + FN} \tag{3.36}$$

Binary Cross-Entropy Loss

The Cross-Entropy for binary problems, defined in equation 3.37 measures the dissimilarity between two distributions. In the application of deep learning, the predictions $\hat{y}$ and their ground truth correspondence y, where both the labels and predictions are scalars ranging from 0 to 1 [Goo16]. Let

$$L_{CE} = -\left[y \log\left(\hat{y} + \epsilon\right) + (1 - y) \log(1 - \hat{y} + \epsilon)\right] \tag{3.37}$$

be the (Binary-)Cross-Entropy loss where ϵ is a small smoothing factor to avoid $\log(0)$. The loss used in this thesis is the sum of the binary Cross-Entropy loss, which is motivated by information theoretic considerations and the Jaccard loss, which emerged by measuring the overlap and thus is more practically motivated.

$$L = L_{CE} + L_{Jacc} \tag{3.38}$$

3.3 Attention Analysis for Neural Networks

Deep Learning models are often classified as "black box" models, which is why deep learning models are difficult to establish in highly regulated fields like medical image processing. However, in the past years, several approaches have been published like [Mai19, Bac15, Shr17], addressing this problem, to get meaningful understanding what is happening inside a deep neural network. In this thesis, two approaches have been used, the Layer-wise Relevance Propagation algorithm in subsection 3.3.1 proposed by [Bac15] and the DeepLIFT approach proposed by [Shr17] in subsection 3.3.2. Sebastian Lapuschkin, one of the reasearches who invented the LRP algorithm, whose dissertation [Lap19] summarized and presented many different approaches, contributed to the major part of this section and is highly recommended for further readings.

3.3.1 Layer-wise Relevance Propagation

The goal of the Layer-wise Relevance Propagation (LRP) is to measure some quantity R called the relevance, which can be interpreted as an importance of a neuron, and its contribution to the output of a model f. To be more specific, the LRP method, decomposes a non-linear classifier, say a neural network, by the relevance of its components (weights). Let $f : \mathbb{R}^n \to \mathbb{R}$ be a trainable classifier mapping inputs $\boldsymbol{x} \in \mathbb{R}^n$ to some classes $\hat{y} \in \mathbb{R}$. One possible approach of decomposing the prediction $f(\boldsymbol{x})$ as a sum of the separate input dimensions is

$$f(\boldsymbol{x}) \approx \sum_{s=1}^{n} R_s \tag{3.39}$$

where R_s is called the relevance of the s-th input vector component. The qualitative interpretation of the relevance score is $R_s < 0$ contributes against the presence of a structure, whereas a score of $R_s > 0$ represents a contribution for its presence [Lap19]. The underlying main assumption for all computations in this algorithm is, that the sum of relevances in one layer is equal to the sum of relevances in all other layers. Let $R_i^{(l)}$ be the i-th relevant score of the l-th layer for each

component $z = z_i^{(l)} \in \mathbb{R}^{\dim(l)}$ and the $(l+1)$-th layer which is one step closer to the output with $R_j^{(l+1)}$ and its outputs $z = z_j^{(l+1)} \in \mathbb{R}^{\dim(l+1)}$. The relevance preservation assumption is then:

$$\sum_{n=1} R_n^{(input)} = \cdots = \sum_{i=1} R_i^{(l)} = \sum_{j=1} R_j^{(l+1)} = \cdots = f(\boldsymbol{x}) \tag{3.40}$$

The LRP algorithm assumes further, that f computes mappings z_{ij} between the intermediate result of the l-th layer $z_i^{(l)}$ to the next subsequent layer $l+1$ with outputs $z_j^{(l+1)}$, such that

$$z_j = \sum_i z_{ij}, \text{ s.t.: } i \text{ is the mapping to } j \tag{3.41}$$

Thus each mapping z_{ij} has its contribution to z_j at output j. For instance the mapping in a fully connected layer might be $z_{ij} = w_{ij}x_i$. In order to get the input relevance score, intermediate relevances $R_i^{(l)}$ must be computed first. To compute this, the relevance $R_i^{(l)}$ must be expressed in terms of relevance messages $R_{i\leftarrow j}^{(l,l+1)}$, which corresponds to the appropriate forward mapping z_{ij} and depends on the given relevance values of the $R_j^{(l+1)}$ of the succeeding layers. First the local relevance conservation constraint must be defined which describes the backpropagation of relevances from j at layer $(l+1)$ to its inputs i at layer (l):

$$R_i^{(l)} = \sum_i R_{i\leftarrow j}^{(l,l+1)} = R_j^{(l+1)} \tag{3.42}$$

Messages that satisfy equation 3.42 can be of the following form

$$R_{i\leftarrow j}^{(l,l+1)} = \frac{z_{ij}}{z_j} R_j^{(l+1)} \tag{3.43}$$

where $\frac{z_{ij}}{z_j}$ is the proportional contribution from the activation z_j to the j-th output component [Lap19]. This rule is called z-rule and treats positive activations and negative activations equally. This process is being considered as decomposition into individual relevances and continues in a recursive way, until the input layer is reached. In summary the Layer-wise relevance propagation algorithm is the following:

As an extension to the algorithm 1, activation higher or lower to zero can be processed

Algorithm 1 The LRP Algorithm [Lap19]

procedure LRP($R^{(d)} = f(\boldsymbol{x})$, mappings z_{ij} and z_j for all layers)
  for $l \in \{d-1, \ldots, 1\}$ **do**
    $\forall i, j : R_{i \leftarrow j}^{(l,l+1)} = \frac{z_{ij}}{z_j} R_j^{(l+1)}$ ▷ z-rule
    $\forall i : R_i^{(l)} = \sum_i R_{i \leftarrow j}^{(l,l+1)}$
  end for
  return $\forall i, l : R_i^{(l)}$
end procedure

individually, which stabilizes relevance propagation. Let

$$z_j^+ = \sum_i z_{ij}^+ \tag{3.44}$$

$$z_j^- = \sum_i z_{ij}^- \tag{3.45}$$

where $+$ and $-$ denote the positive and the negative part of z_{ij} [Lap19]. An alternative to the relevance propagation formula in 3.43 is the $\alpha\beta$-rule defined as

$$\begin{aligned} R_{i \leftarrow j}^{(l,l+1)} = \left(\alpha \frac{z_{ij}^+}{z_j^+} + \beta \frac{z_{ij}^-}{z_j^-} \right) R_j^{(l+1)} \\ \text{s.t. } \alpha + \beta = 1 \\ \alpha \geq 1 \end{aligned} \tag{3.46}$$

The $\alpha\beta$-rule enables the manual impact adjustment of positive or negative contributions, by choosing different values for α and β. The restriction $\alpha + \beta = 1$ ensures the relevance conservation property embedded in equation 3.40. The underlying assumption of this rule is, that outputs of layers are the consequences of positive inputs, whose propagation is reinforced by $\alpha \geq 1$. It thus suits the paradigm of having neural networks containing ReLU activations especially well [Lap19]. Since in this thesis entirely ReLU activation were integrated into the networks, the $\alpha\beta$-rule has been used additionally to the z-rule in equation 3.43. An implementation for the LRP-Algorithm was provided by the python package *innvestigate* [Alb18].

3.3.2 DeepLIFT

An alternative to the LRP approach is the Deep Learning Important Features approach (DeepLIFT). DeepLIFT computes its relevances by comparison of each neuron to its “reference activation“ and

assigns them according to the difference [Shr17]. In the thesis at hand, this reference image is the background of all images. Similar to its $\alpha\beta$-rule, DeepLIFT considers positive and negative contributions separately [Lap19]. In contrast to LRP, the advantage of DeepLIFT is, that its based on the gradients computation which can be done efficiently by backpropagation on graphics hardware. A DeepLIFT implementation was provided the python package *innvestigation* [Alb18].

Chapter 4

Materials and Methods

4.1 Data

As most deep learning approaches, the proposed deep learning methods are highly data driven. This chapter describes the data used in this thesis, including an overview in section 4.1.1 describing properties of each data set source, a detailed analysis about the quality of the various data sources in section 4.1.2, and the pre-processing pipe line which is necessary to normalize the data among different data sets which is described in section 4.1.3.

4.1.1 Data Overview

In the context of this work, three sources for data have been acquired for CTP volumes with their corresponding follow up DWI data. Two of these sources, abbreviated with A (37 volumes) and B (15 volume), were provided by SIEMENS Healthcare GmbH. The third source was provided by the ISLES Challenge 2018 at MICCAI [Mai17] with 94 volumes in total. These three sources had different setups, thus they have different image properties which are listed in table 4.1. Table 4.1 is sorted by Scanner models. Furthermore, most image properties rather relate to the scanner and their reconstruction parameters rather than to clinical workflows. The scanners used to acquire the ISLES data set are unknown. All Siemens scanners acquired the perfusion data in the spiral mode. The Canon Aquilion One, with a detector of 320 rows, was large enough to cover the whole head region and did neither require dual slabs nor a spiral acquisitions. The ISLE data was scanned entirely in single slab or dual slab mode. For the latter, the individual slabs were stored separately, thus the large number of volumes, such that studies of 63 patients in total were provided by the ISLE Challenge. Considering A, B and ISLES data, volumes of 115 patients in total were available.

Table 4.1: Overview of all data sets sorted by scanner model

Scanner Model	Nr. of Volumes	Average Nr. of Slices per Volume	Resolution in px	Average Spatial Resolution in mm/px
SIEMENS				
Definition Flash	16	26	512×512	$4.8 \times 0.4 \times 0.4$
Definition As+	20	19	512×512	$5 \times 0.4 \times 0.4$
Sensation 64	3	2	512×512	$14.4 \times 0.4 \times 0.4$
Force	1	37	512×512	$3 \times 0.4 \times 0.4$
CANON				
Aquilion One	12	320	512×512	$0.5 \times 0.4 \times 0.4$
Unknown				
(ISLES 2018)	94	5.3	256×256	$9.5 \times 0.9 \times 0.9$

For sources A and B, the corresponding MRI DWI data was annotated and approved by experienced radiologists, using the SIEMENS Syngo RT Workflow which includes a tool for annotations that generates polygonal chains after the contour has been drawn. Each polygonal chain consists of a set of coordinates sorted for each slice. Since the polygon is located in the MRI coordinate system, the MRI volume must be registered onto the CT volume, preferable the BASE channel. The registration is also performed using the Syngo RT Workflow. The resulting transformation is used to transform the polygon point-by-point into the CT coordinate system. Transforming the contour coordinates directly into the CT system instead of a segmentation mask reduces errors that emerge due to interpolations of bitmasks, hence this method was chosen. The ISLES ground truth data was provided already in a registered set up. Its corresponding MR data was not provided, so neither the annotation accuracy nor the registration quality could be validated.

4.1.2 Data Analysis

As shown in table 4.1, the complete data set consists of volumes in different image and spatial resolutions. Neural networks require a fixed image size (for batch-wise training), hence the data must be resampled onto a common resolution. In addition, convolutional neural networks are not scale invariant, thus data sets with identical spatial resolutions are preferred, since patients appear in the same scaling, despite from some variances. Besides the general resolution and its scaling, partial volume effects are a problem as well. For instance, slices acquired by the SIEMENS Sensation 64 System are reconstructed with slice distances (z-axis) of approximately $14\,mm$ to

each other, which on the one hand results in less noisier images but on the other hand anatomical structures are more difficult to differentiate. Its counterpart, images with thin sliced distances, as the CANON Aquilion One System generates, are highly overrepresented in the entirety of the data with each volume consisting of 320 slices. Thin slices are preferred in the representation of fine brain structures but appear with rather higher noise levels.

As [Lin16a] denoted, single slab or dual slab acquisitions are rather outdated and had been replaced in the past years by spiral or full detector acquisitions. A further analysis of the ISLES data has shown, that ground truth and the CT volumes are inaccurately registered to each other as image 4.1 visualize. Having inaccurate ground truth causes several problems. Assuming the network yields a perfect segmentation, the results would still be punished as imperfect and so is the training being directed into undesired solutions. Secondly, different registrations enlarge the inter-data set-variance, which must be implicitly trained by the network as well, causing a reduction of segmentation accuracy. An evaluation has shown, that some neural network models were still able to segment the regions properly without colliding with invalid regions (as air or skull), since the majority of the data sets is still well registered. The segmentation result of one of these networks is shown in figure D.1.

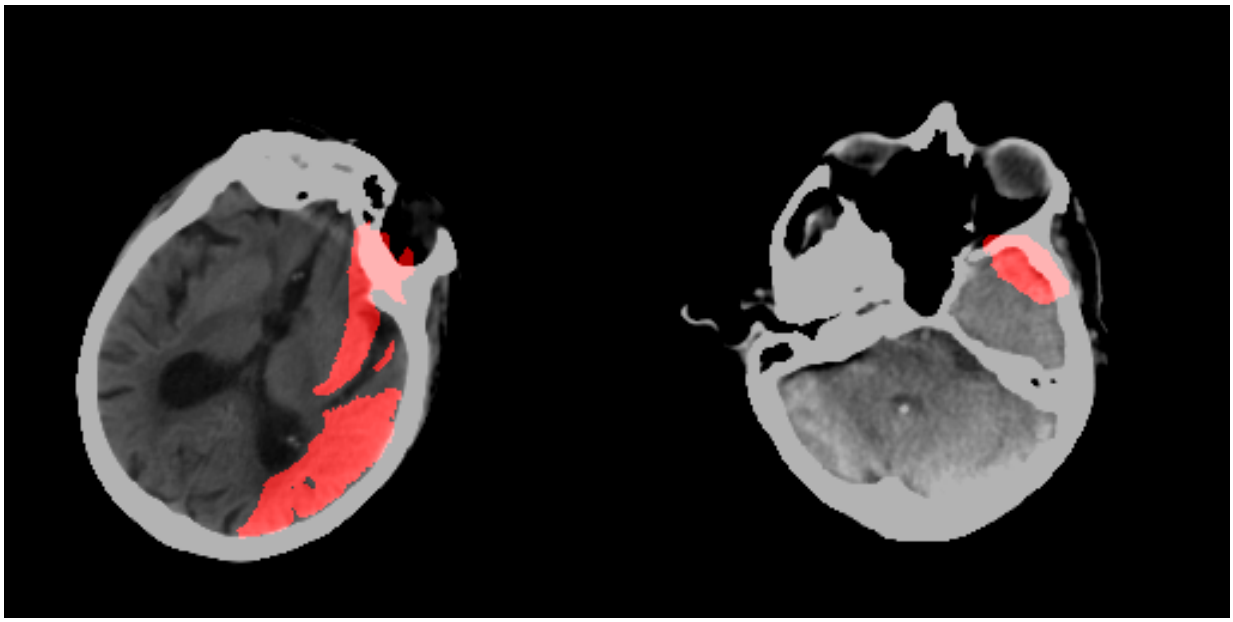

Figure 4.1: Two BASE slices of the ISLES 2018 data set, overlayed with the corresponding DWI ground truth annotation of the ischemic core lesion (red). In both examples, the annotation collides with the skull, which is an indicator for an inaccurate registration or annotation.

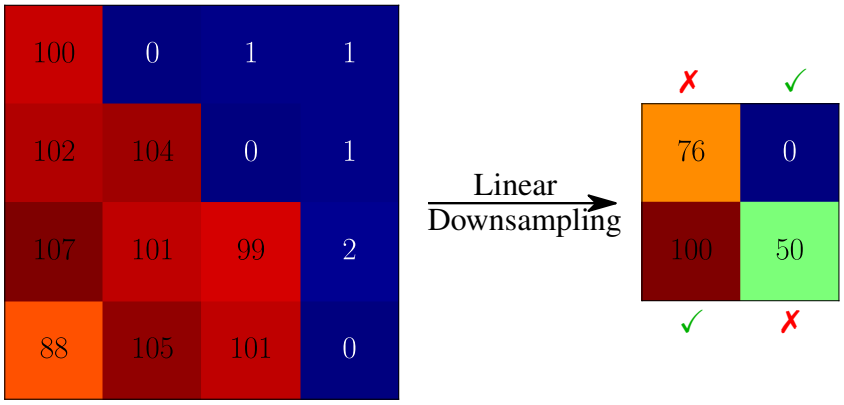

Figure 4.2: Subwindow of a CBF image (left), visualizing the interpolation problem on hard border on pixel basis, with their pixel values written in the center of each pixel. At hard border, e.g. from tissue (≈ 100) to air or background (≈ 0) all interpolations methods that incorporate neighboring pixels, cause artifacts in terms of invalid pixels, which are marked with red crosses. Non-border pixel can be interpolated, here marked with green checkmarkes.

4.1.3 Data Preparation

As presented in section 4.1.2, it is necessary to change the resolution of all data sets onto a common size and beneficial to resample the spatial resolution to isotropic voxel sizes so that anatomical structures are more likely to appear in the same scaling. The standard approach to achieve this is interpolation, but resampling on CT perfusion data causes artifacts. In cases of hard borders, which are very likely to appear on perfusion images, because voxels removed by segmentation are marked with dedicated values, are polluting valid pixels in the brain tissue, as illustrated in figure 4.2.

This effect gets amplified the more polluting pixel are involved in the interpolation process. Thus higher order interpolation algorithms (cubic, quadratic, etc.) and/or interpolations on 3D data, instead of 2D data as in image 4.2, exacerbate that effect. Except for nearest neighbor interpolation which causes no artifacts, but reduces the amount of information in cases of subsamplings, hence is not desirable as well. Image 4.3 illustrates the different interpolation methods and the resulting artifacts on a CBF image. As image 4.3 shows, some interpolation methods can introduce new clusters as possible candidates for infarcted regions. In this section, an image processing pipeline is being proposed, that allows linear interpolations on brain regions with a reduced amount of artifacts compared to the mentioned approaches.

The proposed method automatically combines the artifact-free nearest neighbor interpolation, which does not integrate a set of pixels in the subsampling process, and the linear interpolation

that is erroneous at border but integrates the content of neighboring pixel in the subsample process. The proposed pipeline is shown in figure 4.4. The processing pipeline starts with the CBF image of the volume which must be interpolated at (a) with its native resolution, in this example 512×512. Based on the CBF image, the processing splits into two branches. One branch performs the actual interpolation with both the nearest neighbor and linear interpolation. The purpose of the second branch is to compute a mask where "1" marks polluted pixel which must be replaced with the nearest neighbor interpolation and "0" pixels that are allowed to be replaced with pixels of the linear interpolation. The mask is computed based on the CBF channel, but is being used for all remaining channels. Alternatively to CBF, the other Non-HU maps could be used for the processing as well, as long as they have denoted background pixels. All images used for training purposes were first rescaled such that they have a pixel spacing of $1\,mm/px$ in both height and width (axial - x and y direction) followed by individual edge padding to reach a resolution of 256×256. First experiments included a rescaling of all data sets so they have $3\,mm/px$ in z-direction, but still artifacts were introduced in data sets by the Sensation 64 and the ISLES data similiar to these in image 4.3. As a consequence, the ISLES and Sensation data sets were not interpolated along the z-axis, except the CANON Acquilion One data, which was resampled with $3\,mm/px$ because these slices are without interpolation in z-direction highly oversampled with distances of $0.5\,mm$.

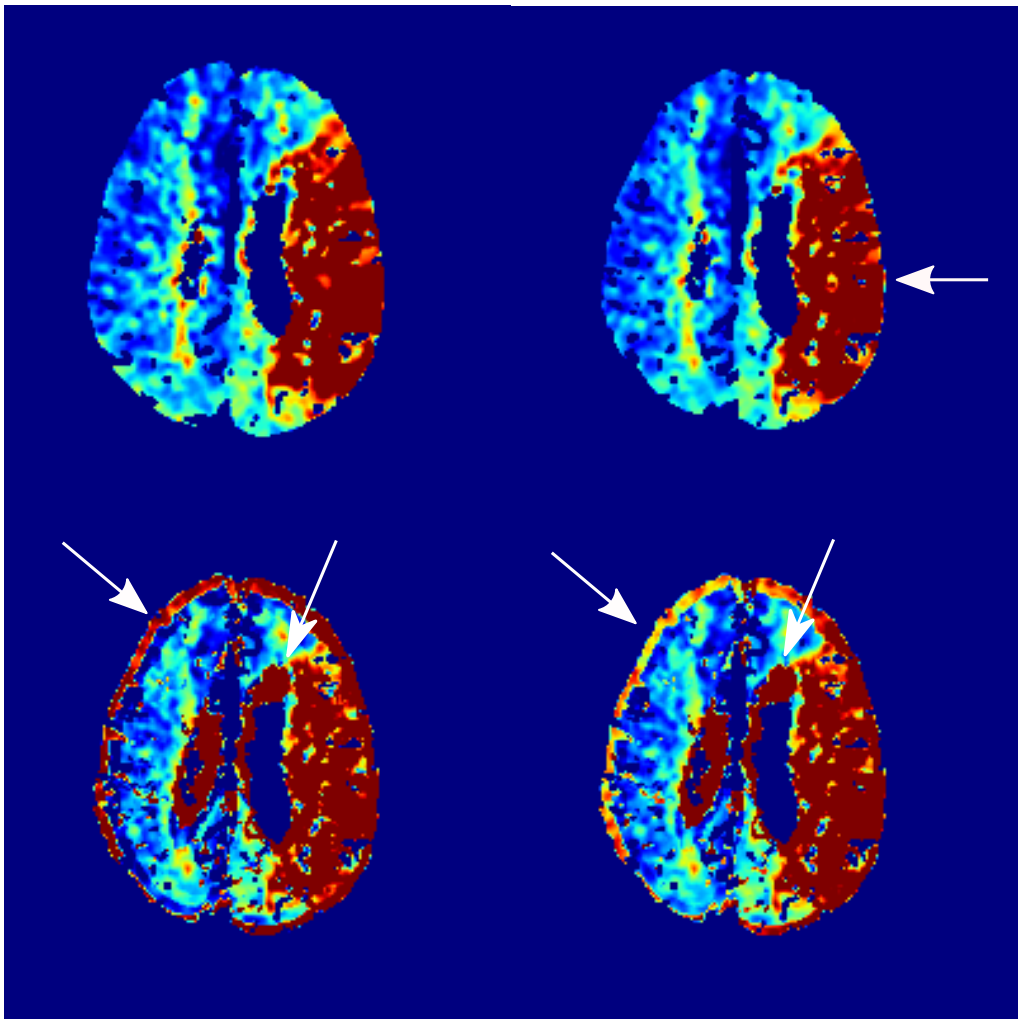

Figure 4.3: CBF image 3D interpolated with different algorithms from a native resolution of 512×512 to 256×256. (a) is interpolated with Nearest Neighbor. This method is considered to be the method that introduces no further artifacts but is not able to interpolate between slices (3D) and reduces information in downsampling. The arrows in (b), (c) and (d) point to typical artifacts where artifacts appear with each interpolation algorithm. (b) shows the bi-linear interpolation, that appear with typically thin edges as it is also illustrated in 4.2. (c) shows the bi-quadratic interpolation that exacerbates interpolation artifacts on borders compared to linear interpolation. (d) shows the bi-cubic interpolation and introduces the same kind of artifacts as (c). Bi-cubic and bi-quadratic introduce regions near the CSF that are possible candidates for infarcted regions.

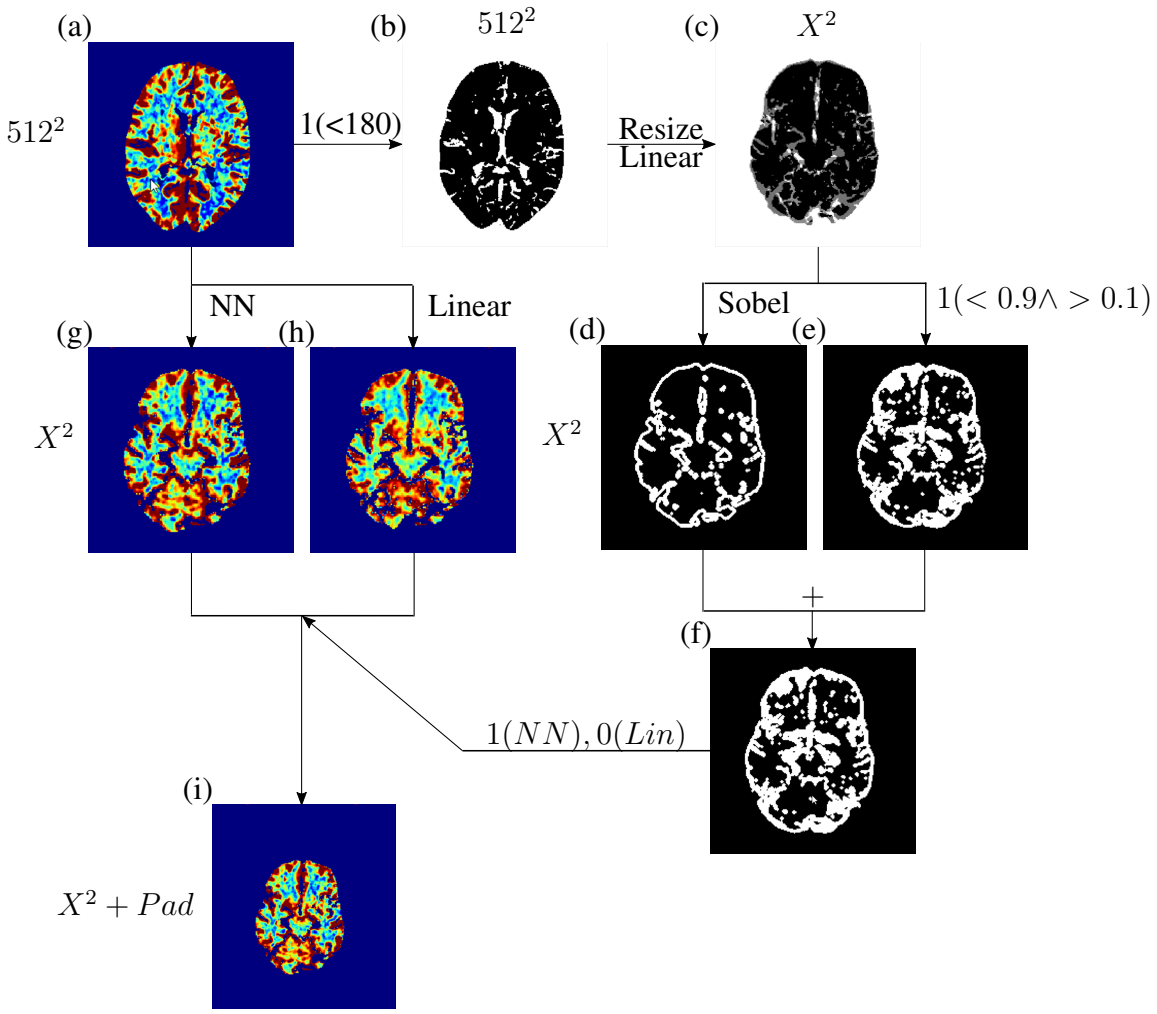

Figure 4.4: Illustration of the pre-processing pipeline, that reduces the amount of interpolation artifacts. Next to each image is the resolution, where X denotes the desired individual resolution for each volume that results in isotropic voxels. The pipeline starts with (a) the CBF channel of each volume. In order to segment background and foreground in (a), a thresholding segmentation with a threshold of 180 is to be done, where "1" represent background pixel and "0" foreground pixel, which results in (b). A linear interpolation of (b), contains either "0" or "1" at positions of valid interpolations and invalid polluted values between "0" and "1" visualized as grey regions in (c). These grey regions are again segmented in (e) which marks all pixel in the grey range, here lower than 0.9 and higher than 0.1 as "1". Experiments have shown that not all polluted regions can be located as computed in (e), thus the pollution mask is enhanced by the sobel edge detection image (d). Both, (d) and (e) are logically conjuncted resulting in (f), which represents the mask of pixels that must be replaced with the nearest neighbor interpolations for all "1" and linear interpolation for all "0". (a) is interpolated with both nearest neighbor and and linearly resulting in (g) and (h) respectively. Arranging the image with (g) and (h) according to (f) results in the validly interpolated image. In addition to that the image is padded onto the desired resolution as shown in (i).

4.2 Segmentation

4.2.1 U-Net

One established neural network architecture for image segmentation purposes is U-Net [Ron15], which is shown in figure 4.5, which consists of an encoder and a decoder part both linked with skip connections. The encoding process follows a typical CNN, which means an alternated application of several convolutions and sub sampling operations. The subsampling can be realized by using either max pooling layers or strided convolution layers. It is suggested in this architecture to increase the number of feature maps within two consecutive subsampling operations by a certain factor. With every subsampling layer the resolution of the feature map decreases. From the encoding follows the decoding process, which successively increases the resolution of the feature maps again by applying upsampling layers, for instance up pooling layers or transposed convolutions. In addition to that, after every upsampling operation, the upsampled result is concatenated with the corresponding feature maps of the encoding path. Due to this concatenation, information can be transferred and integrated into the decoding process. The concatenated feature maps are then forwarded to several convolutions, decreasing the number of feature maps by the inverse of the factor they got multiplied in the encoding. The decoding path is therefore symmetrical to the encoding path, hence the name 'U-Net' [Ron15].

The U-Net architecture, like many other deep learning models, is scalable and configurable to a certain extent. One contribution of the thesis at hand is the systematic search for an ideal set of either optimization related parameters, as the learning rate or the batch size, but also on architectural parameters, which are described in the table 4.2. Since the systematic search proposed in this work, distinguishes on the one hand between discrete and continuous parameters and on the other hand boolean parameters, the table 4.2 also shows the domain of each parameter.

4.2.2 U-Net with 4D-Extractor

The U-Net architecture presented in section 4.2.1 is intended to receive the parameter maps for the individual slices as inputs and predicts one segmentation map each. Technically, U-Net allows to receive the 4D data of the CTP acquisition as well, also in addition to the paramter maps. Since a 4D volume typically consists of 30 or more time slices, the parameter maps would be highly undersampled, as the time slices would be oversampled. Early publications [Rob18] were researching in that direction, but are nowadays outperfomed by recent ones [Son18]. The network that won the ISLES Challenge 2018, was proposed by Song et al. in [Son18], and consists of three

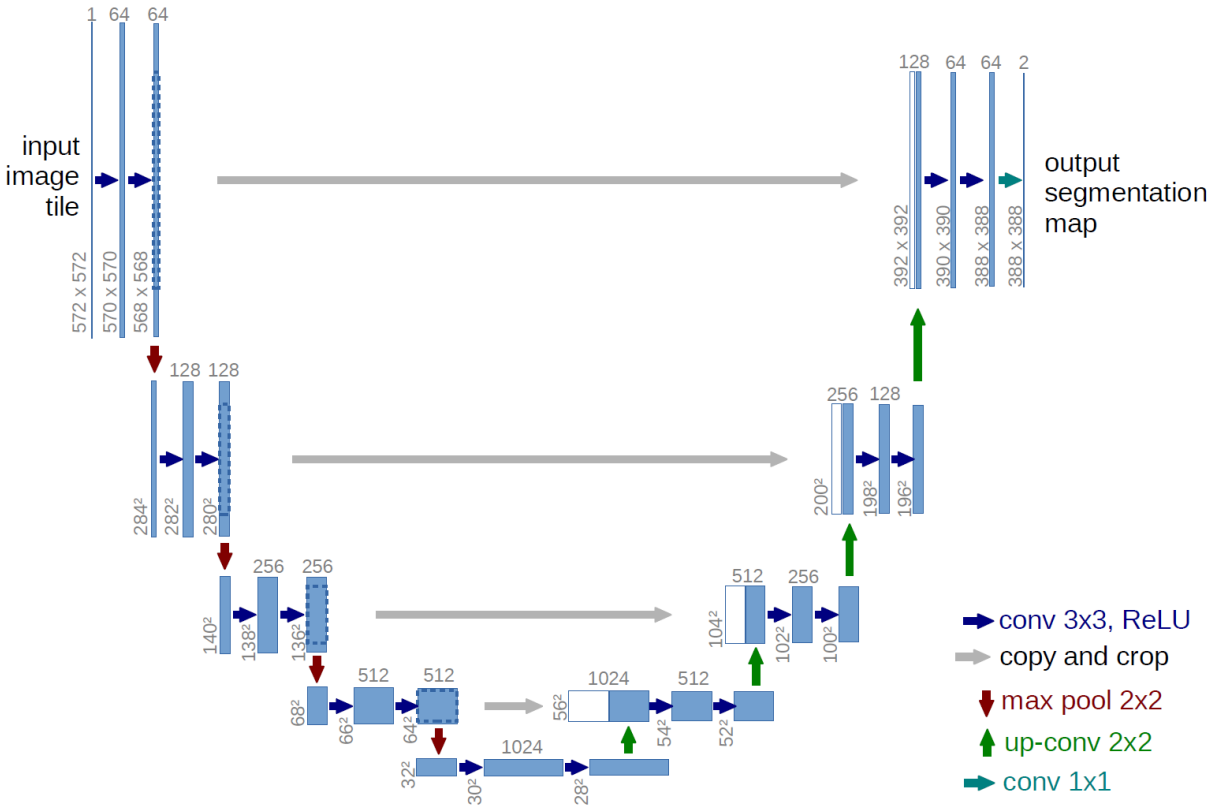

Figure 4.5: U-Net architecture [Ron15] receives in this example 572 x 572 images, compresses them to 32 x 32 in the encoding and expands them to a resolution of 388 x 388 in the decoding path. Each blue box represents the intermediate feature maps between the layers, with their number of channels written on the top of each box. The resolution is marked vertically next to each box. White boxes are feature maps that were copied according to the arrow next to them. The different kinds of arrows, denote different operations according to the legend [Ron15]. According to table 4.2, this network would be classified as follows: Channels = 64, Depth = 4, MaxPool = True, UpConv = True, BatchNorm = False, Residual = False.

stages. In the first stage the network compresses the 4D data of one spatial slice to one single feature map before merging it with the other parameter maps, using a dedicated U-Net. In the context of this challenge only the maps TMAX, CBV, CBF and MTT were available. The authors have named that subnetwork “extractor“, because by theory, this model learns which information have positive impact on the segmentation outcome, thus it must extract the relevant information based on the 4D data. In the second stage, the concatenation of parameter maps and the extraction is used to predict a virtual DWI image, as an intermediate step, again with an U-Net. The third stage, finally segments the core lesion based on that virtual DWI image. As mentioned in section 4.1.2, the ISLES Challenge data contains volumes of poor registration. As the proposed network

Table 4.2: Architectural parameters of U-Net

Parameters	Description
Depth	Number of pooling/upsampling Operations. Restricted by the resolution of the input image. Typically the number of feature map doubles per each downsampling, thus this parameter determines the total size of trainable parameters.
Channels	Number of channels of the first convolution layer. Like the depth, this number is determining the total size of the network in terms of its paramters.
MaxPool	Boolean, if true then max pooling operations are used for downsampling, otherwise strided convolutions are used.
UpConv	Boolean, if true then up pooling layers combined with one convolutional layers are used for upsampling, otherwise a transposed convolution is used.
BatchNorm	Boolean, if true a batch normalization layer is used between the convolution layer and the activation.
Residual	Boolean, if true then residual connections are used, within two subsequent downsampling and upsampling operations. A residual connection adds the input of one layer to the output of a deeper layer, hence certain layers are skipped. According to a common theory, residual connections perserve the gradients strength during backpropagation [He16]

predicts the DWI image intermediately, the registration mismatch is being implicitly learned by the second stage, and so it happens that improvements in the total segmentation outcome are observable. On request, the organizers of the challenge refused to publish the DWI data, to enforce solutions that process CTP data only.

In this thesis, a simplified version of Song et al's work is being presented. Image 4.6 shows extractor which compresses the 4D data to one feature map, which gets concatenated with all (9) parameter maps. An U-Net is used to predict the segmentation directly instead of predicting the DWI data as an intermediate result. The 4D was interpolated with the proposed method in section 4.1.3 onto 48 time steps per slice. The extraction network and its configurations can be found in the appendix A.1.

4.2.3 Post-Processing

The output of the network are floating point values ranging from 0 to 1 representing the probability of how likely the respective pixel belongs to the core class. In order to compare this continuous

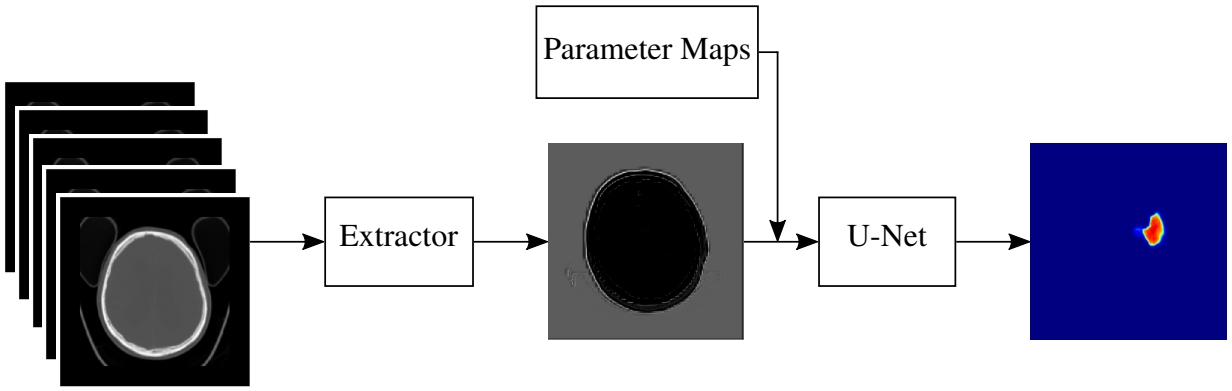

Figure 4.6: Data pipeline of the U-Net architecture enhanced with the extractor. The extractor is again one U-Net which receives the 4D data of one slice as input and reduces it to one single image. This image is concatenated with the parameter maps for the second stage, where a U-Net is used to segment the final core.

prediction to the state-of-art thresholding segmentation it is necessary to quantize the floating values to booleans. At the same time, a threshold value must be found that maximizes the segmentation according to an accuracy measure. Thus the problem can be formulated as follows:

$$T^* = \underset{T}{\operatorname{argmax}} \sum_{i=1}^{N} MIoU(\mathbf{y_i}, T(\hat{\mathbf{y}}_i)) \tag{4.1}$$

where T^* is the threshold value such that the sum of the MIoU between all predicted segmentation masks and their predictions is maximized. $T(\cdot)$ denotes the thresholding operation based on the threshold value T which ranges between 0 and 1. As optimization procedure, brute forcing the values for T in steps of 0.01 has been chosen. The evaluation was performed on the validation data sets. As the last post-processing step the prediction are linearly upscaled to their original pixel-spacing and resolution size. Since the threshold is to a certain accuracy, scale independent, the prediction are thresholded after they are upscaled, for a fair comparison between the state-of-the-art segmentation and deep learning results.

Chapter 5

Model Analysis

Model configuration on architectural basis and the choice of right optimization parameters is crucial in the development of deep learning models. The aim of this chapter is to present a method that enables a statistical analysis of the impact of certain boolean hyperparameters on the total segmentation outcome (section 5.1.1). As a refinement method for discrete parameters, the Bayesian Optimization published in [Sno12] is presented in section 5.1.2. The chapter ends with the presentation of an automated method to analyze the relevant perfusion parameters for the core segmentation based on the LRP algorithm and DeepLift approach.

5.1 Systematic Model Parameterization

5.1.1 Correlation Analysis on Boolean Parameters

Hyperparameter search, either on grids or random, are common practice in the field of deep learning. In a typical scenario, the top performing model that emerged by that search, is being considered to be the best performing model in general. Since models are randomly initialized and a non-convex optimization problem is underlying, it cannot be guaranteed, that the top performing model is not an outlier. Furthermore, after the search has been done, it is still not clear, which parameters have beneficial impact on the total performance and which not. In the research field of deep learning, it became common practice to simply accept the architectural constellation without questioning the impact of certain parameters. However, in this section a method is presented, enabling the analysis of certain architectural decisions on the segmentation outcome. The proposed method, consists of two phases and are explained in the following.

In the first phase, several networks are trained using the grid search method. This particular

grid search covers all possible permutations of all boolean parameters, each with a different subset of discrete parameters, the so called search table. The search table used in this thesis for the U-Net in its original version published in [Ron15] and in the extractor version is shown in table 5.1.

Table 5.1: Search table showing the different architectures

Type	Parameter	Space	Number
Binary	MaxPool	$[0, 1]$	2
	UpConv	$[0, 1]$	2
	BatchNorm	$[0, 1]$	2
	Residual	$[0, 1]$	2
Discrete	Depth	$[2, 3, 4]$	3
	Channels	$[16, 32]$	2
	Learning Rate	$[1 \times 10^{-3}$,1×10^{-4},$1 \times 10^{-5}]$	3
	Batch Size	$[8, 16]$	2

Each of the 16 networks defined by the binary permutations gets one of the 36 unique discrete parameter sets. 576 networks are trained in total. Every network is trained and validated on the same data split for training and validation, each with 100 epochs using randomized orders for data feeding. No early stopping was used. For each training cycle, the network that performed best on the validation set, in terms of the validation loss was stored. At the end of that grid search, 576 trained network models, were each models was the best performing network within the 100 epochs, are available. The result is a list similar to a highscore table, with rows as the models and columns the performance measures the respective model reached. These lists (up to the best 20 models) can be found in table B.1 for the vanilla U-Net and table B.2 for the U-Net with extractor.

The second phase consists of a correlation analysis. The general assumption, in fact a restriction in this approach, is the mutual independence of all parameters on the segmentation outcome. Since the parameters of interest are boolean and discrete values, uniformly distributed, correlation coefficients as Pearson's, are not recommended. Unlike the use of Pearson's correlation coefficient [Sed12], Spearman's rank correlation coefficient ρ [Spe61] is designed to work on trends and is not assuming any underlying distribution. Most parts of the following explanations are adapted from the Encyclopedia of Statistics by Yadolah Dodge [Dod08]. Spearman's rank correlation coefficient ρ is a nonparametric measure to determine the relation between two sets of data. Let $\{X_1, X_2, \ldots, X_n\}$ and $\{Y_1, Y_2, \ldots, Y_n\}$ be two samples of size n (here $n = 576$). For example, X_i is the best dice coefficient that the i-th model reached and Y_i is indicating whether batch normalization has been used in the i-th network, for $i = 1, 2, \ldots, n$. Furthermore, R_{X_i} denotes the statistical rank of X_i. $R_{X_i} = 1$ if X_i is the smallest value, here the worst Dice coefficient, and $R_{X_i} = 2$ if X_i is the second smallest value of X until $R_{X_i} = n$ if X_i is the largest

value. Exactly the same way, R_{Y_i} denotes the rank for the set Y [Dod08]. In general Spearman's rank correlation ρ is computed as follows:

$$\rho = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n\left(n^2 - 1\right)} \tag{5.1}$$

$$d_i = R_{X_i} - R_{Y_i} \tag{5.2}$$

If several samples of X or Y have the same values, d_i is not a representative measure anymore, which arises the necessity of using average ranks. If there are many average ranks, it is beneficial to correct them accordingly with

$$\rho = \frac{S_x + S_y - \sum_{i=1}^{n} d_i^2}{2\sqrt{S_x S_y}} \tag{5.3}$$

where

$$S_x = \frac{n\left(n^2 - 1\right) - \sum_{i=1}^{g}\left(t_i^3 - t_i\right)}{12} \tag{5.4}$$

with g representing the number of groups with average ranks and t_i the size of group i for sample X and

$$S_y = \frac{n\left(n^2 - 1\right) - \sum_{j=1}^{h}\left(t_j^3 - t_j\right)}{12} \tag{5.5}$$

with h representing the number of groups with average ranks and t_j the size of group j for sample Y. The Spearman's rank correlation coefficient has values between -1 and $+1$, where a positive value indicates a positive correlation between X and Y, vice versa for negative values. The perfect correlation values either -1 or $+1$. No correlation is present if the value is close to 0 [Dod08]. The Null-Hypothesis H_0 claims, X and Y are mutually independent. For further proceedings, those boolean parameters are chosen, that have positive impact on the performance measures according the correlation coefficient, given statistical significance. If no significance is present, the boolean value that leads to the simpler architecture is then the preferred one, following the idea of "Occam's Razor" [Blu87]. The significance threshold in this approach valued 0.05.

Once all trainings according to table 5.1 are done, models that were not able to model the problem at all (with a MIoU below 0.5), were excluded from the correlation analysis. Section 5.1.1 presents the correlation coefficients for all parameters and metrics for U-Net with and without extractor, as well as the conclusion deduced from that analysis.

5.1.2 Bayesian Optimization on Discrete Parameters

Spearman's rank correlation coefficient is well suited for binary variables, e.g. MaxPool or BatchNorm. But variables of discrete (batch size) or continuous (learning rate) nature are underrepresented in table 5.1. A statistical way of approaching this problem is the use of the bayesian optimization algorithm, which is well suited for optimization problems of expensive black-box derivative-free functions [Fra18]. And so is the grid search problem a problem of that kind, whose goal is to find the best parameters. Let $g(p_i) : \mathbb{R}^{\dim(p)} \to \mathbb{R}$ denote the training of one network as observation, by choosing the i-th sample of parameters p_i, such that

$$l_i = g(p_i) \tag{5.6}$$

where l_i is best MIoU value which has been achieved in e.g. 100 epochs of training [Bro10]. $g(p)$ is the objective function to maximize

$$p^* = \underset{p}{\operatorname{argmax}}\, g(p) \tag{5.7}$$

The Bayesian optimization consists of two further components. Firstly, a statistical model, describing the bayesian posterior probability distribution of the potential values of $g(p)$ for a given point p, here an invariably Gaussian Process (GP). As new values for $g(p)$ are observed, the Bayesian model is being updated. Secondly, an acquisition function $u(p|\mathcal{D}_{1:t})$, that predicts most likely value of $g(p)$ given p, based on the current posterior probability over g and all previous observations $\mathcal{D}_{1:t} = \{p_{1:t}, l_{1:t}\}$ from the first iteration to the t-th iteration [Fra18, Bro10]. Figure 5.1 shows the GP and its estimation for $g(p)$ with three sample points. The algorithm starts by evaluation of n_0 random samples, uniformly distributed to set up an initial space for posterior probability distribution. The algorithm continues by allocating the remainder of a budget of N evaluations and chooses p according to the acquisition function and keeps updating the posterior probability distribution and accumultes the observations $\mathcal{D}_{1:t}$ by the current one [Fra18]. The Bayesian optimization algorithm is shown in algorithm 2.

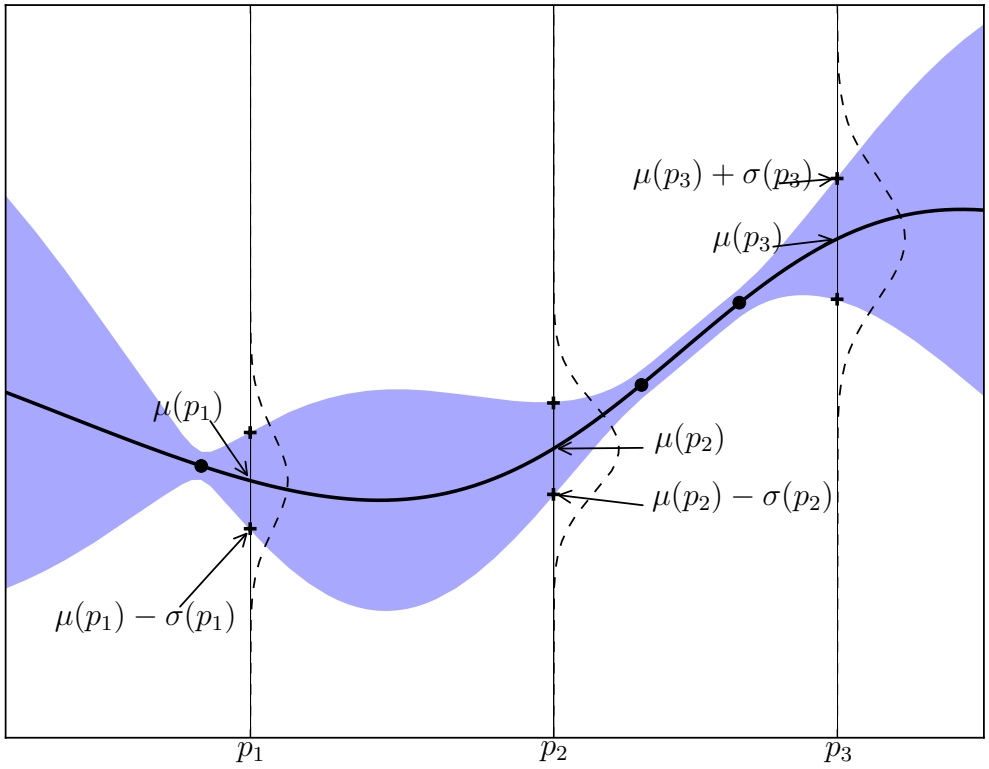

Figure 5.1: Illustration of a 1D Gaussian Process (GP) at three points of the objective function $g(p)$. The solid black line indicates the prediction of $g(p)$ based on the surrogate means of the GP. The blue hull curves, show the confidence intervals of the mean values with plus minus the variances. The superimposed Gaussians to the GP mean and standard deviation ($\mu(\cdot)$ and $\sigma(\cdot)$) of the predictions at the points, $x_{1:3}$ [Fra18, Bro10].

Algorithm 2 Basic Bayesian Optimization [Fra18, Bro10]

Place a GP prior to g
$\mathcal{D}_{0:0} = \emptyset$
for $t = 1, 2, \ldots N$ **do**
    if $t > n_0$ **then**
        Find p_t by optimizing u over the GP: $p_t = \text{argmax}_p \, u\,(p|\mathcal{D}_{1:t})$
    else
        Choose random sample p_t
    end if
    Evaluate new sample with the objective function: $l_t = g(p_t)$
    Update observations $\mathcal{D}_{0:t+1} = \{\mathcal{D}_{0:t}, (p_t, l_t)\}$
    Update GP
end for
return Point p_t that yielded the largest l_i

The proposed method for a systematic model parametrization can be summarized as follows.

1. The first step consists of a grid search that entirely covers all boolean combinations, each with a subset of different values for the discrete parameters. The search table denotes all permutations needed to evaluate. Once, the grid is evaluated, correlations are being computed between the architectural parameters and the performance metrics, using the Spearman's rank correlation coefficient. If one parameter is statistically significant, the value is chosen, as the correlation coefficient suggests for beneficial impact. If no statistical significance is present, the parameter is chosen, that results in the simpler architecture.

2. The second step further refines discrete parameters like the depth or the batch size as well as continuous parameters like the learning rate. The optimization is done by the Bayesian Optimization algorithm, that utilizes Gaussian processes to model, the performance of one architecture. By iterative sampling, the underlying posterior probability is updated and is used to predict the next sampling using a so called acquisition function.

5.2 Relevant Parameter Analysis

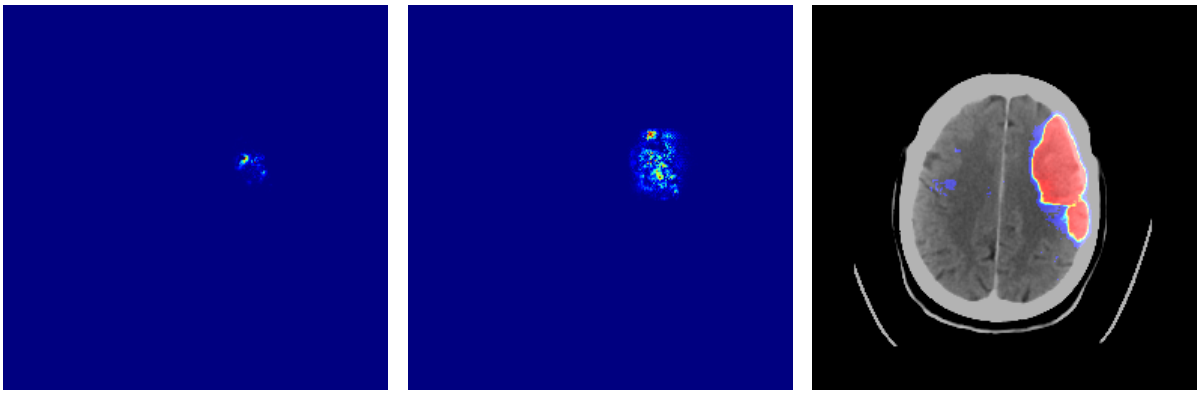

Figure 5.2: Three images based on the same slice of the validation dataset. Left: Relevance map of the CBV channel computed using the LRP algorithm and the $\alpha\beta$-rule. Center: The DeepLIFT relevant map of the CBV channel on the same slice as left. Right: Overlay of the slice (BASE) with the non-thresholded prediction (blue ≈ 0 and red ≈ 1) of a vanilla U-Net model.

The LRP algorithm and DeepLIFT are methods used to identify input pixels that contributed the most to the output. Both methods return feature maps, denoted as "relevance maps", according to each input channel. Image 5.2 shows the relevance map of the CBV channel computed with the LRP algorithm (left) and the DeepLIFT method (center). For a comparison, the right image shows

the original input (only BASE) with the prediction of the U-Net model overlayed. The relevance maps are of qualitative nature, which disables a fair comparison between both approaches and various inputs. The thesis was amongst other reasons motivated by the question, which perfusion parameter are of importance in the segmentation of the ischemic core lesion. This section proposes a method that addresses the issue of finding the most relevant channels - across different methods and datasets. The method is driven by three restricting assumptions. 1) One particular channel is either relevant or not relevant. 2) Every dataset has always κ (out of c) channels which are considered to be relevant, and $c-\kappa$ channels which are not relevant. 3) The most relevant channels are assumed to be constant within one dataset. This means that the most relevant channels are computed dataset wise. The reasons for these assumptions are explained at the end of this section. Let $D = \left\{\mathbf{D_i} \in \mathbb{R}^{s\times h\times w\times c} | i = 1, \ldots, N\right\}$ be entirety of in total N datasets, where each dataset $\mathbf{D_i}$ has s slices, a height h , a width w and c channels. s varies between the different datasets, but for simplicity is depicted as constant. Further, let $R = \left\{\mathbf{R_i} \in \mathbb{R}^{s\times h\times w\times c} | i = 1, \ldots, N\right\}$ be the corresponding relevance maps for each dataset. R can either be computed using the LRP algorithm or DeepLIFT. Let q_i be the boolean set, which indexes the c channels with 1 if they are relevant and 0 otherwise. This is the embedding for assumption 1).

$$q_i = \{q_{ij} \in \{0, 1\} \,| j = 1, \ldots, c\} \tag{5.8}$$

$$\text{where } q_{ij} = \begin{cases} 1 & \text{if channel } j \text{ is relevant} \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

Assumption 2) is embedded by the following:

$$\kappa = \sum_j q_{sj} = \sum_j q_{tj} \quad \forall s, t = 1, \ldots, N \tag{5.10}$$

where κ denotes the number of relevant channels. The proposed procedure, named the Channel-Wise Relevance Counting Algorithm shown in algorithm 3, counts how often each channel is relevant using a deep learning model f.

To compute r_c the Frobenius norm on $\mathbf{R}_i$ adds up the height, the width and the slices, whereas the latter embeds assumption 3). In the last step of Algorithm 3, the sum implements the counting of how often which channel was considered as being relevant. The result is a vector $\boldsymbol{q}$ which contains the count for each channel. Since the most relevances differ between the different architectures and trainings, this algorithm is applied to all models of the search table 5.1 that achieved a certain perfomance score (here MIoU > 0.55). For each model the resulting q values

Algorithm 3 Channel-wise Relevance Counting

procedure CRC(D, a deep learning model f)
  for $i = 1, 2, \dots N$ **do**
    $\mathbf{R}_i = \text{LRP}(f, \mathbf{D_i})$ or $\mathbf{R}_i = \text{DeepLIFT}(f, \mathbf{D}_i)$
    $\mathbf{r}_c = \|R_{i,s,h,w}\|_F$ Energy of R_i across the slice, height and width axis, $\mathbf{r}_c \in \mathbb{R}^c$
    Set the largest κ entries of $\mathbf{r}_c$ to 1 and 0 otherwise. This returns $\mathbf{q}_i$
    Store $\mathbf{q}_i$
  end for
  $\mathbf{q} = \sum_{i=1}^{N} \mathbf{q}_i$
  return $\mathbf{q}$
end procedure

are added up again and are shown in table C.1 and C.2.

LRP and DeepLIFT are both methods, that can only compute relevance values under certain assumptions (see section 3.3.1 and 3.3.2). However, they still cannot deliver quantitative measures of the actual relevance, due to the difficulty of defining relevance in general. The difficulty of measuring relevance leads to assumption 1), which replaces the interpretation of continuous values for relevances into a classification, which is easier to follow as described later. On top of that, both algorithms deliver different values for each dataset at hand. Due to assumption 1), a threshold must be defined that classifies the channel to either relevant or irrelevant. Since the values are not comparable to each other, no global threshold can be found. Thus, assumption 2) allows to classify the relevance independent to the method and the dataset, although it may happens, that more or less channel than κ can actually be relevant. A closer look to one individual dataset reveals that only a certain part of the volume contains the necessary information needed to segment the final core. For example, some datasets contain slices of air only without any patient parts. Empty or useless slices can classified as irrelevant with a high confidence, which illustrates the meaning of assumption 1), whereas the quantitative relevance of such slices would be hard to estimate. As a consequence, it makes no sense to determine the most relevant channels on irrelevant empty slices. In addition to that, counting the relevant channels slice-wise would weight larger datasets with more slices with a higher impact to those volumes with smaller slice numbers. From this follows assumption 3), since all datasets are equally important.

Chapter 6

Results

This chapter presents the model evaluation, regarding the correlation analysis (section 6.1.1) and the relevance analysis (section 6.1.2). The chapter ends with a comparison of all approaches including the thresholding segmentation (section 6.2).

6.1 Evaluation

For the following evaluations, the data sets were split into two parts. The training data consisted of 123 volumes, the validation set of 20 volumes and the testing set of 3 volumes. All performance measures in section 6.1 were computed based on the validation set. In order to enable the computation of classification metrics, a threshold of 0.5 has been used to compute the binary masks. Every network involved in this evaluation was trained 100 epochs. All deep learning related computations were based on TensorFlow (version 1.14) [Aba15]. The construction and the training/testing of the various models was done using Keras (version 2.2.4) [Cho15]. The Spearman's rank correlation coefficient was computed with Scipy (version 1.2.0) [Jon01].

6.1.1 Correlation Analysis

This section describes the results using the correlation analysis presented in section 5.1.1. For this analysis only models from the search table 5.1 were considered that achieved a MIoU higher than 0.5 out of 100 epochs training to exclude networks that were not able to train at all. Table 6.1 displays the Spearman's rank correlation coefficient of the U-Net architecture and table 6.2 the coefficients for the architecture with extrator respectively. Cells with coefficients which are statistically significant were highlighted as green. As presented in section 5.1.1 the

experimental setup is designed for the analysis of the boolean architectural parameters. The correlation coefficients in both tables can be interpreted as follows. For example, the correlation coefficient between BatchNorm and Dice (0.22) is greater than 0 and shows statistical significance. This indicates that an increase of the Dice value can be expected, if BatchNorm is set “True“ (1). Notice, the loss has been negated for the following tables, since the loss value is the only presented measure which is preferred to be as negative as possible. Thus, the negation allows to interpret the loss correlation the same way as the others. All computed correlations, with statistical significance, were in their strength weaker than expected. The highest correlation value was 0.27. Thus, no strong correlations (close to 1) occurred. Nevertheless, the results were interpreted as the following.

For the vanilla U-Net architecture, the use of batch normalization shows a statistical significance across all metrics. However, batch normalization has beneficial impact regarding the Loss, Dice, MIoU and recall, whereas it causes a decrease in the volumetric similarity and precision. Despite this contradictory outcome, using batch normalization is still the preferred architectural decision, since recent works have experimentally shown beneficial influence by using it [San18]. Moreover, Dice, MIoU and recall are weighted higher in this specific decision. UpConv and MaxPool have neither high correlation rates nor statistical significance, except the correlation between recall and MaxPool. Still, using upsampling layers (UpConv = True) instead of transposed convolutions and Max Poolng layers (MaxPool = True) is the preferred parametrization, since this setting leads to a simpler architecture (in terms of lower space- and time-complexity). The use of residual connections shows a significant beneficial correlation on all metrics, except recall where no significance could be observed.

Table 6.1: Spearman’s rank correlation coefficients for the U-Net architectures. Based on the search table 5.1, 301 out of 567 achieved a MIoU score higher than 0.5 and were included in the statistical analysis. Entries highlighted with green showed a statistical significance (< 0.05).

	BatchNorm	UpConv	MaxPool	Residual
-Loss	0.18	0.02	-0.05	0.27
Dice	0.22	0.05	-0.07	0.23
MIoU	0.17	0.04	-0.09	0.18
VS	-0.22	-0.09	0.09	0.23
Precision	-0.17	-0.06	0.11	0.15
Recall	0.25	0.105	-0.13	-0.06

Concluding the correlation analysis, BatchNorm, UpConv, MaxPool and Residual were set True for the following Bayesian Optimization. The parameters to optimize were the learning rate,

the batch size and the depth. After 25 random iterations and 25 iterations in the optimization procedure, the best network had a depth of 3, a batch size of 7, 16 feature maps as starting channels and a learning rate close to 0.0002. The best network had in total 670090 parameters.

With respect to the U-Net architecture with extractor, the use of batch normalization has beneficial and statistical significant influence on the loss, Dice and recall. Therefore, batch normalization was enabled for the subsequent evaluations. UpConv has a significant correlation close to 0 between precision and recall, means UpConv has no impact on the metric. MaxPool shows no significance across all measures. Since the use of UpConv and MaxPool lead to the simpler architecture both are set to True, as with the vanilla U-Net architecture. The use of residual connections had beneficial impact on the volumetric similarity and precision, but caused low reductions in recall. However, residual connections were used in the following considerations, despite the fact that it showed weak improvements, but this enables a fairer comparison to the plain U-Net model.

Table 6.2: Spearman's rank correlation coefficients for the U-Net architectures. Based on the search table 5.1, 463 out of 567 achieved a MIoU score higher than 0.5 and were included in the statistical analysis. Entries highlighted with green showed a statistical significance (< 0.05).

	BatchNorm	UpConv	MaxPool	Residual
-Loss	0.22	-0.01	0	0.02
Dice	0.10	-0.02	-0.05	0
MIoU	0.08	-0.02	-0.07	-0.01
VS	-0.08	0.01	0.04	0.19
Precision	-0.06	-0.06	0.10	0.17
Recall	0.11	0.03	-0.11	-0.11

As a result of the correlation analysis, just as for the U-Net model, all boolean parameters were set True. Bayesian Optimization was applied to the same variables as with the U-Net version. The best network had a depth of 2, was trained with a batch size of 8, was using 16 feature maps and had a learning rate close to 0.0001.

6.1.2 Relevance Analysis

For the relevance analysis, the Channel-wise Relevance Counting Algorithm (algorithm 3) was applied on all models based on the search table 5.1 which achieved a higher MIoU than 0.55. This threshold of 0.55 has been chosen firstly to involve the better performing networks into this consideration and secondly to reduce the computational effort for this analysis. The algorithm calculated the relevance maps with the DeepLIFT approach (section 3.3.2) and the LRP algorithm

(section 3.3.1) using either the $\alpha\beta$-rule (see equation 3.46) and the z-rule (see equation 3.43). It was assumed that always three channels are relevant per data set ($\kappa = 3$). With a larger value for κ the analysis will result in more channel counts and more distinctive results are expected. As this is just an assumption, the analysis must be repeated for $\kappa = 2$ and $\kappa = 1$ as well, but could not be covered in this thesis, due to the large computational effort of this procedure.

294 out of 576 U-Net models and 456 out of 576 U-Net models with extractor achieved a higher MIoU score than 0.55 and were included in this analysis. A listing of the counting results is shown in the appendix in table C.1 and C.2. In order to compare both architectures based on the absolute counting values, the extractor variant has been evaluated in addition to table C.2 for the best 294 models. The sum over all three relevance map algorithms is shown per channel in figure 6.1. Across both architectures MIP appears to be the most prominent channel with higher importance for the U-Net architecture with 11420 counts and 8679 counts for the extractor model. Followed by MIP, TMAX is the second most relevant channel for the U-Net model with 7939 counts and TTD respectively for the extractor enhanced version with 8041 counts. Noticeable is the symmetry that occurs for the third most relevant channel. Namely, TTD appears to be relevant for U-Net as well with 7887 counts on the third place, whereas the third most relevant channel for the enhanced architecture is TMAX with 7035 counts. On the same count range is BASE with 7477 counts for U-Net as the forth relevant channel, CBV respectively for the extractor model with 6448 counts. A clear irrelevance shows CBV for the U-Net architecture with 2429 counts and CBF across both architectures (277 for U-Net, 1081 for U-Net + extractor) are observable. For further evaluations, MIP, TMAX, TTD and BASE were considered to be the most relevant channels.

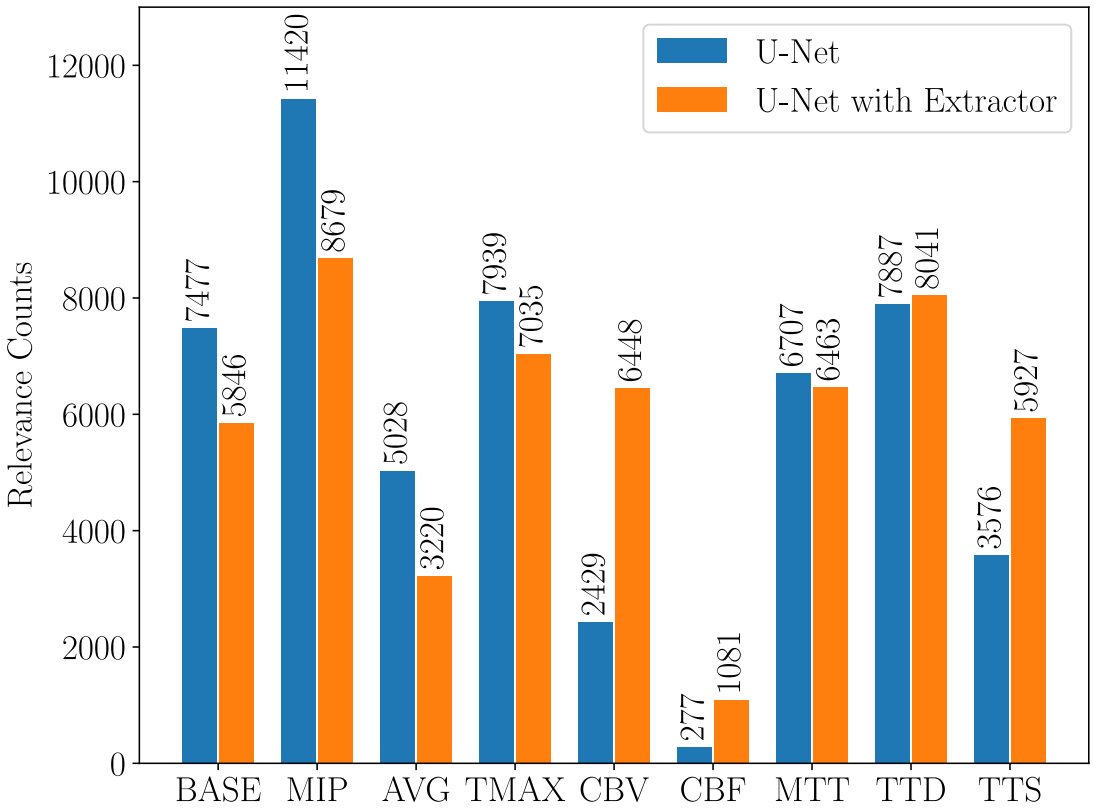

Figure 6.1: Channel-wise Relevances for U-Net and U-Net with Extractor. For this evaluation the best 294 out of 576 models have been considered, since 294 U-Net models achieved a MIoU score than 0.55.

6.2 Comparison of all Approaches

As the evaluation in section 6.1.2 showed, BASE, MIP, TMAX and TTD were the most relevant channels. Furthermore, the networks were also trained on all channels, on MIP, TMAX and TTD, on MIP and TMAX and on TMAX only. Since the medical litarature suggests also to take CBV for the core and CBF for the penumbra into account, both channels were considered in this evaulation as well. For the thresholding, Abels et al.'s threshold presented in section 2.2.2 has been used. These threshold levels are the default values in the neuro imaging software syngo.via by SIEMENS Healthcare GmbH and hence are recommended by SIEMENS. Image 6.2 shows a BASE slice from the testing data (of the evaluation in section 6.1.1), overlayed with the label (center), with the mentioned thresholding segmentation (left) and with the segmentation result of an U-Net. The network used for this segmentation was neither trained nor validated on the data set of image 6.2. In this example the thresholding approach leads to clustered and noisier segmentations, whereas the segmentation of the U-Net is rather compact but partially dislocated.

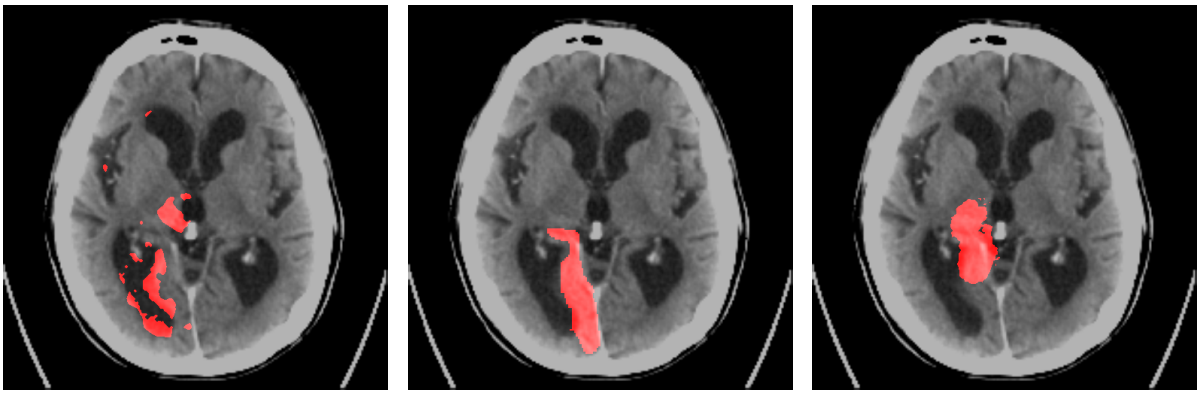

Figure 6.2: Test image overlayed with corresponding label (center) showing an overlay of the thresholding segmentation on the left and the U-Net segmentation on the right. The thresholding segmentation leads to nosier clusters, in contrast to the U-Net segmentation which is here rather compact but dislocated.

In order to compare the various architectures including the thresholding approach, 5-fold cross validations were performed. In this setup the data was three folds were used for training, one fold for validation and one fold for testing purposes. In each round of the 5 iterations of the cross validation, another 1/5 partition of the entire data is used for validation and testing while the remaining 3/5 of the data is left for training. This is repeated five times so every partition was the validation set once. Each architecture with their channel constellation was trained and validated with such a 5-fold cross validation. Since the training underlies a stochastic process, each cross-validation was repeated twice, so at the end each model was trained 10 times in total.

With every training, the respective best performing (on the validation fold) model within 100 epochs was saved and tested on the respective test fold. The average test performances per each metric is listed in table 6.3.

Table 6.3: Performance of the U-Net architecture with and without extractor trained on different channels. The data was split in a 5-fold cross-validation setup where 3/5 was for training purposes, 1/5 for validation and 1/5 for testing. The model that performed best on the validation data was chosen to be evaluated with the test data. The numbers shown here are based on the test data performance. Every cross validation was repeated 2 times. Across the different rows, the cross validation data splits were identical for a fair comparison. The thresholding segmentation was also included in order to compare the results with the state-of-the-art method. According to each metric, the best values are highlighted with a bold font.

Model	**Loss**	**Dice**	**MIoU**	**VS**	**Precision**	**Recall**
U-Net + Extractor						
All channel	0.788	0.382	0.623	0.806	0.408	0.420
BASE MIP TMAX TTD	0.849	0.303	0.593	0.774	0.298	0.410
MIP TMAX TTD	0.848	0.305	0.595	0.745	0.311	0.410
MIP TMAX	0.839	0.318	0.598	0.797	0.310	0.419
TMAX	0.867	0.274	0.583	0.742	0.306	0.348
CBF CBV	0.843	0.306	0.600	0.790	0.328	0.381
CBV	0.879	0.260	0.581	0.719	0.300	0.326
U-Net						
All channel	**0.755**	**0.425**	**0.638**	0.889	**0.450**	0.434
BASE MIP TMAX TTD	0.819	0.351	0.608	0.744	0.345	0.467
MIP TMAX TTD	0.840	0.327	0.598	0.736	0.307	**0.471**
MIP TMAX	0.833	0.331	0.600	0.796	0.318	0.429
TMAX	0.841	0.317	0.596	0.742	0.297	0.449
CBV CBF	0.780	0.390	0.625	0.835	0.418	0.419
CBV	0.779	0.397	0.627	0.863	0.416	0.415
Thresholding	1.161	0.198	0.551	**0.973**	0.204	0.193

Using all channels with the U-Net architecture in its original form showed best performances regarding the loss, Dice, MIoU and precision. The U-Net variant using MIP, TMAX and TTD achieved the best recall. The best volumetric similarity was achieved by the thresholding approach. Unlike precision and recall of thresholding which is significantly worse compared to all considered deep learning architectures. From this it follows that the thresholding approach leads to segmentations that in total rather have the same volume sizes as the actual infarct region, but are rather falsely located and noisier clustered which explains the low precision and recall. In image 6.2 and D.1 are examples of the noisy thresholding segmentation. All networks trained on a reduced number of channels, showed consequently a weaker performance. Furthermore, a

comparison between the CBV versions and the networks with most relevant channels (BASE, MIP, TMAX, TTD) shows no significant difference in performance regarding the extractor version. In contrast to the extractor version, the U-Net version performed better on CBV and CBF or CBV only compared to the relevant channels, despite the fact that CBV and CBF were considered as irrelevant in the analysis of section 6.1.2.

Chapter 7

Summary and Outlook

The overall goal of this thesis was the segmentation of the infarcted core in CT perfusion data using deep learning models. The underlying data is a crucial component in the training of deep neural networks. Thus, one goal of this thesis was the analysis of the CTP data and the labels from various sources. This work has shown, interpolation for rescaling purposes as it is required to enable the processing of multiple sources, causes artifacts and may introduce new artificial infarcted regions. From this follows the necessity of a pre-processing algorithm that has been proposed in this thesis, which automatically combines different interpolation techniques to change the resolution of the data at hand, while keeping the amount of artifacts as low as possible.

As a simplification of Song et al's work [Son18] which won the ISLES 2018 challenge, a new deep learning architecture has been proposed. This architecture builds on top of Ronneberger et al's U-Net [Ron15]. The idea of the new architecture is the use of a so called extractor that compressed the cine data to one slice, which is being concatenated to the perfusion maps. Both sources are used in order to segment the final core using a U-Net. Since both architectures provide a variety of different hyperparameters a method has been proposed that allows the systematic decision on boolean parameters. This analysis has shown that the use of batch normalization and residual connections have statistical significant beneficial impact on the segmentation outcome. Further, experiments have shown, no superior performance of the extractor network over the U-Net architecture could be achieved.

Moreover one goal of this thesis was the analysis, which perfusion channels have the most relevance on the segmentation outcome. For this interrogation an algorithm has been proposed, the Channel-wise Relevance Counting Algorithm. Under certain assumptions, this algorithm determines the channels of highest contribution based on the Layer-wise Relevance Propagation Algorithm [Lap19] and the DeepLIFT [Shr17] method. An evaluation of the algorithm on the

U-Net architecture as well in its enhanced version with the extractor network, showed that MIP, TMAX and TTD are the most relevant and CBV and CBF are the least relevant channels. Respective networks trained either with the relevant channels, and those which are recommended by literature (CBV, CBF) were trained in a cross-validation set up. Regarding the U-Net architecture using CBV and CBF outperformed MIP, TMAX and TTD, although they were evaluated as irrelevant. The U-Net model with extractor showed no significant difference between both channel variations. As a conclusion, the functionality of the Channel-wise relevance Counting Algorithm could not experimentally be confirmed.

Furthermore, one goal was to compare the deep learning segmentation with the state-of-the-art thresholding method. A cross-validation experiment allowed such a comparison. Across all considered deep learning models the thresholding segmentation returned significantly worse segmentations in terms of Dice, MIoU, precision and recall. U-Net combined with all channels performed best and achieved a Dice of 0.425 and a MIoU of 0.638 whereas the thresholding achieved a Dice of 0.198 and a MIoU of 0.551. In one measure, namely volumetric similarity, thresholding outperformed all models with a score of 0.973 against the best deep learning value of 0.889 (U-Net with all channels).

As certain steps could not be covered in the frame for this work, the following denotes suggestions for further work on this topic. It is suggested to measure the actual impact of the proposed pre-processing on the total segmentation outcome, probably in a cross-validation set up. Especially the impact of having isotropic voxels might be subject of this investigation, since no detailed tests have been performed in this direction so far. Isotropic voxel lead to comparable anatomical scalings, but reduce the information density due to downscalings. Furthermore, the correlation analysis using the Spearman's rank correlation coefficient should be extended, such that inter-variable correlations are being under consideration as well. So far, the proposed analysis assumes that all hyperparamters behave mutual independent, which might not necessarily be the case. Due to a limited time frame for this thesis, the relevance analysis using the Channel-wise Relevance Counting Algorithm was only done with $\kappa = 3$ and should be repeated and compared with results using $\kappa = 1$ and $\kappa = 2$ as well. Furthermore, the Counting Algorithm should be extended using more approaches alongside DeepLIFT and LRP, e.g. with the DeepTaylor decomposition [Mon17].

Appendix A

Extractor Model

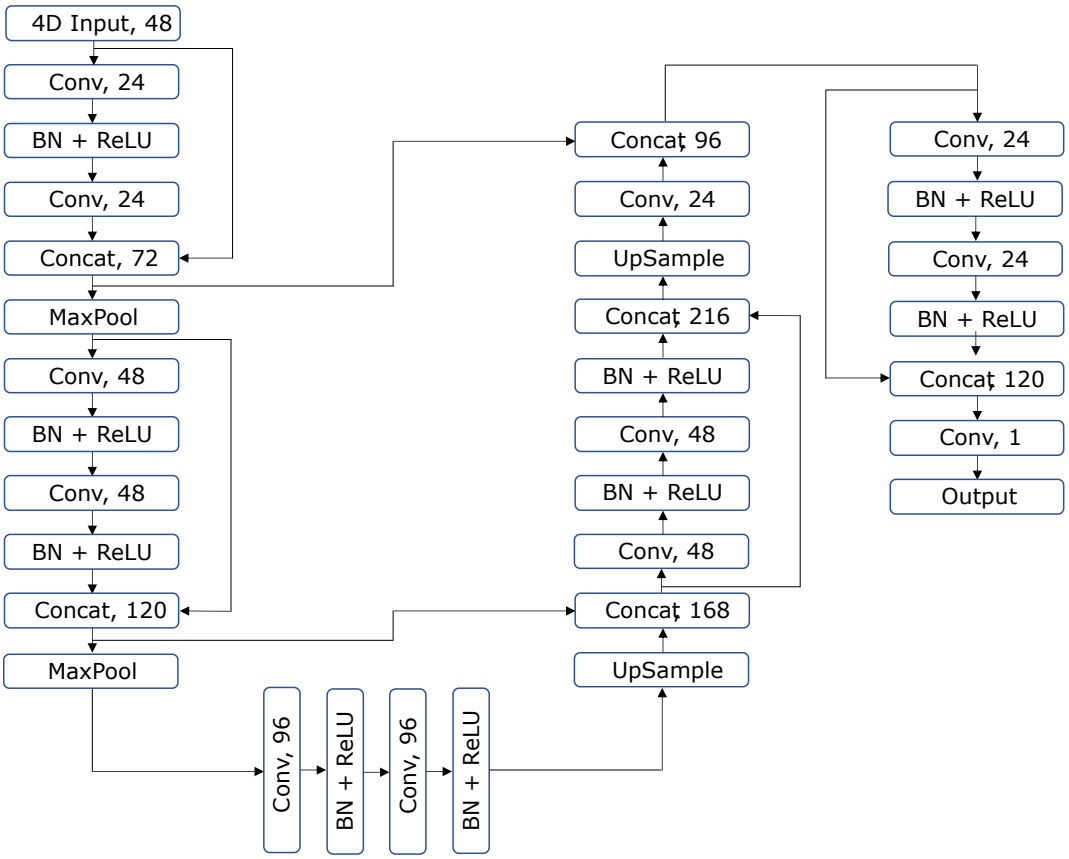

Figure A.1: Extractor network which is based on the U-Net architecture. It receives the 4D data of one slice as input (here 48 time steps, hence 48 channels) and reduces the data to one single channel. The number of feature maps the respective layer returns is written next to the layer type. If a layer is not changing the number of feature maps, no number is written next to it. All convolutional layers (abbreviated with 'Conv'), max pooling and upsampling layers have 3×3 kernels. All convolutional layers have a stride of 1×1 whereas the max pooling as well as the upsampling layers have a stride of 2×2. This architecture utilizes residual connections, batch normalization (BN) and ReLU layers. Concatenations are abbreviated with 'Concat'.

Appendix B

Correlation Tables

Table B.1: The best 20 U-Net models, sorted by descending Dice scores, after the grid search according to the search table. The correlation analysis is based on this table.

#	LOSS	DICE	MIoU	VS	Prec	Rec	Batch Norm	UpConv	Maxpool	Residual
1	0.744	0.452	0.645	0.897	0.510	0.415	1	1	0	1
2	0.755	0.445	0.642	0.804	0.379	0.563	1	1	1	1
3	0.764	0.435	0.637	0.921	0.406	0.476	1	0	1	0
4	0.761	0.428	0.636	0.805	0.367	0.545	1	1	0	1
5	0.765	0.427	0.635	0.757	0.351	0.577	1	1	0	0
6	0.762	0.427	0.637	0.962	0.423	0.456	1	1	1	1
7	0.764	0.427	0.635	0.861	0.382	0.506	1	0	1	0
8	0.761	0.427	0.638	0.808	0.371	0.548	1	1	0	1
9	0.764	0.426	0.635	0.913	0.401	0.477	1	1	1	0
10	0.771	0.425	0.632	0.787	0.354	0.546	1	1	1	0
11	0.765	0.424	0.635	0.832	0.372	0.522	1	0	0	0
12	0.769	0.424	0.634	0.818	0.367	0.531	1	1	0	1
13	0.770	0.424	0.635	0.845	0.378	0.516	1	0	0	1
14	0.766	0.424	0.635	0.899	0.395	0.485	1	0	0	0
15	0.768	0.424	0.634	0.806	0.363	0.538	1	0	1	1
16	0.770	0.424	0.633	0.823	0.366	0.523	1	1	1	1
17	0.763	0.424	0.635	0.777	0.356	0.560	1	1	0	0
18	0.765	0.423	0.635	0.804	0.365	0.544	1	1	0	1
19	0.768	0.422	0.634	0.937	0.407	0.462	1	1	0	1
20	0.780	0.422	0.632	0.845	0.370	0.506	1	1	0	1

Table B.2: The best 20 U-Net models with extractor, sorted by descending Dice scores, after the grid search according to the search table. The correlation analysis is based on this table.

#	LOSS	DICE	MIoU	VS	Prec	Rec	Batch Norm	UpConv	Maxpool	Residual
1	0.837	0.436	0.637	0.927	0.409	0.474	0	0	0	1
2	0.753	0.434	0.641	0.956	0.430	0.469	1	0	1	0
3	0.756	0.433	0.641	0.760	0.363	0.593	1	0	1	0
4	0.761	0.430	0.637	0.920	0.406	0.476	1	1	1	0
5	0.768	0.428	0.635	0.902	0.397	0.483	0	0	0	0
6	0.765	0.427	0.637	0.859	0.385	0.511	0	0	0	1
7	0.766	0.426	0.635	0.762	0.352	0.572	0	1	1	1
8	0.767	0.425	0.634	0.771	0.353	0.563	1	1	0	1
9	0.836	0.425	0.632	0.959	0.410	0.445	0	1	0	0
10	0.761	0.425	0.637	0.938	0.413	0.467	1	1	0	1
11	0.768	0.424	0.634	0.846	0.375	0.511	1	1	0	0
12	0.765	0.424	0.636	0.900	0.396	0.484	1	0	1	0
13	0.762	0.424	0.637	0.834	0.378	0.529	0	1	1	0
14	0.772	0.423	0.634	0.890	0.389	0.485	0	0	1	1
15	0.763	0.423	0.636	0.803	0.367	0.546	1	0	1	0
16	0.774	0.423	0.632	0.758	0.346	0.566	0	1	0	1
17	0.767	0.423	0.634	0.835	0.372	0.518	1	1	0	1
18	0.775	0.422	0.632	0.813	0.361	0.528	0	1	1	0
19	0.772	0.422	0.633	0.999	0.429	0.429	1	1	1	1
20	0.767	0.422	0.633	0.823	0.366	0.523	1	1	0	1

Appendix C

Relevance Counting Tables

Table C.1: Channel-wise Relevance Counting on U-Net architectures (based on the search table 5.1) on 293 (out of 576) models which achieved a higher MIoU score than 0.55.

Method	BASE	MIP	AVG	TMAX	CBV	CBF	MTT	TTD	TTS
DeepLIFT	2666	4006	1719	2355	968	201	2225	2428	1012
LRP$\alpha\beta$	1859	3075	1412	3330	463	50	2500	3120	1771
LRPz	2952	4339	1897	2254	998	26	1982	2339	793
Sum	7477	11420	5028	7939	2429	277	6707	7887	3576

Table C.2: Channel-wise Relevance Counting U-Net + Extractor 456

Channel-wise Relevance Counting on U-Net architectures (based on the search table 5.1) on 456 (out of 576) models which achieved a higher MIoU score than 0.55.

Method	BASE	MIP	AVG	TMAX	CBV	CBF	MTT	TTD	TTS
DeepLIFT	1782	2775	1180	3855	3654	1402	3539	4325	4221
LRP$\alpha\beta$	3007	4353	2241	4473	1374	194	3249	4730	3185
LRPz	3194	4305	1492	3555	3540	172	2796	4656	3296
Sum	7983	11433	4913	11883	8568	1768	9584	13711	10702

Appendix D

Segmentation Examples

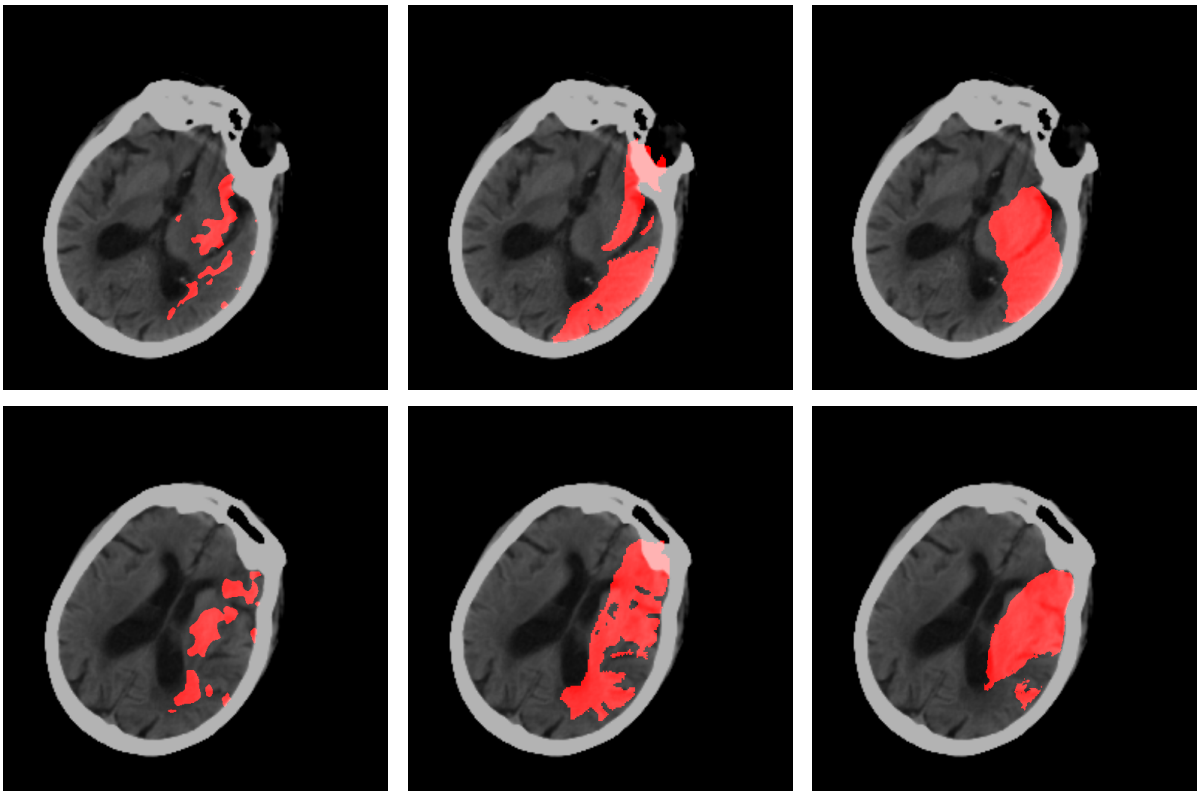

Figure D.1: Center column shows the slices overlayed with their inaccurately registered labels (see section 4.1.2). The left column shows the segmentation of the thresholding approach and the right column the U-Net segmentation (the used network was not trained but validated on this data set). Although the network was trained with wrong labels, it was still able segment properly without exceeding the skull. Since the major part of the training data was well registered, the network was able to identify the skull as non-infarcted tissue. As mentioned in section 4.1.2, in this case the segmentation of the network would be punished as partially incorrect although the network segmented right.

List of Figures

List of Tables

Bibliography

[Aba15] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[Abe10] B Abels, Ernst Klotz, Bernd F Tomandl, Stephan P Kloska, and Michael M Lell. Perfusion ct in acute ischemic stroke: a qualitative and quantitative comparison of deconvolution and maximum slope approach. *American Journal of Neuroradiology*, 31(9):1690–1698, 2010.

[Abu18] S Mazdak Abulnaga and Jonathan Rubin. Ischemic stroke lesion segmentation in ct perfusion scans using pyramid pooling and focal loss. In *International MICCAI Brainlesion Workshop*, pages 352–363. Springer, 2018.

[Alb18] Maximilian Alber, Sebastian Lapuschkin, Philipp Seegerer, Miriam Hägele, Kristof T. Schütt, Grégoire Montavon, Wojciech Samek, Klaus-Robert Müller, Sven Dähne, and Pieter-Jan Kindermans. innvestigate neural networks! *CoRR*, abs/1808.04260, 2018.

[Ard19] Arden Dertat. Applied deep learning - part 4: Convolutional neural networks, 2019. [Online; accessed 21.08.2019].

[Bac15] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[Bad17] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[Bam03] Roland Bammer. Basic principles of diffusion-weighted imaging. *European journal of radiology*, 45(3):169–184, 2003.

[Bam16] Roland Bammer. *MR and CT perfusion and pharmacokinetic imaging: clinical applications and theoretical principles*. Lippincott Williams & Wilkins, 2016.

[Bis06] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[Biv11] Andrew Bivard, Patrick McElduff, Neil Spratt, Christopher Levi, and Mark Parsons. Defining the extent of irreversible brain ischemia using perfusion computed tomography. *Cerebrovascular diseases*, 31(3):238–245, 2011.

[Biv13] Andrew Bivard, Christopher Levi, Neil Spratt, and Mark Parsons. Perfusion ct in acute stroke: a comprehensive analysis of infarct and penumbra. *Radiology*, 267(2):543–550, 2013.

[Blu87] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Occam's razor. *Information processing letters*, 24(6):377–380, 1987.

[Bro10] Eric Brochu, Vlad M Cora, and Nando De Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599*, 2010.

[Cam11] Bruce CV Campbell, Søren Christensen, Christopher R Levi, Patricia M Desmond, Geoffrey A Donnan, Stephen M Davis, and Mark W Parsons. Cerebral blood flow is the optimal ct perfusion parameter for assessing infarct core. *Stroke*, 42(12):3435–3440, 2011.

[Cho15] François Chollet et al. Keras, 2015.

[Cir12] Dan Ciresan, Alessandro Giusti, Luca M Gambardella, and Jürgen Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in neural information processing systems*, pages 2843–2851, 2012.

[Cor95] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[Den14] Li Deng, Dong Yu, et al. Deep learning: methods and applications. *Foundations and Trends® in Signal Processing*, 7(3–4):197–387, 2014.

[dL08] Enrique Marco de Lucas, Elena Sánchez, Agustín Gutiérrez, Andrés González Mandly, Eva Ruiz, Alejandro Fernández Flórez, Javier Izquierdo, Javier Arnáiz, Tatiana Piedra, Natalia Valle, et al. Ct protocol for acute stroke: tips and tricks for general radiologists. *Radiographics*, 28(6):1673–1687, 2008.

[Dod08] Yadolah Dodge. *The concise encyclopedia of statistics*. Springer Science & Business Media, 2008.

[Don08] Geoffrey A Donnan, Marc Fisher, Malcolm Macleod, and Stephen M Davis. Stroke. *The Lancet*, 371(9624):1612 – 1623, 2008.

[Dum16] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.

[ET17] Salwa El Tawil and Keith W Muir. Thrombolysis and thrombectomy for acute ischaemic stroke. *Clinical Medicine*, 17(2):161–165, 2017.

[Fal19] Thorsten Falk, Dominic Mai, Robert Bensch, Özgün Çiçek, Ahmed Abdulkadir, Yassine Marrakchi, Anton Böhm, Jan Deubner, Zoe Jäckel, Katharina Seiwald, et al. U-net: deep learning for cell counting, detection, and morphometry. *Nature methods*, 16(1):67, 2019.

[Faw06] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[Fic55] Adolf Fick. Ueber diffusion. *Annalen der Physik*, 170(1):59–86, 1855.

[Fra18] Peter I Frazier. A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[Gau16] Maxime Gauberti, Sara Martinez De Lizarrondo, and Denis Vivien. The “inflammatory penumbra” in ischemic stroke: From clinical data to experimental evidence. *European stroke journal*, 1(1):20–27, 2016.

[Gon11] R Gilberto González, Joshua A Hirsch, WJ Koroshetz, Michael H Lev, and Pamela W Schaefer. *Acute ischemic stroke*. Springer, 2011.

[Goo16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[Gra06] Robert M Gray et al. Toeplitz and circulant matrices: A review. *Foundations and Trends® in Communications and Information Theory*, 2(3):155–239, 2006.

[Hak19] Arsany Hakim and Roland Wiest. Ct brain perfusion: A clinical perspective. In Alessandro Crimi, Spyridon Bakas, Hugo Kuijf, Farahani Keyvan, Mauricio Reyes, and Theo van Walsum, editors, *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, pages 15–24, Cham, 2019. Springer International Publishing.

[He16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Her16] Francisco Herrera, Francisco Charte, Antonio J Rivera, and María J Del Jesus. Multilabel classification. In *Multilabel Classification*, pages 17–31. Springer, 2016.

[Ho95] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[Hoe04] Ellen G Hoeffner, Ian Case, Rajan Jain, Sachin K Gujar, Gaurang V Shah, John P Deveikis, Ruth C Carlos, B Gregory Thompson, Mark R Harrigan, and Suresh K Mukherji. Cerebral perfusion ct: technique and clinical applications. *Radiology*, 231(3):632–644, 2004.

[Hsi09] Jiang Hsieh et al. Computed tomography: principles, design, artifacts, and recent advances. In *Computed Tomography: Principles, Design, Artifacts, and Recent Advances*. SPIE Bellingham, WA, 2009.

[Iof15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[Jac12] Paul Jaccard. The distribution of the flora in the alpine zone. 1. *New phytologist*, 11(2):37–50, 1912.

[Jar09] Kevin Jarrett, Koray Kavukcuoglu, Yann LeCun, et al. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pages 2146–2153. IEEE, 2009.

[Joh16] Walter Johnson, Oyere Onuma, Mayowa Owolabi, and Sonal Sachdev. Stroke: A global response is needed. *Bulletin of the World Health Organization*, 94:634–634A, 09 2016.

[Jon01] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001. [Online; accessed 14.10.2019].

[Kin14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Kis13] Michael Kistler, Serena Bonaretti, Marcel Pfahrer, Roman Niklaus, and Philippe Büchler. The virtual skeleton database: an open access repository for biomedical research and collaboration. *Journal of medical Internet research*, 15(11):e245, 2013.

[Kra09] Peter G Kranz and John D Eastwood. Does diffusion-weighted imaging represent the ischemic core? an evidence-based systematic review. *American Journal of Neuroradiology*, 30(6):1206–1212, 2009.

[Lan00] Maarten G Lansberg, Gregory W Albers, Christian Beaulieu, and Michael P Marks. Comparison of diffusion-weighted mri and ct in acute stroke. *Neurology*, 54(8):1557–1561, 2000.

[Lap19] Sebastian Lapuschkin. *Opening the machine learning black box with Layer-wise Relevance Propagation*. PhD thesis, TU-Berlin, 2019.

[LeC95] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[LeC98] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[Lin16a] Longting Lin, Andrew Bivard, Venkatesh Krishnamurthy, Christopher R Levi, and Mark W Parsons. Whole-brain ct perfusion to quantify acute ischemic penumbra and core. *Radiology*, 279(3):876–887, 2016.

[Lin16b] Michelle P Lin and David S Liebeskind. Imaging of ischemic stroke. *Continuum: Lifelong Learning in Neurology*, 22(5):1399, 2016.

[Lon15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[Luc18] Christian Lucas, André Kemmling, Nassim Bouteldja, Linda F Aulmann, Amir Madany Mamlouk, and Mattias P Heinrich. Learning to predict ischemic stroke growth on acute ct perfusion data by interpolating low-dimensional shape representations. *Frontiers in neurology*, 9, 2018.

[Mai17] Oskar Maier, Bjoern H Menze, Janina von der Gablentz, Levin Häni, Mattias P Heinrich, Matthias Liebrand, Stefan Winzeck, Abdul Basit, Paul Bentley, Liang Chen, et al. Isles 2015-a public evaluation benchmark for ischemic stroke lesion segmentation from multispectral mri. *Medical image analysis*, 35:250–269, 2017.

[Mai18] Andreas Maier, Stefan Steidl, Vincent Christlein, and Joachim Hornegger. *Medical Imaging Systems: An Introductory Guide*, volume 11111. Springer, 2018.

[Mai19] Andreas K Maier, Christopher Syben, Bernhard Stimpel, Tobias Würfl, Mathis Hoffmann, Frank Schebesch, Weilin Fu, Leonid Mill, Lasse Kling, and Silke Christiansen. Learning with known operators reduces maximum error bounds. *Nature machine intelligence*, 1(8):373–380, 2019.

[Mar19] Mark West. Convolutional neural networks : The theory, 2019. [Online; accessed 21.08.2019].

[McC43] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[Mei54] Paul Meier and Kenneth L Zierler. On the theory of the indicator-dilution method for measurement of blood flow and volume. *Journal of applied physiology*, 6(12):731–744, 1954.

[Mon17] Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222, 2017.

[Moz16] Dariush Mozaffarian, Emelia J Benjamin, Alan S Go, Donna K Arnett, Michael J Blaha, Mary Cushman, Sandeep R Das, Sarah de Ferranti, Jean Pierre Després, Heather J

Fullerton, et al. Heart disease and stroke statistics-2016 update a report from the american heart association. *Circulation*, 133(4):e38–e48, 2016.

[O'S15] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[Pal15] Murugan Palaniswami and Bernard Yan. Mechanical thrombectomy is now the gold standard for acute ischemic stroke: implications for routine clinical practice. *Interventional neurology*, 4(1-2):18–29, 2015.

[Pas17] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[Pea02] Thomas A Pearson, Steven N Blair, Stephen R Daniels, Robert H Eckel, Joan M Fair, Stephen P Fortmann, Barry A Franklin, Larry B Goldstein, Philip Greenland, Scott M Grundy, et al. Aha guidelines for primary prevention of cardiovascular disease and stroke: 2002 update: consensus panel guide to comprehensive risk reduction for adult patients without coronary or other atherosclerotic vascular diseases. *Circulation*, 106(3):388–391, 2002.

[Pha00] Dzung L Pham, Chenyang Xu, and Jerry L Prince. Current methods in medical image segmentation. *Annual review of biomedical engineering*, 2(1):315–337, 2000.

[Psc11] Willibald Pschyrembel. *Klinisches wörterbuch*. Walter de Gruyter, 2011.

[Ram14] Joana N Ramalho and Isabel R Fragata. 17 computed tomography (ct) perfusion: Basic principles and clinical applications. *Vascular Imaging of the Central Nervous System: Physical Principles, Clinical Applications, and Emerging Techniques*, 3:257, 2014.

[Ras13] Maryam Rastgarpour and Jamshid Shanbehzadeh. The problems, applications and growing interest in automatic segmentation of medical images from the year 2000 till 2011. *International Journal of Computer Theory and Engineering*, 5(1):1, 2013.

[Rob18] David Robben, Anna MM Boers, Henk A Marquering, Lucianne LCM Langezaal, Yvo BWEM Roos, Robert J van Oostenbrugge, Wim H van Zwam, Diederik WJ Dippel, Charles BLM Majoie, Aad van der Lugt, et al. Prediction of final infarct volume from native ct perfusion and treatment parameters using deep learning. *arXiv preprint arXiv:1812.02496*, 2018.

[Ron15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[Ros58] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[Rud16] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[Rum88] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

[Rus16] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[San18] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems*, pages 2483–2493, 2018.

[Sch02] Pamela W Schaefer, Javier M Romero, P Ellen Grant, Ona Wu, A Gregory Sorensen, Walter Koroshetz, and R Gilberto González. Diffusion magnetic resonance imaging of acute ischemic stroke. In *Seminars in roentgenology*, volume 37, pages 219–229. Elsevier, 2002.

[Sch08] Pamela W Schaefer, Elizabeth R Barak, Shahmir Kamalian, Leila Rezai Gharai, Lee Schwamm, Ramon Gilberto Gonzalez, and Michael H Lev. Quantitative assessment of core/penumbra mismatch in acute stroke: Ct and mr perfusion imaging are strongly correlated when sufficient brain volume is imaged. *Stroke*, 39(11):2986–2992, 2008.

[Sed12] Philip Sedgwick. Pearson's correlation coefficient. *Bmj*, 345:e4483, 2012.

[Shr17] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3145–3153. JMLR. org, 2017.

[Sno12] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.

[Soa10] Bruno P Soares, Elizabeth Tong, Jason Hom, Su-Chun Cheng, Joerg Bredno, Loic Boussel, Wade S Smith, and Max Wintermark. Reperfusion is a more accurate predictor of follow-up infarct volume than recanalization: a proof of concept using ct in acute ischemic stroke patients. *Stroke*, 41(1):e34–e40, 2010.

[Son18] Tao Song and Ning Huang. Integrated extractor, generator and segmentor for ischemic stroke lesion segmentation. In *International MICCAI Brainlesion Workshop*, pages 310–318. Springer, 2018.

[Spe61] Charles Spearman. The proof and measurement of association between two things. *Jstor*, 1961.

[Sze10] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[Tah15] Abdel Aziz Taha and Allan Hanbury. Metrics for evaluating 3d medical image segmentation: analysis, selection, and tool. *BMC medical imaging*, 15(1):29, 2015.

[The16] Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May 2016.

[Tor13] Michel T Torbey and Magdy H Selim. *The Stroke Book*. Cambridge University Press, 2013.

[Win06] Max Wintermark, Adam E Flanders, Birgitta Velthuis, Reto Meuli, Maarten Van Leeuwen, Dorit Goldsher, Carissa Pineda, Joaquin Serena, Irene van der Schaaf, Annet Waaijer, et al. Perfusion-ct assessment of infarct core and penumbra: receiver operating characteristic curve analysis in 130 patients suspected of acute hemispheric stroke. *Stroke*, 37(4):979–985, 2006.

[Win07] Max Wintermark, R Meuli, P Browaeys, M Reichhart, J Bogousslavsky, P Schnyder, and P Michel. Comparison of ct perfusion and angiography and mri in selecting stroke patients for acute treatment. *Neurology*, 68(9):694–697, 2007.

[Wit12] Rachel Wittenauer and Lily Smith. Background paper 6.6 ischaemic and haemorrhagic stroke. *Priority Medicines for Europe and the WorldÄ Public Health Approach to Innovation*, 2012.