University of Bergen
Department of Informatics

Automatic Detection of the Arterial Input Function in DCE-MRI

Author: Sven Alrik Solemdal
Supervisors: Pekka Parviainen and Ole Gunnar Johansen

UNIVERSITETET I BERGEN
Det matematisk-naturvitenskapelige fakultet

June, 2023

Abstract

Purpose: In modern DCE-MRI image analysis, the AIF is essential in getting correct pharmacokinetic values. The AIF can be extracted semi-automatically or manually. Either way, the method is subject to poor reproducibility and suffers from operator bias. This study aims to create a pipeline for the automatic extraction of the arterial input function used in cerebral DCE-MRI in a repeatable and reproducible way.

Methods: Dynamic contrast-enhanced T1-weighted MRI images of 59 patients were used to train a DenseNet model to identify slices with a visible middle cerebral artery. Two methods are proposed to select one or more axial slices within a DCE-MRI series that is likely to contain the MCA. These slices are subsequently used to estimate voxels that are likely to represent the AIF using an established approach [1]. Method A used the single slice, which the model gave the highest probability of containing the MCA. Method B uses all slices where the model had assigned a probability higher than 0.5. These methods were compared against baseline method C, using all slices. The intra-variability of 5 patients imaged over a short period (weeks) was measured by considering differences in AUC, TTP, PTB, and signal correlation to evaluate the different model performances. Finally, a qualitative examination of method B's best and worst results was made.

Results The MCA detection model tested on 831 samples of unseen heterogeneous data achieved an accuracy of 94.9%, a precision of 87.1%, and a recall of 96.6%. Comparing the different methods, method A performed the worst with mean differences and standard deviations for AUC -10.51 (185.82), TTP 0.63 (2.94), PTB -0.39 (5.13), and a mean correlation between the series of 0.90. Method B performed the next best with mean differences and standard deviations for AUC 28.74 (128.04), TTP 0.70 (2.32), PTB 0.69 (3.96), and a mean correlation between the series of 0.92. The best model was model C, achieving the mean differences and standard deviations for AUC 5.09 (54.70), TTP 0.59 (3.41), PTB 0.06 (1.74), and a mean correlation between the series of 0.91. AIFs extracted with methods A and B had an average TTP 2.67 seconds earlier than method C.

Conclusion Through this thesis, a modular and easily adaptable pipeline for extracting AIFs has been defined. Method C yielded the lowest average variance. Without some outliers, method B shows promising results and should be worth developing further.

Acknowledgements

First of all, thanks to Pekka for agreeing to supervise the thesis of a student suddenly walking into his office. This has been a year with a single, long, learning curve. Having your guidance has been essential for the thesis. While also working together in a course you lecture, you have provided good humor and always answer even the vaguest questions to the best of your ability.

Next, I would like to thank Ole Gunnar for providing this hard but exciting task. You have an enthusiasm for the field, which is infectious. I thank you for your critical thinking and good conversations throughout the time we have worked together.

I would also like to thank Atle Bjørnerud for helping out with the not-so-small problem of having good medical data to work with. You have provided insightful answers to sporadic emails throughout the year, often calming an acute state of crisis.

To the Bergen office of NordicNeuroLab, I would like to show my depthless gratitude for having me in your offices and caring for me like I were one of your own. I thank you for showing interest in the progress of my thesis, insightful conversations, and several one-way "rubber-duck" monologues. Your great working environment, with events, competitions, food, and fresh coffee, has been the largest motivating factor of them all.

I would like to thank my friends and family for creating a healthy work-life balance. An especially large thanks to my fellow students that I have shared my griefs and joys with through these last five years.

Lastly, I would thank Tyra for always listening to my "man-splaining" and encouraging me through the work. You have seen the ups and downs and always helped shine a light on the positive things. A long day of work is always worth it when I come home to you.

`Sven Alrik Solemdal`

Thursday 1st June, 2023

Contents

List of Figures

List of Tables

Introduction

Dynamic contrast-enhanced magnetic resonance imaging (DCE-MRI) is an imaging technique able to measure tissue perfusion and disruptions in the blood-brain barrier(BBB) in the central nervous system(CNS)[10, 11].

While commonly used for renal and prostate diagnostics, DCE-MRI has not seen a wide clinical adaptation in routine cerebral imaging[12]. To increase trust in DCE-MRI analysis, the Quantitative Imaging Biomarkers Alliance (QIBA) calls for repeatability and reproducibility of DCE-MRI studies[13].

When using DCE-MRI images with mathematical models to estimate the leakage of a contrast agent (CA) to the extracellular extravascular space (EEE), an arterial input function (AIF) describing the intravascular CA concentration is needed[14, 15]. The AIF is traditionally manually extracted from a region of interest(ROI) thought to accurately describe the vascular input of contrast agent to the eloquent tissue, or by using a statistically averaged function from a patient population[16].

A population-averaged function is inherently reproducible and repeatable. On the other hand, since it is not patient-specific, important information might be missed from the analysis. Manually extracted AIF is patient-specific but inherently prone to operator bias. Different operators often select significantly different AIFs[17]. Modeling the leakage of CA is inherently sensitive to variations in the AIF[12, 18, 19]. Therefore, a patient-specific repeatable and reproducible AIF selection pipeline is a problem that must be solved to enable repeatable and reproducible DCE-MRI studies in the future.

Several semi- and fully-automatic AIF-extraction methods have been proposed in recent years [20, 1, 21, 22, 23, 24, 25, 26]. Some semi-automatic methods rely on the user selecting a slice or ROI to be searched in [20, 1, 21, 22]. Other methods use a deterministic

approach to segment out a vascular region and then extract a vascular input function from this region [24]. More recently, deep learning has been used to extract an AIF directly from a full brain volume [23, 27, 26].

Semiautomatic models need manual input, thus they are prone to operator bias reducing repeatability and reproducibility. The deep-learning approaches [23, 27, 26] suffer from the trained models being closed-source, which makes them impossible to test. Most of the models also suffer from the black box problem, meaning that it is impossible to derive how they arrived at a prediction.

1.1 Thesis motivation

This thesis aims to investigate whether a reliable, fully automatic AIF extraction pipeline can be made using machine learning. Second, when comparing DCE-MRI sequences taken over long periods, the pipeline should extract AIFs with low intra-patient variance. The main goals of this thesis are, therefore:

1. Define and train a deep-learning model that reliably detects when the MCA segment is visible in a slice.
2. Implement an image-processing pipeline that uses said deep-learning model and k-means to produce an AIF from a DCE-MRI image sequence.
3. Evaluate the intra-patient variance of AIFs by applying the method to patients imaged several times over a short period.

1.2 Thesis structure

The thesis is composed of 4 parts. First, the theory needed to describe the thesis problem and method is given in chapter 2.
Next, chapter 3 describes how the MCA detection model is made and how it is used in an AIF extraction pipeline. Three variations of this pipeline are described, as well as how the model and the pipeline variations are to be evaluated.
The results acquired by applying the three variations of the pipeline on a dataset are provided in chapter 4. Here, the results are described, and some notable observations are

highlighted.
As the last chapter, chapter 5 contains reflections on the method and the results produced by applying it. The limitations of the thesis are discussed, as well as advice for further work, and a conclusion.

Background

This chapter contains two parts. The first part unfolds into three segments: (i) an introduction to MRI, (ii) how using a contrast agent and dynamic imaging can be used to model perfusion characteristics from DCE-MRI, (iii) an in-depth explanation of the AIF, as it is at the core of the thesis problem.
The second part unfolds into four segments: (i) a brief introduction to machine learning (ML), how bias and variance impact model performance, and a description of the k-means algorithm that will be used in the thesis, (ii) artificial neural networks and how they work, (iii) convolutional neural networks and a description of different parts often found in modern convolutional networks, (iv) a description of the DenseNet architecture that is used in the thesis.

2.1 Principles in Magnetic Resonance Imaging

Magnetic Resonance Imaging is based on the fact that nuclei with a spin angular momentum(spin) can interact with a magnetic field [28, 29]. An initial belief that nuclides possessed a physical spin, like a gyroscope, had been refuted by discoveries indicating that the spin of nuclei could only exist in discrete, quantized levels. Spin should rather be seen as an intrinsic value, just as mass and charge [30]. Although particle spin is a quantum mechanical property, useful analogies can be made using classical mechanics for understanding how the spin interacts with a magnetic field.

When particles with spin experience a magnetic field b_0, the particles align their axis of rotation along the direction of the magnetic field. If these particles are disturbed by another magnetic field for a short moment, their axis of rotation falls out of equilibrium and starts to precess around the direction of the main magnetic field [2].

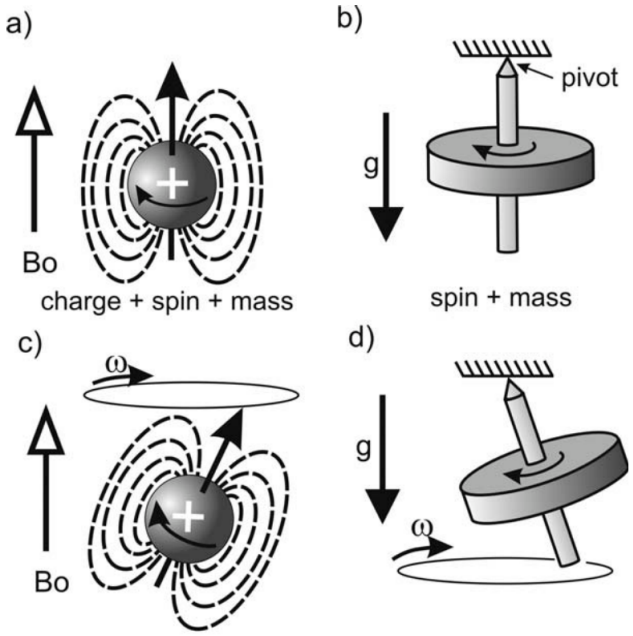

Figure 2.1: A simplified figure showing how a proton tipped out of equilibrium will precess around an external magnetic field. (**a**) The spin and charge of a proton cause it to generate its own magnetic field and align its rotational axis along the direction of the external magnetic field B_0. (**b**) This can be imagined as a gyroscope hanging from a pivot, aligning itself to the earth's magnetic field. (**c**) When a disturbance is introduced, the proton gets out of equilibrium and starts to precess around the direction of B_0 with the angular momentum ω. (**d**) The same applies to the gyroscope when it is disturbed. Courtesy of Plewes et al. [2]

The angular frequency of the precession, ω, denoted as the Larmor frequency, is determined by the strength of the applied magnetic field and the characteristics of the nuclei. By combining all of the characteristics into a single number, the gyromagnetic ratio γ, the relationship between the Larmor frequency and the magnetic field can be expressed in simple terms. The resulting relationship, the Larmor equation, yields:

$$\omega = \gamma B_0 \tag{2.1}$$

To acquire a signal from a targeted nuclei, an alternating magnetic field (RF-pulse) is applied with a frequency equal to the Larmor frequency of the targeted nucleus to tip it out of equilibrium [2]. This, in turn, creates a detectable signal that can be used in various ways, as imaging.

2.1.1 T1 and T2 relaxation

When the magnetic vector of many nuclei is tipped out of equilibrium at the same time, they produce a magnetic vector in the transverse plane. Immediately after tipping, all magnetic vectors are in phase and the total magnetic vector can be detected by an external coil. Not long after being tipped, the vectors will lose coherence and the magnetic vector in the transverse plane gradually disappears. This is called T2-relaxation.
When the nuclei are tipped, the magnetic vector along the longitudinal plane diminishes. But, as the nuclei will try to re-align themselves with the main magnetic field, the longitudinal magnetic vector will gradually return back to its equilibrium state. This recovery is called T1-relaxation.

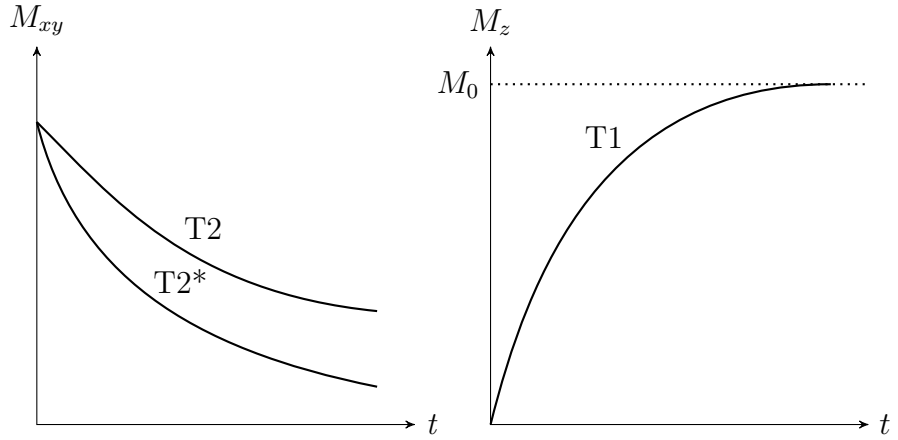

Figure 2.2: T1 and T2 relaxation processes following an RF-pulse. Note that the observed T2 decay (T2*) happens faster than the theoretical decay T2.

The time at which the T1 signal recovers 63% of its equilibrium state is called the T1 time constant. The T2 time constant is defined as the time needed to reduce the T2 signal to 37% of its maximum value [2].
Due to the different T1 and T2 time constants in tissue, it is possible to capture a contrast (differentiate) between different types of tissues in the body [31]. With special sequences of inducing an RF sequence and then reading the signal, an MRI machine can be configured to focus on differences in T1 or T2 relaxation rates. When the MRI machine, for example, is configured to focus on T1-relaxation signals, the resulting image is called "*T1-weighted*".

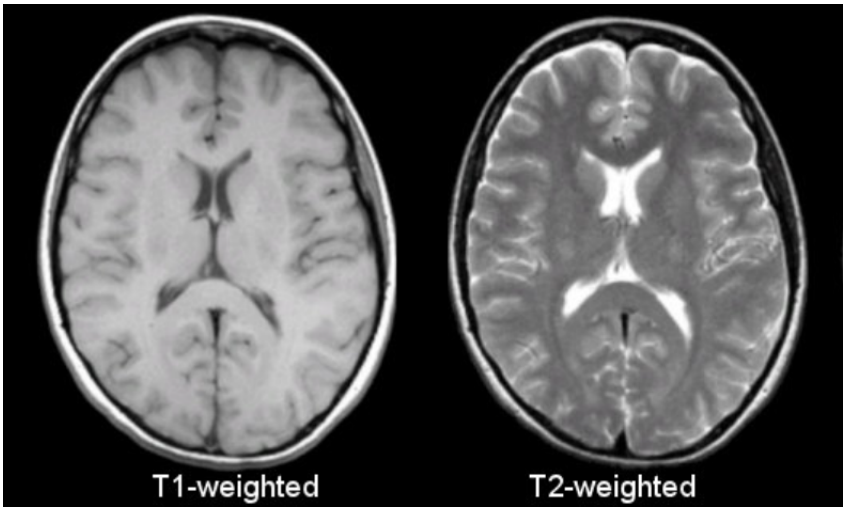

Figure 2.3: A side by side comparison of T1- and T2-weighted images of the brain. Notice how the central spaces in the brain appear dark in the T1-weighted image, while it has a stronger signal and appears brighter in the T2-weighted image. Source: Website [3]

2.1.2 MRI resolution

As seen in Figure 2.3, the MRI machine can produce detailed brain images. It shows two axial *slices* of a human brain, that is, an image from the xy-plane along the z-axis which runs along the length of the body. Although the slice looks like a two-dimensional image, it is actually a three-dimensional image with a *slice thickness*. When the MRI machine is imaging a slice, it images with a depth along the z-axis. This means that each pixel in an MRI image is actually a cube and thus referenced as a *voxel*. To view the image in 2D, the 3D voxel is transformed into a pixel.

In diagnostic brain imaging, the slice thickness is usually in the millimeter scale [32]. Imaging with a low thickness and small voxel size will give a detailed 3D image of the brain [32]. But, with a small voxel size more time is needed to acquire enough signal for each voxel to get a useable contrast. In dynamic imaging where the temporal resolution is important, the voxel size has to be large enough so that each voxel gets enough signal for each time point.

To image a subject's whole brain, several slices are imaged. The final collection of slices is called a volume. When the whole brain is imaged several times, like a movie, a collection of volumes is imaged. Each volume corresponds to a time point; the whole collection is called a time series or, usually, only a series.

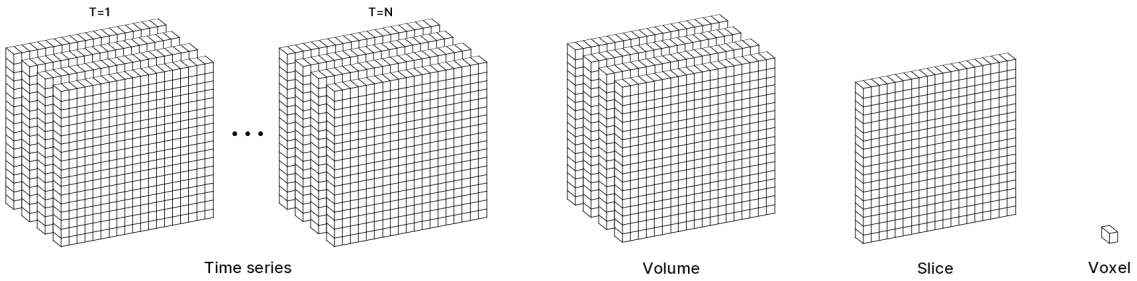

Figure 2.4: A figure showing a time series of volumes, a volume, a slice, and a voxel.

2.1.3 Contrast agent

For T1-weighted MRI images, the difference between the T1 relaxation times of tissues can sometimes be too small to capture. Thus, special kinds of contrast agents(CAs) have been developed to enhance the differences in T1-relaxation rates in the tissue. A common type of such an agent today is a Gadolinium(Gd^{+3}) based contrast agent [33].
Gadolinium is a paramagnetic transition metal. Since the magnetic moment of an electron is about 700 times higher than a proton, paramagnetic ions induce large fluctuating magnetic fields. If these fluctuations are close to the Larmor frequency of the surrounding protons, a notable enhancement of the proton relaxation will occur [34]. In turn, this relaxation effect produces a change in T1-relaxation around it, making it a visible contrast in T1-weighted imaging. When injected into a patient's vascular system, it creates a possibility to measure the characteristics of the vascular dynamics and vascular structure of the patient.
If abnormal changes happen to the vascular tissue that disrupt the blood-brain barrier(BBB), the CA can diffuse into the extravascular extracellular space (EES). In turn, this highlights the area with the abnormality and indicates the appearance of pathology and its characteristics. Such changes in the vascular structure are one of the hallmark indications of cancer and make DCE-MRI especially suitable for cancer detection[35].

2.2 Dynamic Contrast-Enhanced MRI

It is not enough to only image a single timepoint of the brain to quantify the leakage through the BBB into the EES. Quantifications of important biomarkers (quantitative imaging) rely on information about the dynamic behavior of the blood. Developments in MRI techniques have made it possible to acquire such dynamic images. In a sense, this

means "filming" the imaged area.
Today there are several ways to acquire dynamic MRI. Clinically, some important techniques are arterial spin labeling (ASL) [36], dynamic susceptibility-enhanced imaging (DSC) [37], and dynamic contrast-enhanced imaging (DCE) [11]. All differ in what kind of biomarkers they can produce, imaging sequences needed, and signal-to-noise ratios.
DCE-MRI exploits the T1-enhancing properties of Gd-based contrast agents [11]. Analyzing how the raw signal changes when the CA enters the tissue (non-parametric DCE-MRI analysis) has shown to be helpful in several clinical applications [38, 39]. Yet, this kind of analysis does not directly quantify the biological processes in the tissue.

2.2.1 Pharmacokinetic modeling

Pharmacokinetic modeling aims to extract measurements of how the CA diffuses through the BBB into the EES. For this, several models have been proposed through the years. Two general approaches exist today; the compartmental and spatially distributed models [39]. The compartmental approach is seen as the less complex one and is far more often seen in common medical applications [39]. This is also the range of methods relevant to this thesis.
A compartmental model assumes two or more compartments that specify the CA concentrations in the EES and the blood plasma [10]. First, a CA concentration is delivered through the artery inlet and gives a dynamic concentration in the first compartment. This again results in an exchange of CA concentration with the other compartment(s). The rate at which the CA is exchanged between these compartments is related to the capillary permeability, thus giving a related measurement [10]. An important measurement of the permeability is the volume transfer rate of CA into the EES, namely K_{trans}.

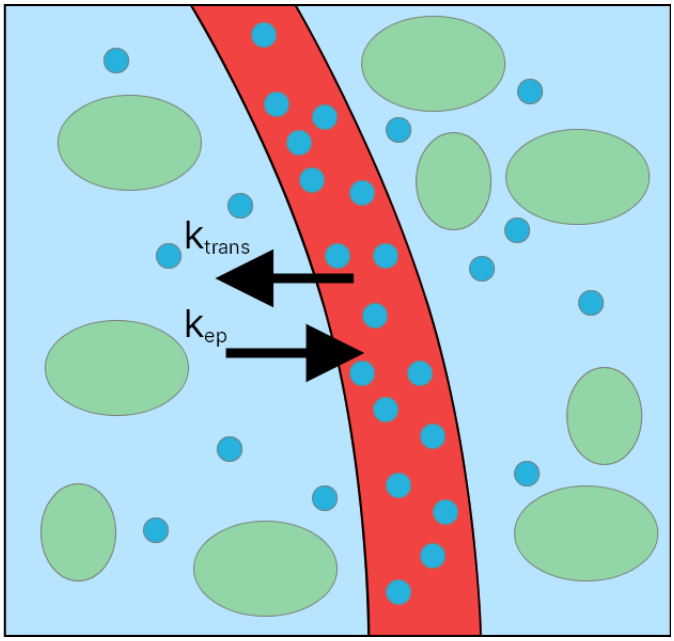

Figure 2.5: Schematic illustration of the TK model parameters. K_{trans} is the rate of CA (blue circles) diffusing from the vascular plasma (red area) into the EES (blue area). k_{ep} is the value of diffusion back into the plasma. The green circles are cells.

Some common models producing a K_{trans} measurement are the Tofts-Kermode(TK) and the extended TK model [15]. The two-compartment exchange model (2CXM) does not produce k_{trans} measurements but models similar kinetical parameters [14].
Common for all models is the need for a function describing the vascular input of CA. Such a function can be derived from an area thought to sufficiently resemble the vascular input of the tissue of interest, or by using an averaged function derived from a patient population [16]. Normally it is wanted for the function to be extracted from an artery, namely an arterial input function(AIF) [4]. Several papers have made it clear that the derived parameters from each model are sensitive to changes in this function [18, 12, 19]. Thus, the AIF is an essential part of DCE-MRI analysis.

2.3 Arterial Input Function

Although hard to detect automatically, the AIF curve often displays recognizable characteristics. The general shape of an AIF curve starts with a *baseline*, which is the signal before the bolus (CA) has arrived in the voxel. When the bolus arrives in the voxel, a sharp rise in the signal can be seen, which then rapidly decreases again. This is the *first passage* of the CA. After the decrease, the signal does not immediately decrease to the

previous baseline levels but slowly decreases as the CA diffuses out of the tissue. Usually, a new peak is seen sometime after the *first passage*, although wider and with a lower amplitude. This is often denoted as the *recirculation*, as it was generally assumed to be the recirculation of CA in the heart-lung system. It is shown not to be from actual recirculation but from parts of the original bolus distributed to other body organs [4]. After recirculation, the contrast curve decreases gradually. Figure 2.6 displays an AIF curve with the typical characteristics.

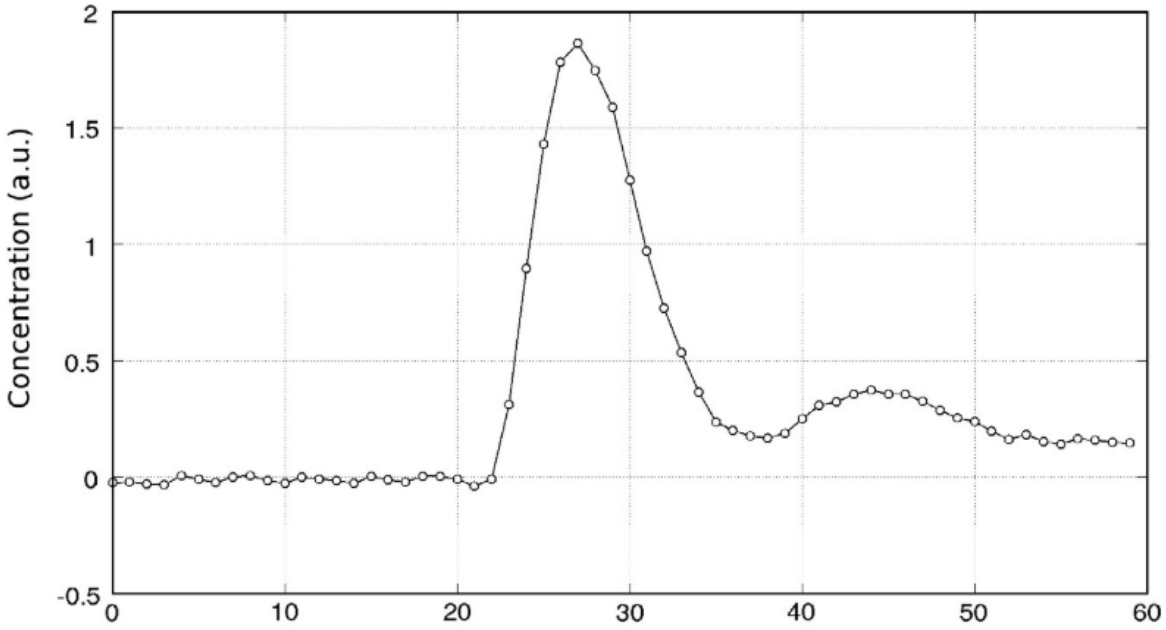

Figure 2.6: A general AIF curve where the baseline, first passage, recirculation, and gradual decrease can be seen. Courtesy of Calamante et. al.(2013) [4]

To acquire an AIF, a measurement can be done directly from a blood vessel supplying the tissue of interest. Although giving a true AIF, in all common medical applications, this is too invasive and time-consuming to do [39].
A less invasive and more common way to measure the AIF is to use the T1 signal from voxels within a certain blood vessel as an AIF. This is called an *individual AIF*. It is convenient because it uses the same signal data as for the DCE-MRI analysis. The actual AIF acquisition happens in post-processing of the DCE-MRI images. To select where the AIF is extracted from, a trained person needs to select the voxels of a feeding vessel from which the AIF will be extracted. A region of interest (ROI) can be determined when using semi-automatic algorithms, and an algorithm will select the most suitable AIFs from this area [1, 21]. There are also fully automatic ways to select an AIF, but these carry their own limitations which will be discussed in subsection 2.3.2.

A population-averaged AIF may be used in many applications. These consist of measured AIFs from a population, which are then fitted to a mixed Gaussian formula [16]. This general AIF can then be used for DCE-MRI analysis of different patients. This way, no time has to be used by a clinician to select an ROI, and one does not need to worry about errors due to low image SNR or other anomalies. Although very efficient, it is not preferred over an individual AIF if the prerequisites for it are right. A population-averaged AIF does not contain patient-specific information that might lead to misleading results. Individual AIF extraction is the most used method today in cerebral DCE-MRI. Several techniques have been developed and applied through the years, carrying different advantages and disadvantages.

2.3.1 Location of the AIF

Ideally, since the vascular structure in different brain regions has unique flow dynamics, the AIF curves from all arterial voxels should be extracted and used in PK modeling[4]. This is called *local AIF* extraction and extracts different AIFs for different brain regions, dependent on their supplying vessel. Yet, techniques using this method are still in development and have not seen wide clinical use [4].
More commonly, a *global AIF* is used, where a single AIF is extracted to be used for the whole volume. Due to the mentioned unique vascular structures connected to the tissue of interest, extracting the AIF from these structures is preferred. However, these vessels can be very small, even smaller than a voxel, which makes extractions from voxels containing these vessels prone to partial volume effects (PVEs) [4]. In short, PVEs are errors originating from measuring an AIF from a voxel containing both tissue and vessels. The signal curve will not be representative of the actual AIF. PVEs can lead to considerable errors in the measurement of the shape of the AIFs, which in turn will propagate to the PK modeling. A solution is to measure the AIF from larger arteries less prone to PVEs. Usually, being further away from the tissue of interest may yield an erroneous representation of the actual true AIF. Thus, some kind of compromise has to be made between these two problems.
In cerebral imaging, a common choice is to extract the AIF from one of the segments of the middle cerebral artery (MCA) or from the internal carotid artery (ICA). More recently, surrogate AIF has been proposed to be extracted from the superior sagittal sinus (SSS), but this has yet to be applied in common medical practice [40]. The SSS is a vein, meaning it will have a bolus arrival delay. This needs to be compensated for to be valid in PK modeling.

Often, the MCA is selected to extract a global AIF [4]. It is a large artery following the ICA in delivering blood to the whole brain. While it consists of several segments, the M1 segment is the thickest and is the most popular extraction target.

To extract an AIF from the MCA, a trained radiologist must identify the MCA within a volume, and select the voxels assumed to represent the AIF most accurately. This is a time-consuming task and is subject to the radiologist's bias. To avoid this issue, there have been several efforts to make algorithms that select voxels to represent the AIF automatically. Some of them are discussed in the next section.

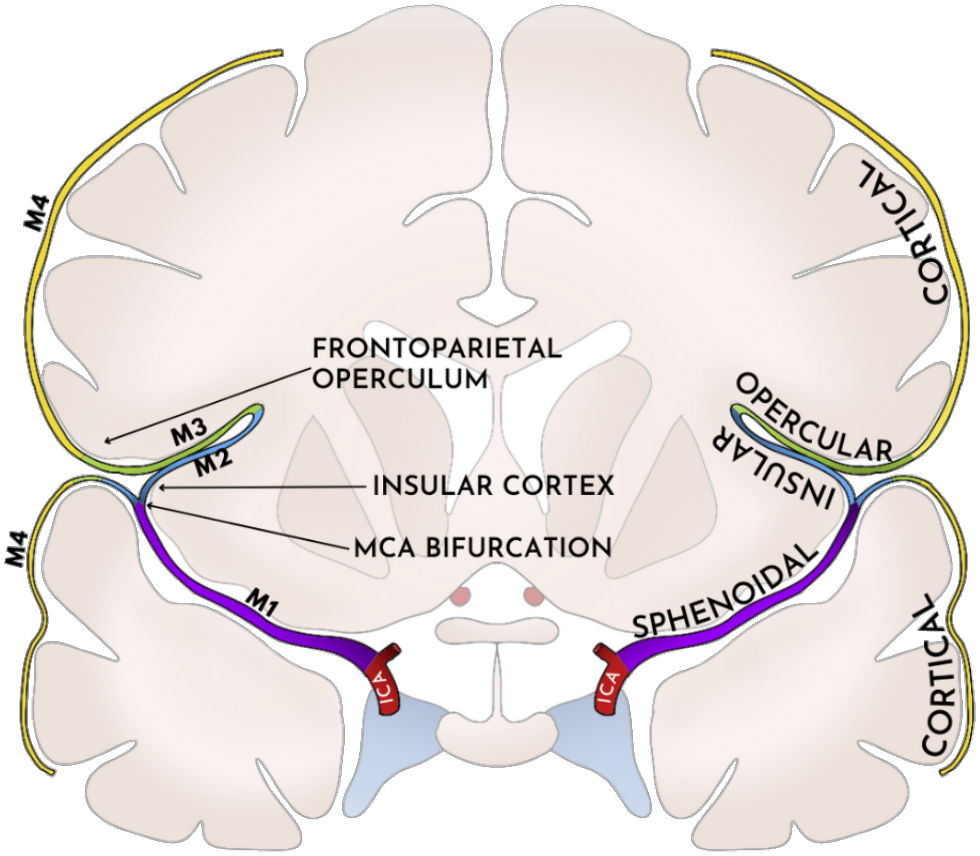

Figure 2.7: A map of the MCA segments and some of the ICA. Note that this is a generalized image of how these structures should look like. Many variations can be seen in the general population.

2.3.2 Semi and fully automatic AIF extraction methods

To extract an AIF automatically, methods have been developed to search through a whole volume (automatically) or a predefined slice (semi-automatically) for a suitable AIF. For the semi-automatical approach, an operator selects a slice or an ROI, and a clustering

technique is then used to select the most suitable group of voxels. Several clustering techniques exist [21, 20, 22], with the most common one being a k-means approach [1]. Theoretically, such an approach can also be applied to a whole volume, without an operator needing to select a slice or ROI.
Some efforts have been made to directly infer parametric maps without finding an AIF or using mathematical models. These are yet to be adopted for medical industry use. One drawback of such models is that they suffer from a black box problem; It is not possible to understand how the models arrived at their results.
By using a deep neural network, Winder et al. proposed a binary model to determine a mask differentiation between arterial and non-arterial voxels in computer tomography perfusion (CTP) and MRI [27]. The AIF is then computed by geometrically averaging the most probable arterial voxels.
Another deep learning network, AIFnet, was proposed by de la Rosa et al. [26]. It computes a 3D probabilistic volume representing the voxelwise contribution to the AIF. Having a time series as input, it can be made to estimate both the AIF and the venous output function (VOF). Compared against the selection from one operator the method achieved a mean correlation of 0.965 with a standard deviation of 0.05 for AIF curves, and a mean correlation of 0.981 with a standard deviation of 0.07 for VOF curves.
Lastly, a method tested on patients with colorectal cancer was recently developed by Tönnes et al. [25]. By assuming the shape of an artery, filtering, blob detection, and a gamma variate function are used to calculate the two largest arterial objects in a 3D volume.

2.3.3 The problem with current AIF extraction methods

As stated by the QIBA, there is a desire for repeatable and reproducible DCE-MRI measurements [13]. Computing parametric maps through PK models mentioned in subsection 2.2.1 is dependent on the selected AIF [12, 18, 19]. Contra-intuitively, viewing manual AIF selections as the best standard is common practice, even though these extractions are prone to human bias as experience, training, and local standard operating procedures. Often, low intraclass correlation can be seen between the AIF selections between two operators [17]. Thus, automatic and repeatable AIF extraction methods are sought.
For a method to be repeatable, it should produce the same result for a patient that has been imaged over a short time, without changes in the imaging procedure or large changes in the patient vascular structure. Additionally, for a method to be reproducible, it should

produce the same results independent of the kind of DCE-MRI imaging procedure used. Having to select a slice manually and then apply a clustering algorithm to extract an AIF is prone to operator bias, and can not be seen as reproducible.
Using a deep learning model such as the proposals by Winder and de la Rosa et al. might yield good results, but several factors may hinder complete reproducibility. Training the models requires large amounts of high-quality, labeled, patient data. This takes time to acquire, and privacy concerns might be raised if the model is to be distributed, hindering reproducibility. Also, the models might not be fitted to the kinds of data seen in the organization where it is used, therefore potentially reaching suboptimal results that can not be used in a clinical setting. Further, both models rely on the manual annotation of suitable AIFs, making the models inherit the operator bias from the annotations.
A method proposed by Tönnes et al. [25] relies on assumptions stated for abdominal arteries and might not apply to cerebral AIF extractions.
Population-averaged extractions are initially a good alternative. But, as previously stated, population-averaged AIFs might risk missing important patient-specific information. If absolute parametric values are important, population-averaged AIF should not be preferred [41].
To make a model that does not rely on manual AIF annotations, is reproducible, and easy to understand, this thesis will simulate the traditional manual AIF extraction method by making the slice selection automatic. This way, the model's decision process is transparent, and different clustering methods can be applied to the selected slice.

2.4 Machine Learning

Conversely to classical programming, where the solution to a task has to be explicitly coded, machine learning train on data and recognizes patterns that can be exploited to analyze unseen data. As a subdomain of artificial intelligence (AI), machine learning (ML) constitutes methods that aim to automate intellectual tasks. This is done by using supervised models, where there is a given truth to the model, unsupervised models, where there is no given truth, semi-supervised, where some of the truth is given, and reinforcement learning, where the truth is learned by interacting with an environment [5].

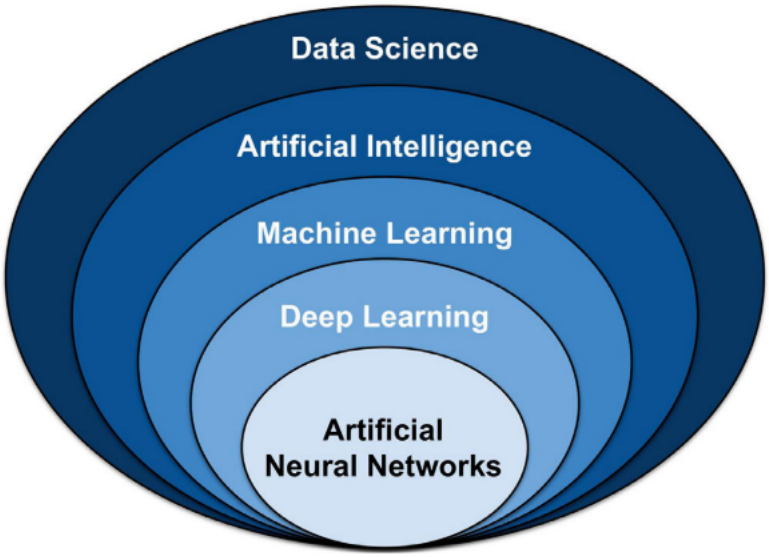

Figure 2.8: Machine learning is a subdomain of artificial intelligence that aims to automate intellectual tasks. Source: Choi et. al.(2020) [5]

In recent times, ML has seen great success in tasks like classification, regression, and clustering. An ML technique typically requires less labor to be functional than other conventional approaches. The merit of ML has been highlighted in many fields of science, such as robotics, computer science, and medicine, due to its universal ability to solve classification- and regression problems [42]. While often being portrayed as a magic solution to every thinkable problem, it has more nuances. Thus it is essential to highlight the basic concepts of ML.

2.4.1 Machine Learning fundamentals

When a problem is to be solved by ML, it is first important to decide the nature of the problem. Normally it belongs to one of three pillars:

- Supervised learning - *The model learns by understanding the connection between features and a given set of labels.*
- Unsupervised learning - *The model learns patterns for a set of features without having any given labels.*
- Reinforcement learning - *The model learns iteratively by receiving feedback from its output.*

Each of these methods requires different models and performance measurements. In this thesis, only supervised learning and unsupervised learning will be used.

2.4.2 Bias and variance trade-off

To make a supervised ML model perform a prediction task, it needs to learn its parameters from a dataset. When the model is made to predict new data points, it will not always produce the correct answer. The error can be decomposed into bias and variance errors. To illustrate them, consider the problem of predicting house prices. If a linear model is applied, and the problem is, in fact, non-linear, it will produce a high bias error. Yet, it has a low variance error as it predicts in linear terms. If a non-linear model is applied to the problem and is trained on previous price data, it might overinterpret random fluctuations in the data and produce the same non-valid fluctuations in its outputs. This time the model suffers from a high variance error.
To mitigate a high variance error, the model can be changed by increasing the model bias. A high bias error can, on the other hand, be mitigated by increasing the variance. The conflict of trying to minimize these sources of errors is known as the *bias-variance tradeoff* and is a central problem in supervised ML.

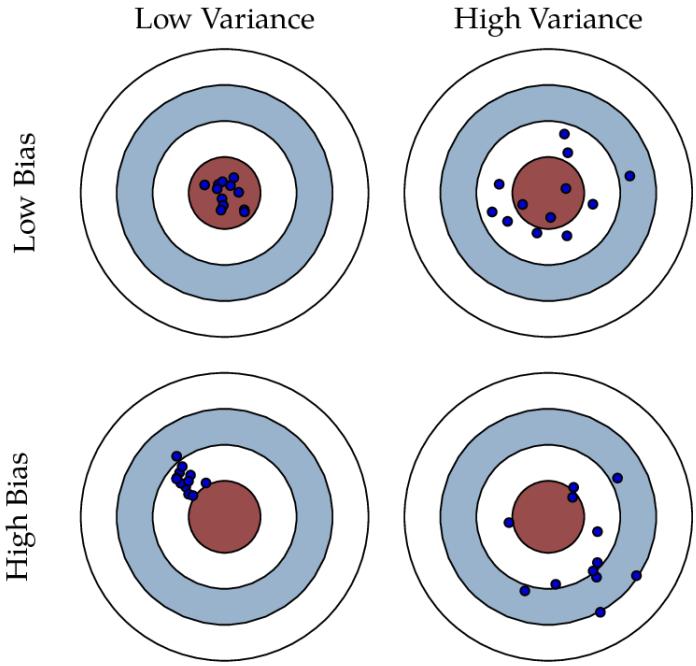

Figure 2.9: The bias-variance tradeoff can be visualized as trying to hit a target. If the bias is high and the variance low, the aim is good but consistently misses the mark. If the bias is low but the variance high, it might hit the mark, but also everything around it.

A model with a high bias but a low variance is said to suffer from *underfitting*. If the

model has a low bias but a high variance, it suffers from *overfitting*.
It is paramount to measure model performance to get a good indication of the bias and variance errors. This is done by analyzing the convergence of training and validation losses and other relevant metrics for the task. The model must be evaluated on completely unseen data to get a true estimate of model bias and variance. This is done by splitting the original dataset into three parts: a training, validation, and testing set. The trainset contains the data the model learns from. This can be modified in different ways to increase a model's ability to generalize to new data points. The validation dataset is used to evaluate and choose between different model configurations. This dataset should not be modified. Ultimately, the test dataset is used to evaluate the final model. Theoretically, this should give the closest estimate of the model performance when in production. As the validation dataset, the test dataset should also not be modified.

2.4.3 K-means

A popular clustering technique is k-means clustering. The algorithm tries to divide a set of data points into k clusters so that the clusters have minimal intra-variance. For each group, there is an average point, called a *centroid*. In simple terms, the algorithm works like this:

1. Select random k centroids
2. Assign each point in the dataset to its nearest centroid
3. Create a new centroid within each cluster that is the mean of the cluster data points.
4. Calculate an objective function
    4.1. If the objective function has converged or is beneath a certain threshold, end the training.
    4.2. If no convergence, repeat steps 2-4

The objective function is formulated following [43], which yields:

$$G_{k-means} = \min_{\mu_1,\cdots\mu_k} \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \mu_i)^2 \tag{2.2}$$

Where μ_i is the centroid of cluster C_i, and $d(x, \mu_i)^2$ is the squared distance between a data point and the centroid in its cluster.
This objective function is tough to converge in many cases, and the objective function is commonly made to be the difference between the last and the new centroid [43]. The algorithm stops if the difference is below a certain threshold.

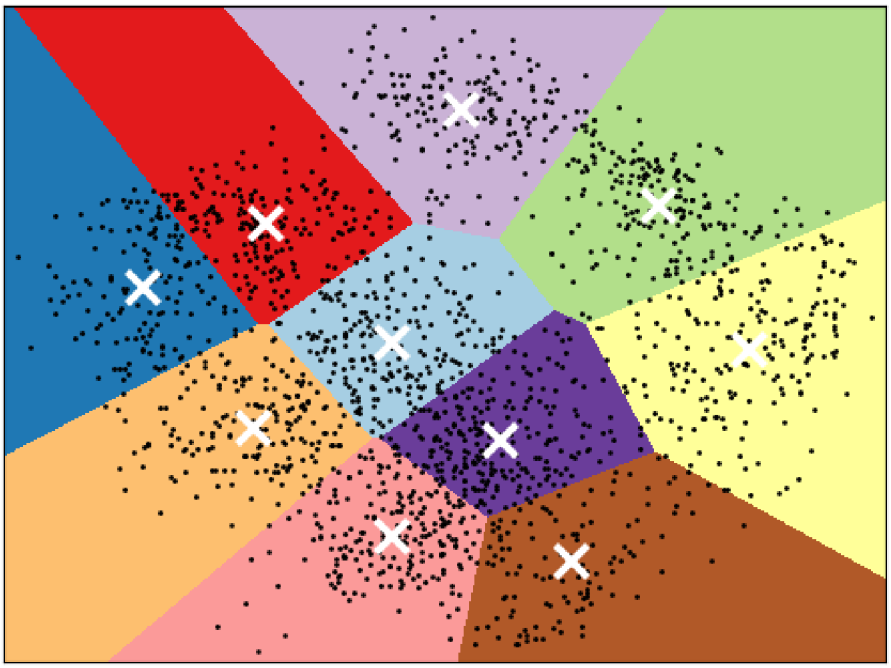

Figure 2.10: k-means divides a set of data points into a set of k clusters. In this case, k was set to ten. The centroids of each cluster are marked with a white cross. Courtesy of the scikit-learn website (2023) [6]

k-means is unsupervised; hence the data points are not guaranteed to be grouped optimally.

2.5 Artificial Neural Networks

In recent times, ANNs have proven outstanding merit in machine learning classification- and regression problems [7]. They mimic biological neurons and their ability to capture and learn patterns from sensory inputs. The most elementary component in an ANN is the perceptron [5]. It takes in a number of features, weighs them, sums them together, and passes them through an activation function as presented in Figure 2.11.

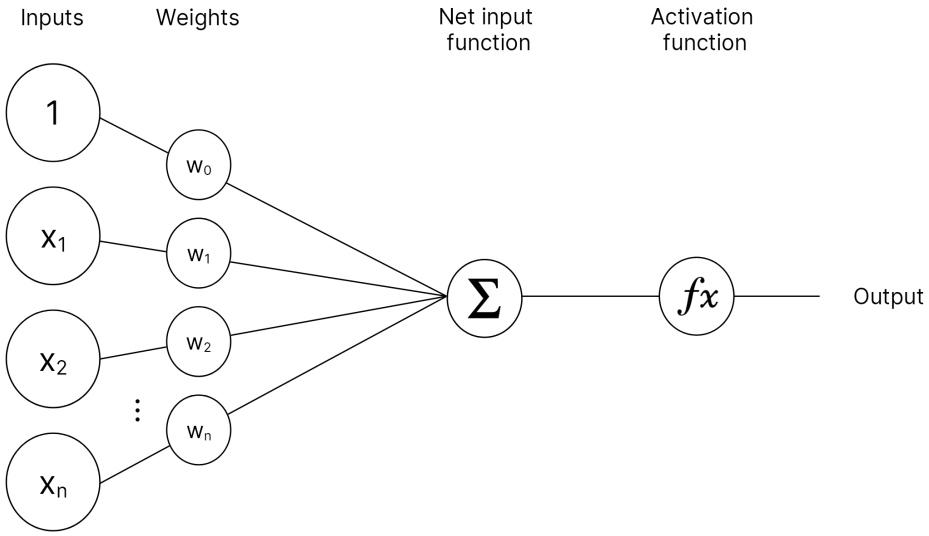

Figure 2.11: An illustration of a simple perceptron.

The model in the latter figure can be used for several purposes as regression or simple classification but can be unsuitable when the problem becomes more complex. When several perceptrons are connected, more advanced patterns can be learned. A model consisting of several perceptrons where one perceptron's output is the input to another is called a multilayered perceptron (MLP). Several perceptrons can be stacked on each other to make a *fully connected layer* to create even more complex models. A model can have several layers between the input and the output layers. These are called *hidden layers* as an end user does not see their workings.

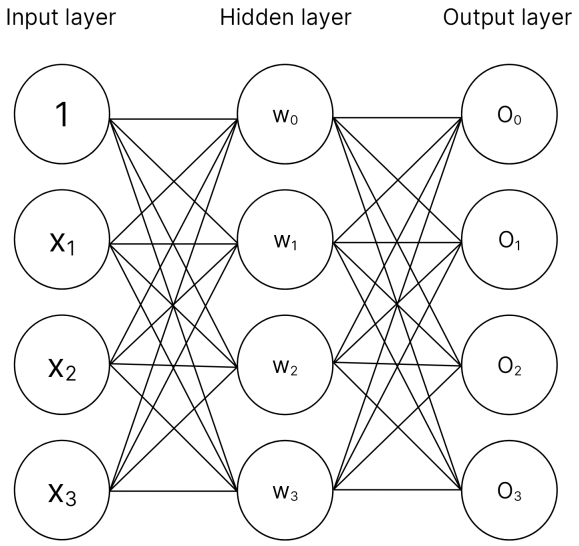

Figure 2.12: An MLP with input, hidden, and output layers. The lines between the layers are weights.

These models can be immense. With each weight in a model being a single parameter, models with millions of parameters are not uncommon. The latest neural language model from OpenAI, ChatGPT-3, has 175 billion parameters [44]. It is important to remember that the number of parameters alone does not indicate whether a model performs well or not.

2.6 Convolutional Neural Networks

A simple image in grayscale can have 64x64 pixels. To accept such an image, a fully connected neural network(FCN) needs an input layer of size 4096. If the image is RGB (meaning the data has three dimensions), the input size would be 12288, resulting in a massive model requiring large computing resources. Today, images are normally much larger than 64x64. HD images contain 1280x720 pixels and are RGB. Such inputs to an FCN are impractical, and the complexity of the model would be immense. Also, two-dimensional images would have to be transformed into one-dimensional. This, again, would make the model focus too much on specific pixels rather than the patterns that could appear anywhere in an image.
Instead, it would be beneficial if the images could be read in their original dimensions and the number of parameters stayed manageable. Using convolutions with a kernel on an image, a neural network could learn how to identify patterns in an image, *invariant* of the position of the patterns. A convolution works by moving a filter(kernel) through the whole image and cross-correlates the values to produce an output.

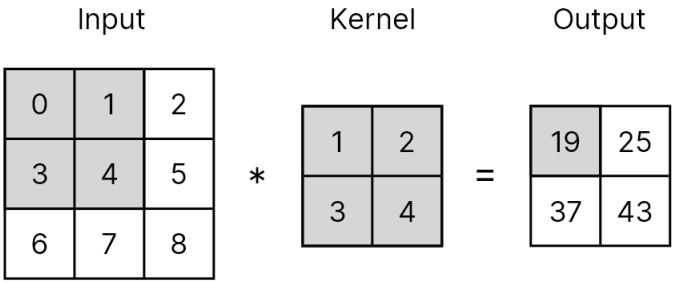

Figure 2.13: Using a kernel with the size 2x2, the whole image can be convoluted to extract features.

The model's job would be to change the weights in the kernel to best generalize to its task. Now, a model's typically fully connected layers can be changed to convolutional layers. These carry many benefits. One benefit is a massive decrease in model complexity

compared with traditional FCNs. Another one is the invariance of the model. Intuitively the model learns to detect things without caring about where it is in the image.
A convolutional neural network can be constructed almost the same way as an FCN. But, as the input of one layer can be multi-dimensional, so can the output.

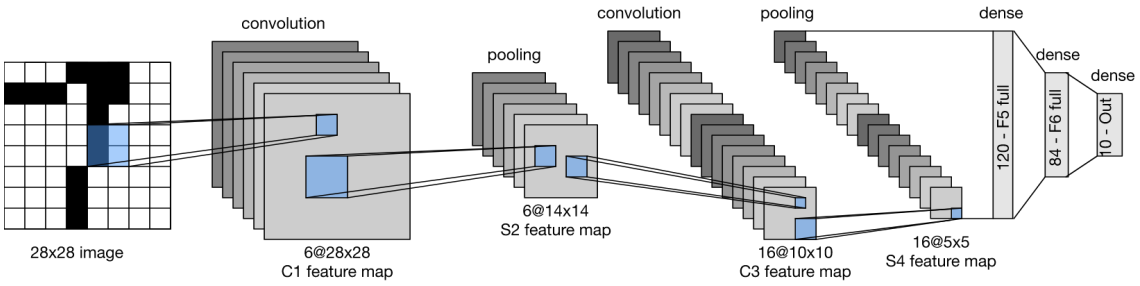

Figure 2.14: An overview of the LeNet model. Notice how the layers accept and output (feature maps) multi-dimensional data. Courtesy of Dive into Deep Learning (2023) [7]

All techniques used in FCNs can be applied to convolutional neural networks, for example, regularization layers and dropout. Regularization layers like batch normalization layers are more important in convolutional neural networks. As seen in Figure 2.14, the dimensions of the feature maps increase when the network is deep. This might lead to spatial sensitivity of the network, decreasing the model invariance. *Pooling* is an important technique to maintain model invariance in larger models, as well as spatially downsampling feature representations [7].

2.6.1 Pooling layers

Two pooling layers are commonly used in convolutional neural networks. The first one is *maxpooling*. Instead of moving a kernel with learned weights over a feature map, the kernel works by only selecting the highest value in the convolution. For *averagepooling*, the kernel computes the average of all values inside the convolution. Using pooling layers, the model might achieve a higher signal-to-noise ratio and reduce the feature maps' dimensions [7]. The pooling operations can be done in the two-dimensional and three-dimensional domains.

2.6.2 Blocks

Earlier convolutional neural networks usually consisted of two parts. One part is several convolutional layers to extract features which are then used in a fully connected part to compute a classification. To develop such networks further, the first part could be extended to calculate larger feature maps. The fully connected part must then be extended to comprehend this large input [7]. The use of blocks was the first step toward stopping this trend. Complex deep models could be developed using a generalized block with convolutional layers and pooling. However, it was not until 1x1 convolutions were used in the blocks that the models could be significantly more efficient.
This discovery was made by Lin et al. in their (NiN) architecture [45]. Using blocks that started with a general convolution layer followed by 1x1 convolutions, the feature maps were kept small. At the end of the network, a global pooling layer calculates the average of each feature map which is then directly used with sigmoid functions to create a classification [45]. This removed the fully connected part of the network, dramatically decreasing its size. Newer models build upon this approach but often include a fully connected layer after the global pooling to apply more non-linearity. Another benefit of this approach is that the model can accept almost arbitrarily sized inputs. Making the model more usable in a production environment.
While blocks, 1x1 convolutions, and average pooling made it possible to create deeper networks, the deepest networks showed a "degradation" problem. Increasing model depth would make a model stop learning and get a high error on the learning data. Adding layers to the model would only make it worse. Combining the input to a block with the output of said block was a discovery that mitigated this problem.

2.6.3 Residual connections

In modern neural networks, a strange problem started to show itself. Increasing model depth would yield better model performance until the performance stagnated and then decreased. Deep models with fewer layers would perform better than deep ones with more. As the performance is measured on training data, it was not an overfitting problem. He et al. recognized this problem and developed a network with *residual connections* [8]. They realized that information from each layer in a neural network would be changed when passed through another layer. This way, information could get lost. Their solution was to let the input to one layer bypass the layer and get added to the input of the next layer. Adding such connections does not increase the model complexity noticeably.

Instead, it makes deeper models converge better and achieve higher performance results than models without residual connections.

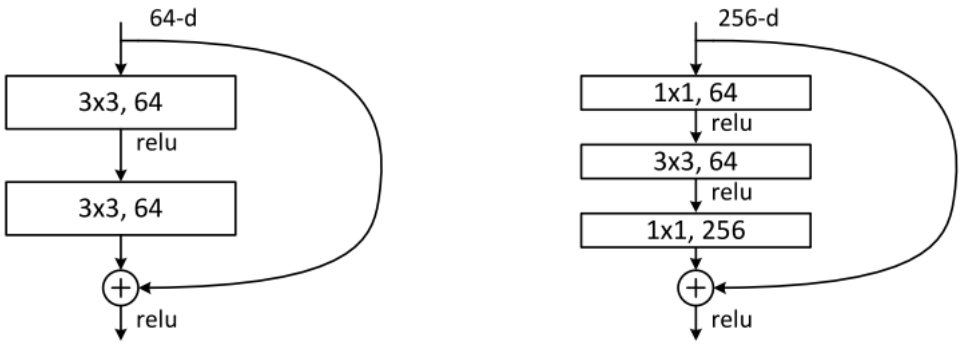

Figure 2.15: Residual blocks. Notice how the input is also added to the output. The block to the right is a "bottleneck" block with 1x1 convolutions. Courtesy of [8]

An interesting feature of residual neural networks(ResNets) is that new layers and blocks can be added to the model *while training* [7]. In some cases, this benefits larger models.
Residual network connections are seen as a breakthrough and are widely adopted in modern neural network architectures.

2.7 DenseNet

ResNets adds one layer's input to the next layer's input. As explained before, a ResNet can have layers added to it while training. Although counterintuitive, the reverse is also possible. Removing layers from a ResNet in training (like dropout) sometimes does not impact the model. This means that a ResNet contains blocks that might contribute little to nothing to the final prediction while leading to expensive complexity. Huang. et al. used this knowledge to propose another type of model architecture. Instead of adding one layer's input to the next layer's output, the input would be *concatenated* to the output of *all* the other layers within a block [9]. These blocks can be seen as having *dense* connections and thus gives name to the DenseNet architecture.

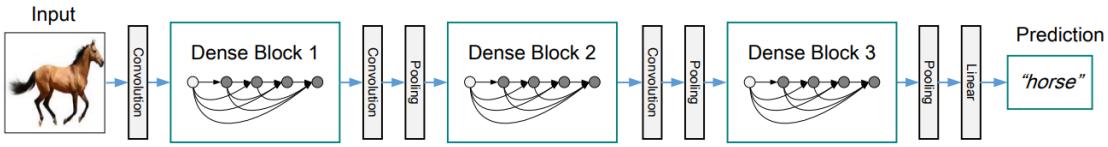

Figure 2.16: Illustration of a DenseNet architecture featuring dense blocks and transitions. Notice how the output from one layer is concatenated to the input of all the other layers in a block. Courtesy of [9]

Concatenating the features drastically increases the number of connections between each layer. Counterintuitively, the model requires fewer parameters to perform comparably to other high-performing models. This is because each layer does not need to learn the feature maps computed by previous layers. Another benefit of DenseNets, rather than the smaller size, is easier learning since each layer can access the gradients from the loss function and the original input signal. Also, a regularizing effect has been observed from the dense connections, which reduces overfitting on smaller datasets [9].

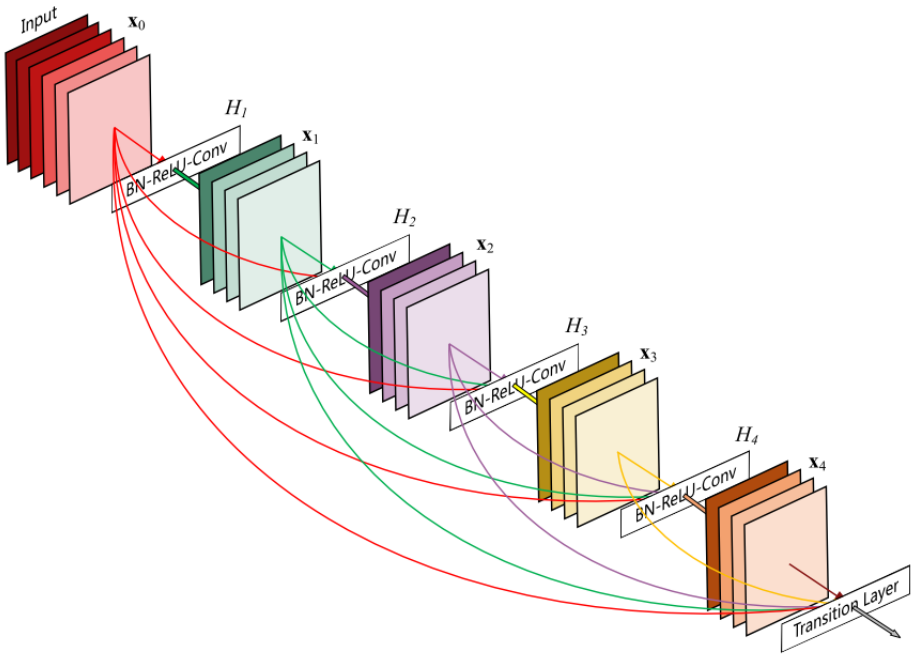

Figure 2.17: An example of a 5-layer dense block with a growth rate of k = 4. Courtesy of [9]

The model architecture can be configured to each specific case on which the model will be used. Two elements are important in the DenseNet architecture, the dense block and transition layers. The dense block comprises smaller blocks(composite function in the paper) consisting of a batch normalization(BN) layer, and a ReLU activation unit,

followed by a 3x3 convolution. How many of these smaller composite functions a dense block consists of is up to the user. Between each dense block is a transition layer consisting of a BN layer, a 1x1 convolution layer, and a 2x2 average pooling layer. A global average pooling layer is used at the network's end, and the output is sent through a single-layer linear classifier.

An important configurable feature of the architecture is the growth rate(k). It denotes how many feature maps each composite function produces and, thus, also regulates how much each layer contributes to the whole network.

As each layer produces k feature maps, they will have many more features as they accept all feature maps from previous layers. A 1x1 convolution before the 3x3 convolution in the composite function acts as a bottleneck layer, downsampling the feature map dimensions. In the paper, it helps with improving computational efficiency.

In the transition layers, the m feature maps from a dense block are compressed to $[\theta m]$. Here θ is a number $0 < \theta \leq 1$. When the bottlenecks are used as well as $\theta < 1$, the architecture is denoted as DenseNet-BC.

Even by using the DenseNet-BC architecture, the architecture might face memory issues due to the concatenation of feature maps, which must be stored throughout the training. With a large input size and a deeper network, this can be an issue that must be dealt with. Thus, a memory-efficient implementation has been developed at the cost of training efficiency [46].

Method

This chapter discusses the characteristic of the patient data, how it was acquired, and how it was divided in the project. Next, the evaluation metrics, the pipeline from pre-processing the images to model configuration, and hyperparameter selection are explained. Implementation details that deviate from cited papers are specifically mentioned. In the end, the final pipeline for AIF extraction is revealed.

The code used for the thesis is provided GitHub repository accessed by following the link: `https://github.com/Skuvrik/Masterthesis`.

3.1 Patient data

The patient dataset consists of two sub-datasets. One dataset contains 70 DCE-MRI sequences of the brain from 70 patients. The patients have no visible pathologies, and the whole head is imaged. Each patient is only imaged once. The other dataset consists of DCE-MRI sequences of 27 patients imaged while undergoing and recovering from radiochemotherapy treatment [47]. This dataset is more heterogenous as several patients show visible signs of pathology, and the volumes only consist of slices of and around the area of interest. This way, some patients only have volumes of the upper, lower, or middle part of the brain. Additionally, each patient has had imaging done several times over different periods of time (series).
Of the total 97 patients in the dataset, only 80 are kept. One patient is excluded due to an incomplete image dataset. One is excluded due to significant artifacts, and 15 patients are discarded due to no visible MCA.

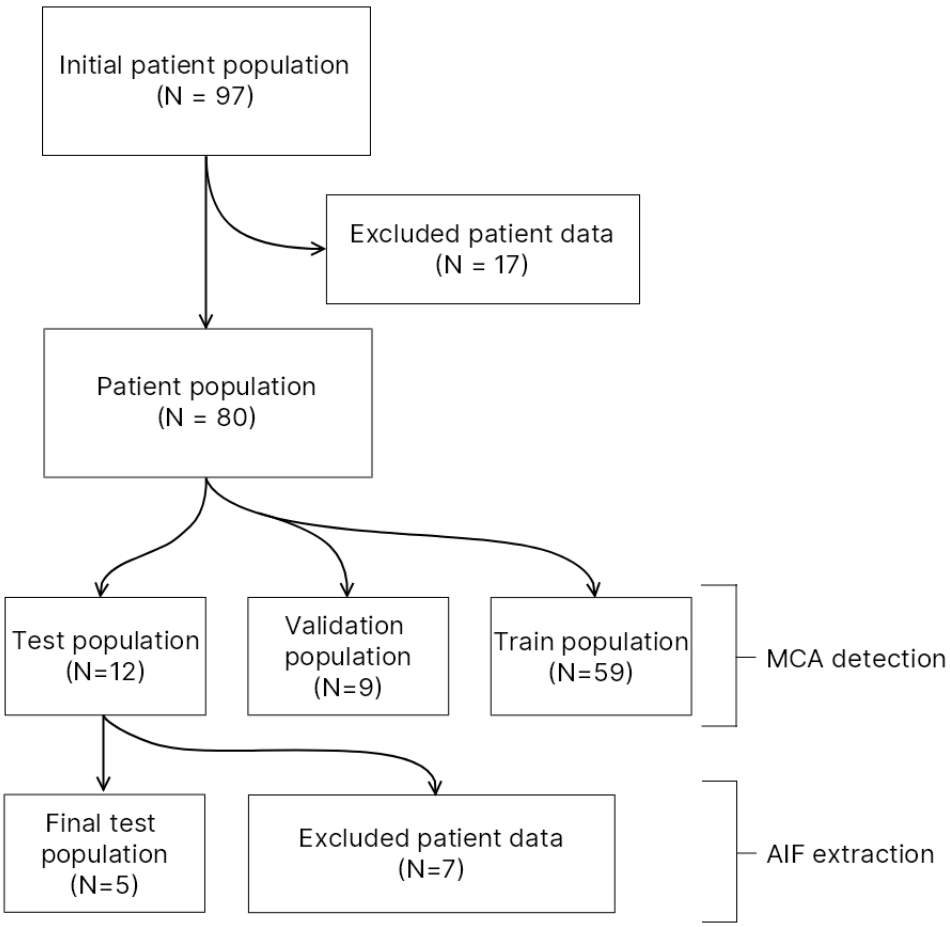

Figure 3.1: Patient exclusion and final division to different data batches.

After exclusion, the dataset was randomly divided into three separate batches. First, 74% of the patients were assigned to the training dataset. This is used only for the training of the model. Second, 11% were assigned to the validation dataset, which is used for hyperparameter tuning. Lastly, 15% of the patients were assigned to the testing dataset, which is used in the end to evaluate the final MCA recognition model. Here only the five first series for each patient are kept, as these series were done within eight weeks. A series did not contain a bolus arrival for one patient in the testing dataset. Thus, the series was discarded.

The pipeline with both the MCA recognition model and a clustering algorithm should predict the same AIFs for the same patients over time. Therefore a final testing set is made by only keeping the patients with several series from the original testing dataset to evaluate the pipeline.

The motivation for the division was to have enough patient data to train a well-performing MCA recognition model, have enough data to get a fair estimate of model performance

in the validation phase, and in the end, have enough patients with several series to do a fair evaluation of the AIF pipeline. Thus, more patients are in the testing than in the validation dataset.

3.1.1 Data aquisition

For the 70 patients with only one image series, in-vivo DCE-MRI sequences were acquired using a 3T MRI system (Skyra, Siemens Healthineers, Germany) [48]. A total of 3840 images were acquired using a 3D T1 spoiled gradient echo sequence (VIBE) with TR = 2.88 ms and a variation of TE and flip angles for each patient. These are shown in the appendix. Each sequence has 80 time points, and the temporal resolution is 7.2 seconds. The slice thickness was 3 mm for all patients. For two patients DOTA-Gd (Clariscan, GE Healthcare, USA) was administered, while the rest of the population was administered DOTA-Gd (Dotarem, Guerbet, France).
For the 27 patients with several image series, the details of the patient group and the image acquisition are described in a paper by Larsson et al. [47]. In short, the patients in the study had been diagnosed with brain cancer and were to begin radiochemotherapy treatment. Each patient was imaged at the start of the treatment and every second week for eight weeks (five scans). Only these five series per patient were used in the testing set for AIF extraction. Additionally, the patients were imaged every third month for three years (11 scans). In the study, T1-baseline images were taken and the tumor ROIs were placed for each patient. These were not provided in the dataset provided for this thesis. For all patients, each slice in each series was labeled 1 if the MCA was visible or 0 if not. As this was deemed a simple task, it was done by the author.

3.1.2 Data shapes

The data used in this thesis is represented by tensors. A visualization of this can be seen in Figure 3.2. As described in subsection 2.1.2, a slice consists of voxels with a depth dimension. In the data, the voxels have been transformed to be two-dimensional, as ordinary pixels. Still, they will be referred to as voxels throughout the thesis.

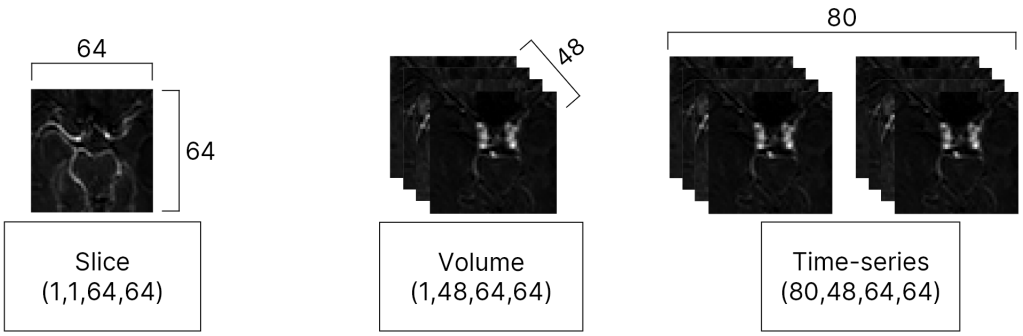

Figure 3.2: A visualization of data shapes used in the thesis. The numbers in the boxes represent the tensor shape.

To extract AIFs from a time series, a slice (or several) has to be selected. In the temporal domain, each voxel represents a signal curve. In Figure 3.3, three voxels are selected, marked by the blue, red, and green squares. Each curve consists of the value for the selected voxel at every time point in the time series.

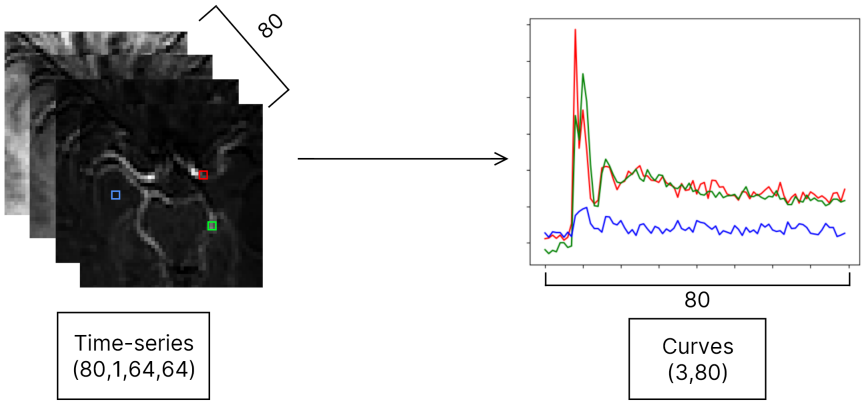

Figure 3.3: A representation of how curves are extracted from a time series with a single slice

The input and output shapes of the data in different parts of the thesis can be seen in Figure 3.5.

3.2 Pre-processing

Before using the data in a model, it has to be preprocessed. This thesis's preprocessing consists of four steps; normalization, cropping, data filtering, and data enhancement. The data enhancement is only used for model training.

Artificial deep neural networks benefit from the normalization of input values. This is done for each slice, where each voxel value is divided by the maximum voxel value in the slice. Thus, the voxel values never exceed 1.
Initially, a whole slice was used as the input for the model. The largest slices consisted of 180 rows and 180 columns. This yielded high training times (days), with an unstable convergence and a bad model generalization. Most likely, the high amount of voxels around the brain contains noise that disturbed the model. To mitigate this issue, a center cropping is done on the slice, inspired by the ROI placement done by Peruzzo et al. [21]. As explained earlier in subsection 2.3.1, the MCA recedes in the middle of the brain, with some segments stretching to the edges. This is true for all of the brains in the training dataset. Therefore, a center-cropping mask was developed only to leave the slice's center, amounting to 70% of the image area. This was tested empirically on all patients in the training dataset to confirm that it did not remove the M1 and M2 segments of the MCA. However, it should be noted that most peripheral parts of the M3 segments are removed. After applying center cropping, the training times were reduced (minutes), and the model achieved a much greater generalization.

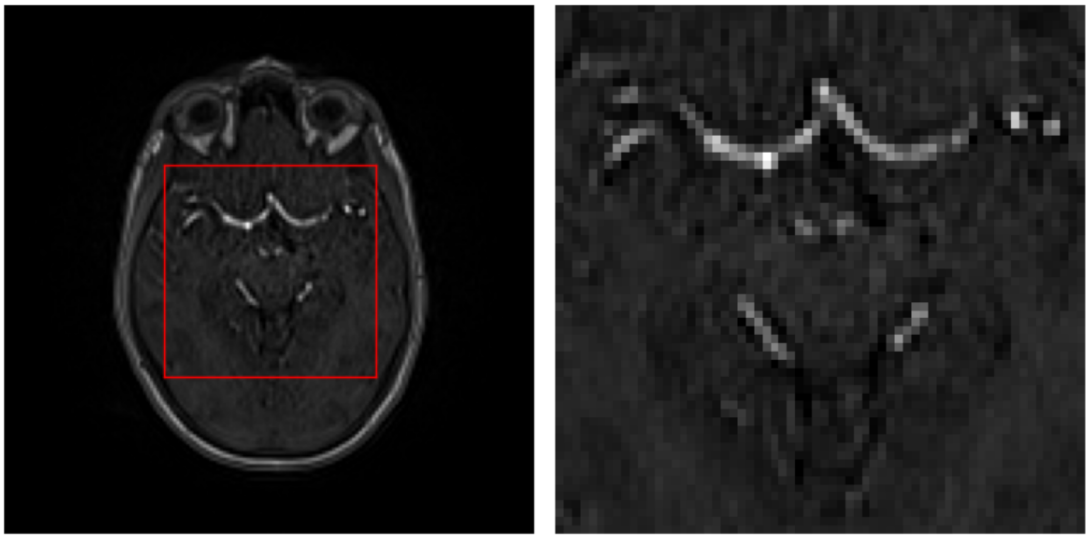

Figure 3.4: Illustration on how the slice is cropped. The M1 segment can clearly be seen.

As model input, the volume showing the bolus arrival is desired. This is when the MCA is most visible and intuitively the best image for a model to recognize the MCA. Each volume in the time series was summed to calculate bolus arrival time, and the gradient was calculated between each timestep. The selected volume was the one with the highest sigmoid value.
After filtering, the model has only a single volume per patient and still a high imbalance

between slices containing an MCA and slices that do not. Therefore a data enhancement was done to include volumes in timesteps after the selected volume of the slices containing an MCA. These display slight variations from the initially selected volume and therefore help the model generalize. Only the training dataset was enhanced this way.
All preprocessing was done by using the Numpy and PyTorch packages. A detailed description can be found in Table B.2.

3.3 Model Architecture and training pipeline

With high success in image recognition tasks, while having a dense architecture, the DenseNet architecture was used for the MCA recognition part of the pipeline. Compared to other equally good models like ResNet, it is faster to train and requires less space on the GPU. The model code is taken from PyTorch's own repository [49]. It is inspired by the original paper, with the added implementation of memory-efficient training[46]. In this thesis, the classifier is changed from 1000-dimensional to binary.

3.3.1 Custom sampler

As mentioned, the training dataset consists of slices with different dimensions. This does not work with PyTorch's standard data loaders. It is not possible to concatenate images of different dimensions to a minibatch. Thus, a custom batch sampler ensured that each minibatch only consisted of images of the same dimensions. The code can be found in the provided GitHub repo.

3.3.2 Model selection

Finding the best model configuration is done by using the Ray-tune library [50], using the recommended ASHA scheduler and a TPE search algorithm [51, 52]. The search space was selected by tuning once with arbitrary value ranges. If the model achieved good results at the edge of a value range, the new HPO was conducted with the range expanded in that direction and contracted in the opposite direction. This can be seen as "coarse" tuning and was not given many iterations. When updating the search space with the new ranges, "fine" tuning was done with more iterations. The final search space

is given in Table 3.1. Here, it is evident that the model architecture is mainly optimized, while only one hyperparameter is optimized for the ADAM optimizer.

Table 3.1: Hyperparameter value ranges used in the model selection.

Hyperparameter	**Value range**	**Value steps**	Explanation
Learning rate	1e-6 to 1e-3	Float	Used in optimizer
Droprate	0.1 - 0.6	Float	Dropout rate after each dense layer
BN size	2 - 7	Int	Growth rate for bottleneck layers
Growth rate	10 - 17	Int	k in the densenet paper [9]
Initial features	20 - 29	Int	Filters in the first convolution

Each model is trained to a maximum of 100 epochs, although the scheduler can interrupt the training. The batch size (size of each mini-batch) is set to 128 samples. All models use the ADAM optimizer. The model with the highest combined accuracy, precision, and recall is selected.
Before evaluating the model on the testing data, the validation and training data are concatenated and used to train the model again. This model is also what is used for the final pipeline.

3.3.3 Training setup

The models were trained on an Nvidia GeForce GTX 1080 GPU with 8 GB of dedicated VRAM. More details can be seen in Table B.1.

3.4 AIF extraction pipeline

The final AIF extraction pipeline consists of two parts: the model that recognizes the MCA structure and a k-means search algorithm based on the paper by Mouridsen et al. [1].

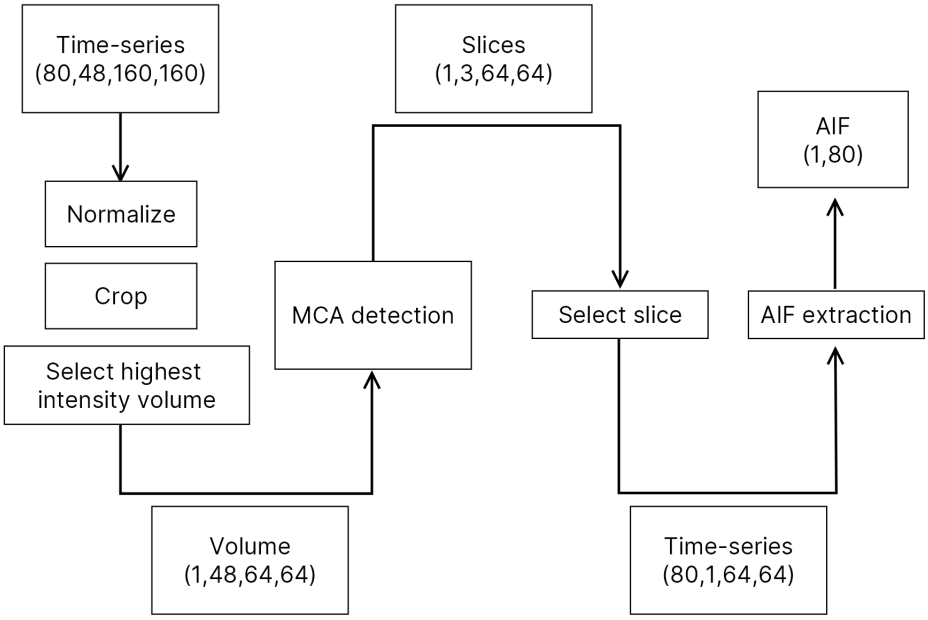

Figure 3.5: A schematic describing the AIF extraction pipeline. The image boxes display how the data shape changes through the pipeline. An example input consisting of a tensor with 80-time points, 48 slices, and 160 rows and columns for each slice is shown. The slice selection only selects a single slice of the three detected slices, and only a single AIF is output.

First, the time series is normalized and cropped. Then the volume with the assumed bolus arrival is selected. For each volume in the series, all pixel values are summed to a single value. Then, the derivative is calculated to find the time of the highest pixel intensity change between two volumes. Of the two volumes having the highest change of intensity between them, the latest volume is selected.
The model predicts a probability on each slice for it to contain an MCA. A sigmoid function is used to make the probability interpretable. If the probability exceeds 0.5, the prediction asserts to 1 (visible MCA). If the probability is below 0.5, the prediction asserts 0 (no MCA).

3.4.1 k-means AIF extraction

To extract the AIF from the selected slice, the method from Mouridsen et al. is applied. Assuming how an appropriate AIF curve would look, Mouridsen et al. developed a semi-automatic method to extract an AIF from DCE-MRI sequences [1]. It was developed early, but due to ease of use and general success in identifying a suitable AIF, it is yet in wide use today. Looking at all curves in a given slice or for several slices, the model

applies different filters to remove curves that do not fit the assumption of an AIF curve. It finally applies k-means on the remaining curves twice. The cluster of curves with the lowest average first moment is then selected, and the procedure continues until a specified number of curves remains.
A variation from the original paper is done by selecting the best cluster based on the highest peak divided by the area under the curve (AUC). This is due to unfavorable results produced while using the first moment as the selection factor.
Intuitively, the highest peak divided by the AUC will indicate the "sharpest" peaks, which also is the purpose of selecting the lowest first moment.
The resulting extraction method can be summarized as the following procedure:

1. Remove 90% of the curves with the lowest area under the curve.
2. Standardize curves to have a unit area under the curve.
3. Remove 25% of the most irregular curves, see Equation 3.1.
    3.1. Use k-means to cluster the remaining curves into 5 clusters.
    3.2. Calculate the average curve for each cluster and use this to decide which cluster to choose. The cluster with the average curve that has the highest peak value divided by the AUC is selected, see Equation 3.2.
    3.3. Repeat once.
4. The average curve from the final cluster is the resulting AIF.

To calculate the irregularity of each standardized curve, the area under the curve of the squared second derivative is calculated, which yields the function:

$$SI(t)_{irr} = \int_0^\infty (SI(t)'')^2 dt \tag{3.1}$$

The peak value divided by the AUC is calculated by:

$$SI(t)_{peak/AUC} = \frac{max(SI)}{\sum(SI(t))} \tag{3.2}$$

This algorithm can be applied to a whole MRI volume, slice, or ROI.

3.4.2 Slice selection method candidates

To determine whether the approach of this thesis is better than the methods in use today, three approaches will be investigated and compared. They only differ in how they select the slices to extract the AIFs. The three methods are:

1. Method **A**: Using the slice with the highest probability given from the MCA-detection model.
2. Method **B**: Using all the slices predicted by the MCA-detection model to contain an MCA. For a slice to be assigned to contain the MCA, the probability must be > 0.5.
3. Method **C**: Using all slices in the whole volume. This is the baseline method.

3.4.3 Conversion from signal to R1

The signal is converted to the relaxation rate R1 to compare the extracted curves with each other. The conversion formula is provided in the paper by Larsson et al. [47], and yields:

$$\delta R_1(t) = R_1(0)[\frac{SI(t)}{SI(0)} - 1] \tag{3.3}$$

Where $R_1(0)$ and $SI(0)$ denote the relaxation rate and signal intensity at timepoint zero. To decrease the effect of random noise, $R_1(0)$ and $SI(0)$ are made respectively by averaging the relaxation rate and intensity for the four first time points.

3.5 Evaluation

There are two kinds of evaluations done in this thesis. First, the ANN model must be evaluated for its recognition of the MCA structure. Second, the whole pipeline must be evaluated on intra-patient variability.

3.5.1 MCA detection evaluation

To evaluate how well the model recognizes the MCA structure, the accuracy, precision, and recall metrics are established methods that, when seen together, clearly indicate how the model performs. A receiver operating characteristics (ROC) curve is also made to evaluate the model sensitivity.
A classification task may consist of a problem classifying data into a single class or many classes. Either way, the model might predict a true positive(TP) or true negative(TN). Both times the model is correct according to the given truth. The most intuitive way to measure the model performance is to calculate the accuracy:

$$Accuracy(y, \hat{y}) = \frac{1}{n} \sum_{i=0}^{n-1} 1(y_i = \hat{y}_i) \tag{3.4}$$

Where $1(y_i, \hat{y}_i)$ is the indicator function that returns 1 if the values are equal or 0 if the values are not equal.
Following this approach, we know how many correct predictions the model has made as a fraction of the total number of possible predictions. Yet, this might lead to little insight if one works with unbalanced class distributions. If a dataset with a class distribution consisting of 70% A-labels and 30% B-labels, the model can predict class A every time and achieve an accuracy of 70%. It is, therefore, interesting to investigate the false negatives(FN) and false positives(FP) to understand how the model performs for each class. Accounting for these in the performance measurements can provide a more nuanced view of the model performance. Precision measures how often the model correctly identifies a class and is given by: $TP/(TP + FN)$. Recall measures how often the model correctly predicts if a data point does not belong to a specific class, and is given by: $TN/(TN + FN)$. These metrics should be evaluated together to understand how a model works properly. In this thesis, the model should ideally achieve higher precision than recall. It is worse to label a slice not containing an MCA as having an MCA than the opposite.

3.5.2 AIF extraction evaluation

Since there is no given truth of how the AIFs should look, the intra-patient variability, or repeatability, is evaluated. As described by QIBA "*Repeatability represents the measurement precision, or closeness of agreement, of replicate measurements made over a short*

period of time." [13]. Since the patients in the testing dataset have been imaged with the same conditions for each series according to the paper [47], the AIF extractions from a patient should, in theory, yield the same results across the series.

Evaluating the intra-patient variability of the AIFs is done by comparing the series of each patient independently from the other patients. After conversion to R1 values, several metrics are calculated and compared to evaluate the intra-series variability.

Since the patients achieve the same amount of CA for each imaging session, it can be assumed that the AUC should also be the same for each session. When injecting CA into the patient, an automatic power injector is used, and it is assumed that the injection is located at the same place every time. Therefore, it can be assumed that the bolus arrives in the MCA simultaneously with the same amount of concentration for each session. This is measured by the time-to-peak (TTP) and peak-to-baseline (PTB), respectively. The TTP is the time when a curve reaches its highest value, and the PTB is the highest value minus the lowest value.

Lastly, if all of the aforementioned metrics are similar, it should also be assumed that the AIFs should look the same. This is measured by the Pearson Correlation Coefficient, given as:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{3.5}$$

For a given set of samples, n, the values x and y are the different signal values to be compared against each other. $\bar{x}$ and $\bar{y}$ are the mean for each signal. The higher the value, the more similar the signals are in shape.

Results

4.1 MCA detection model results

After hyperparameter optimization, the model with the highest combined accuracy, precision, and recall is selected. The five models with the lowest loss are seen in Figure 4.1.

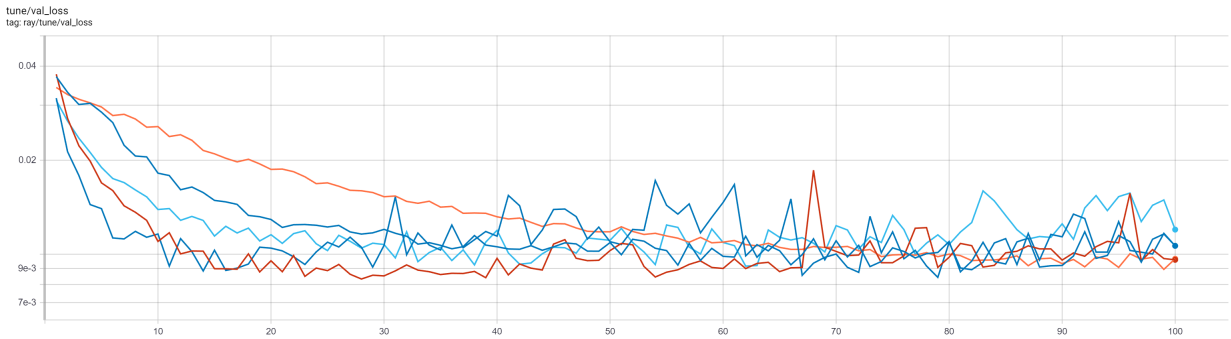

Figure 4.1: The final five best performing models from the hyperparameter optimization. The number of epochs is along the x-axis, and the validation loss is along the y-axis. The y-axis is in logarithmic scale. Each curve represents a single model, and the red curve belongs to the chosen model.

The hyperparameters used for the final model are specified in Table 4.1, which results in a model with 1.5 million trainable parameters.

Table 4.1: The final hyperparameters for the model. The learning rate is used for the optimizer, while the others are used to define the model architecture.

Hyperparameter	**Value**
Learning rate	3.09e-05
Droprate	0.018
BN size	4
Growth rate	15
Initial features	26

After combining the training and validation datasets, the model is trained again. The training and validation losses (validation loss is calculated from the test set) can be seen in Figure 4.2. A convergence of the validation loss happened around epoch 20. Thus, the model would not have needed 100 training epochs. Some volatility is seen in the validation loss curve, but it should be insignificant to the model generalization.

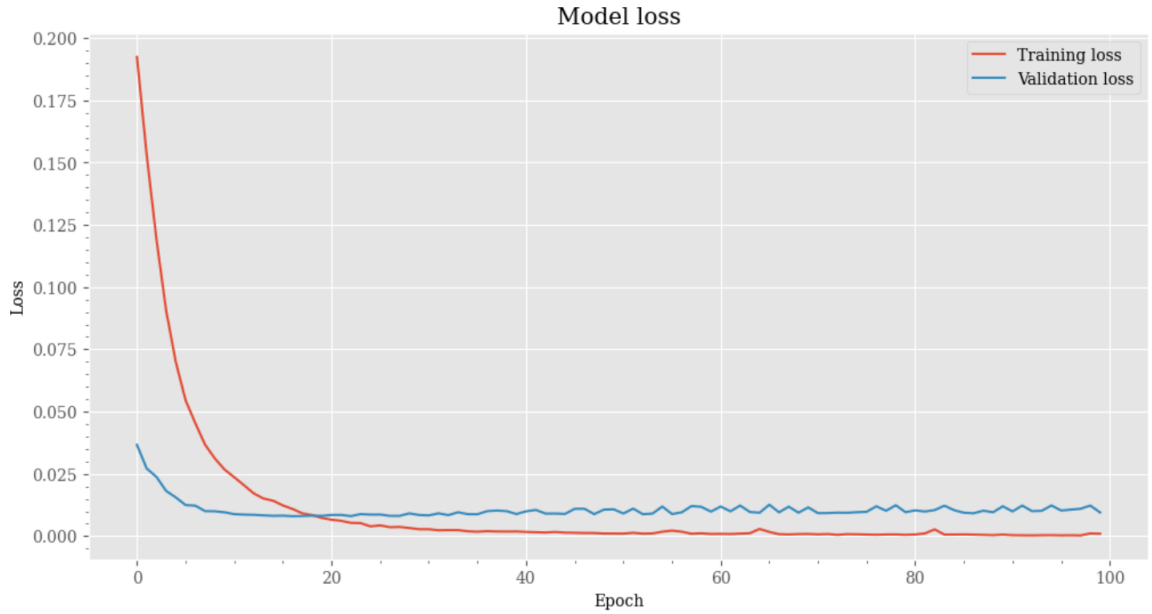

Figure 4.2: Training and validation loss per epoch while training the final model. The validation loss is computed from the testing dataset. Notice how the validation loss is lower than the training loss at the start. This is due to regularization factors (dropout) that are enabled while training but disabled while evaluating.

The accuracy, precision, and recall are calculated for each epoch to inspect the model generalization further. The result can be seen in Figure 4.3. Here it also shows that the accuracy, precision, and recall converge at 20 epochs. The precision and recall vary throughout the training. When one metric increases, the other decrease. One reason for this could be mistakes in the labeling of the data. When the model recognizes a specific pattern to indicate the presence of an MCA in one batch, another batch might contradict it.

Figure 4.3: Accuracy, precision, and recall on the testing dataset per epoch while training the final model.

After training, the model is used on the combined and testing datasets to examine its performance. The results can be seen in tables 4.2 and 4.3. As expected, the accuracy, precision, and recall are high on the dataset the model has been trained on. On the other hand, the model has a lower precision on the testing dataset. The accuracy and recall are also lower, but not have not decreased the same amount as the precision. A low precision metric means that the model has a higher chance of misclassifying a slice not containing an MCA as having an MCA.

Table 4.2: The true positives, false positives, true negatives, and false negatives predicted by the model for each dataset.

Dataset	Total images	**TP**	**FP**	**TN**	**FN**
Training and validation	4616	1388	0	3224	4
Testing	831	128	19	661	23

Table 4.3: The accuracy, precision, and recall for each dataset.

Dataset	**Accuracy**	**Precision**	**Recall**
Training and validation	99.9%	100%	99.9%
Testing	94.9%	87.1%	96.6%

The receiver operation curve and its AUC are calculated to analyze the model's sensitivity, meaning how certain the model must be to provide good classifications. A perfect

classifier has a true positive rate at 1.0 when the false positive rate is at 0.0. As Figure 4.4 shows, the model can correctly classify all positives when the false positive rate is slightly above 40%.

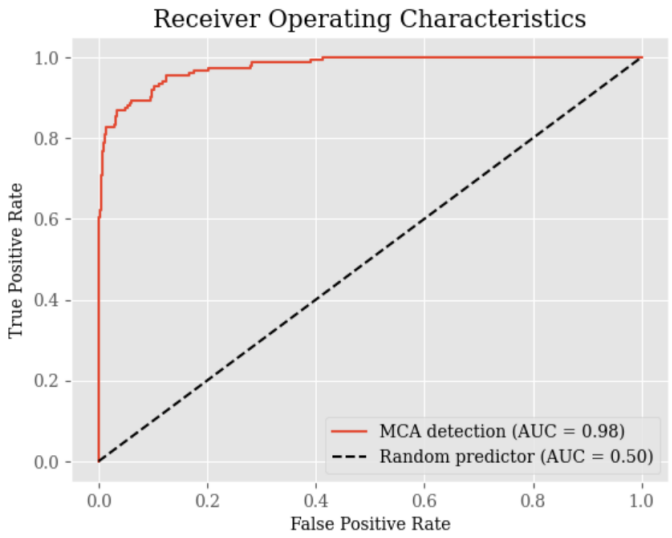

Figure 4.4: The receiver operating characteristics.

To see the actual predictions made by the model, Figure 4.5 displays a random selection of true positive, true negative slices, false positive, and false negative predictions. In the column of false positives, the last two slices contain structures that might resemble an MCA. The first three slices in the false negative column have a distinct curved structure with pronounced centers. These might resemble structures lower in the brain, like the ICA, which might mislead the model.

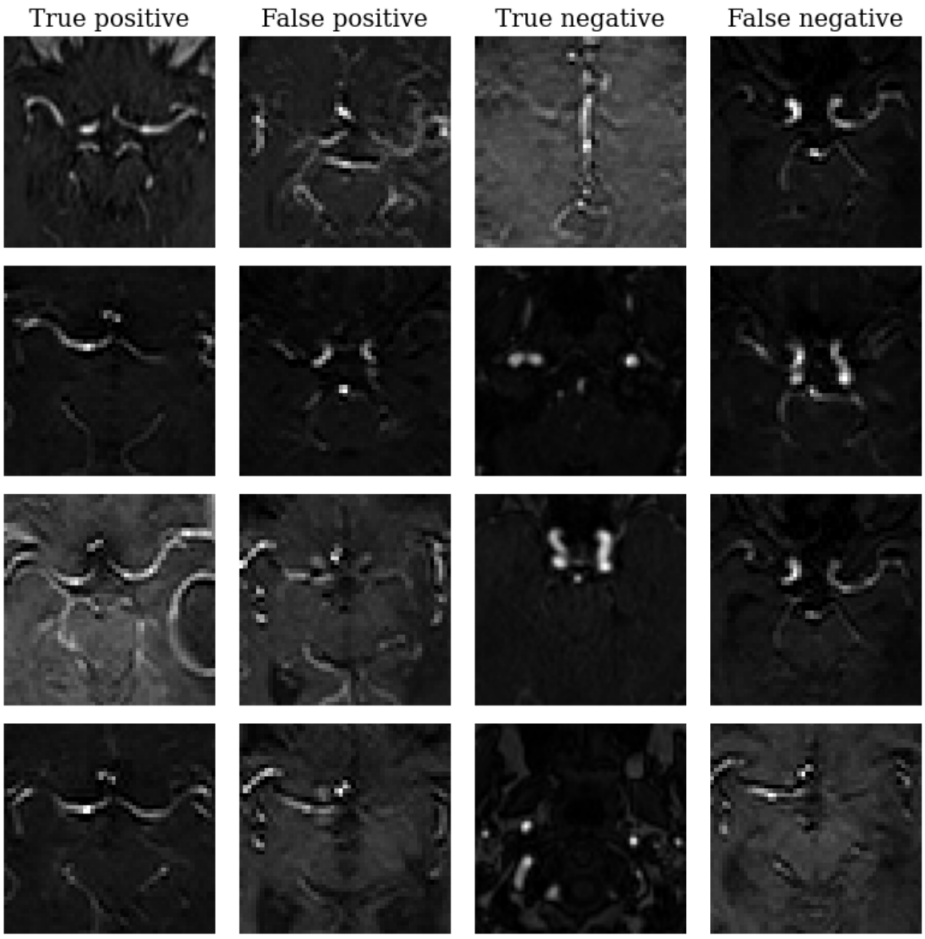

Figure 4.5: A random selection of images the final model classifies correctly and incorrectly. It is possible to see that some of the false positives display structures that resemble an MCA.

4.2 AIF extraction results

With the best model selected, the pipeline is tested on the AIF extraction test set with slice selection methods A, B, and C. The AUC, TTP, and PTB were calculated for each patient's series. Figure 4.6 displays a box plot for all the respective values calculated for the dataset.

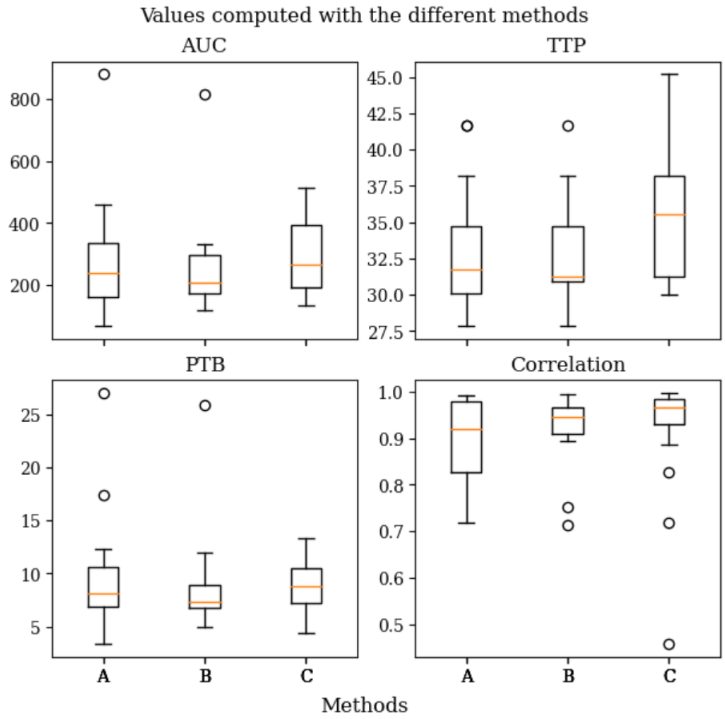

Figure 4.6: Boxplot of the values computed for the different series and the correlation between each following series for each patient. The box extends from the first to the third quartile of the data, with the median indicated by a yellow line. The whiskers extend x1.5 from the interquartile range. The circles are outliers that are more than 1.5 times the interquartile range. The unit for AUC is $[\frac{1}{s}]$, TTP is seconds, PTB is $[\frac{1}{s}]$, and the correlation is arbitrary.

From the figure, it is evident that method B, with the exception of three outliers, produces the most consistent metrics. Method A has some more outliers and less correlation between the curves of each patient series. Method C yields no outliers for the AUC, TTP, and PTB metrics but has a larger variance than the other methods, excepting the outliers. An interesting observation is the larger variance in the TTP metric when using method C compared to the others. This can be further examined by calculating the mean TTP and standard deviation displayed in Table 4.4.

Table 4.4: Mean TTP values and the standard deviation for the different methods.

Method	A	B	C
TTP[s]	32.94 (3.85)	32.69 (3.67)	35.48 (4.18)

Methods A and B display a slightly lower mean TTP and a lower standard deviation than the TTP computed with method C. This might indicate that method C has selected

voxels for the AIF that are veins, or small arteries, further away from the larger vascular structures. This is unfavorable when using the AIF in PK modeling.

To examine the intra-patient differences, the differences in the values are calculated between the series for each patient. The average differences and the standard deviation are shown in Table 4.5.

Table 4.5: The average difference for all patients and series, and the average Pearson correlation. The standard deviation is in parentheses. All correlations have a p-value $<$ 0.01. The lowest averages are in bold.

	Average difference between series			
Method	**AUC**	**TTP**	**PTB**	**Correlation**
A	-10.51 (185.82)	0.63 (2.94)	-0.39 (5.13)	0.90 (0.09)
B	28.74 (128.04)	0.70 (2.32)	0.69 (3.96)	**0.92** (0.07)
C	**5.09** (54.70)	**0.59** (3.41)	**0.06** (1.74)	0.91 (0.13)

The table shows that method C yields the lowest average differences for the AUC, TTP, and PTB metrics. Method B has a slightly better average correlation between the curves, as also shown in Figure 4.6. The higher standard deviations for both AUC and PTB when using methods A and C are not surprising, as these might originate from the outliers shown in Figure 4.6. Such outliers also impact the mean. The boxplot for the differences is displayed in Figure 4.7.

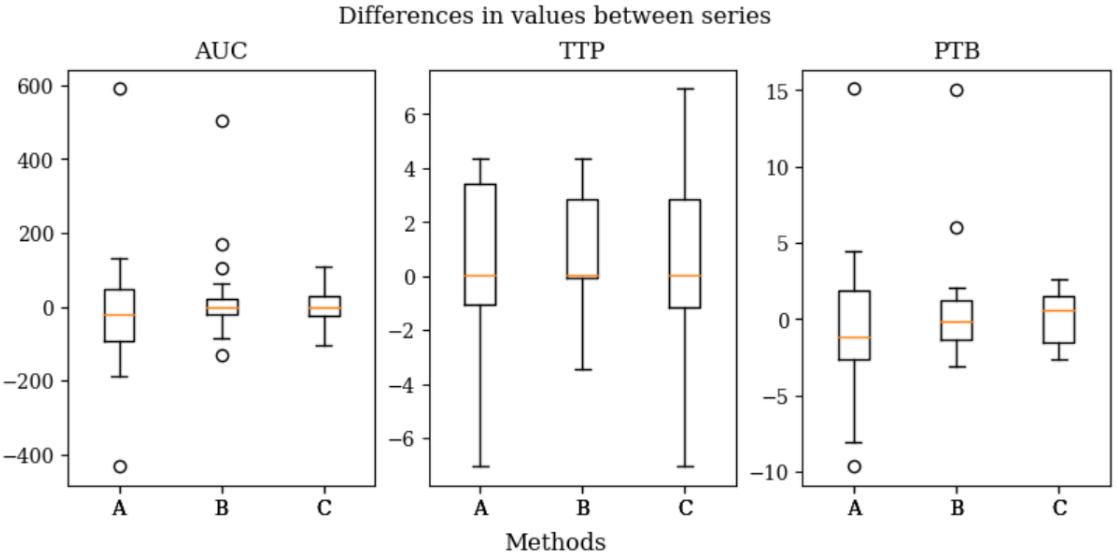

Figure 4.7: Boxplot of the differences for each metric. The box extends from the first to the third quartile of the data, with the median indicated by a yellow line. The whiskers extend x1.5 from the interquartile range. The circles are outliers that are more than 1.5 times the interquartile range. The unit for AUC is $[\frac{1}{s}]$, TTP is seconds, PTB is $[\frac{1}{s}]$, and the correlation is arbitrary.

The figure shows that both methods A and B have produced some outliers, meaning the difference in AUC and PTB between two of the series is large. Still, without these outliers, method B has a lower variance than C and a lower median difference for PTB. Method B displays no outliers for the differences in TTP and has a lower variance than methods A and C.
A detailed table with mean differences and standard deviations can be found in the appendix A.2. From this table, patient 2 stands out as the one in which method B produced the lowest differences and patient 5 the worst. The extractions from these patients are examined further to understand the outliers and differences.

4.2.1 Curves

The produced AIFs for patients 5 and 2 are shown in the figures 4.8 and 4.9, respectively.

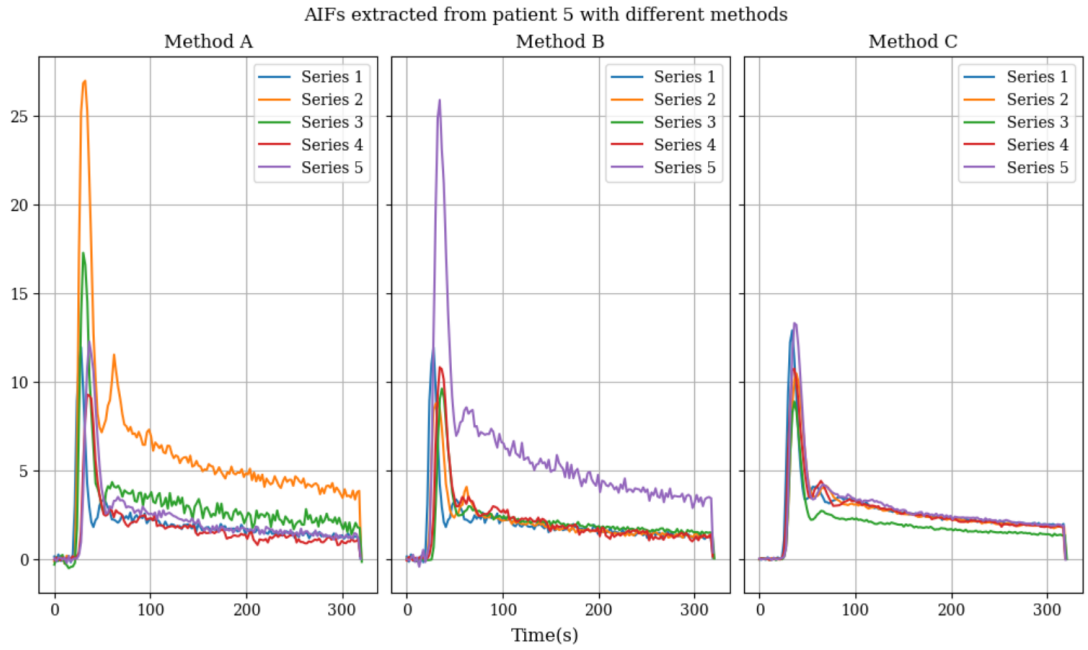

Figure 4.8: The AIFs $[\frac{1}{s}]$ selected from patient 5 with different methods. Each series is a DCE-MRI acquisition of the patient at different times. They are chronologically listed.

It is clearly visible that series 2 for method A and series 5 for method B are outliers. Method B would have produced more similar curves without this outlier. However, it should be noted that there is a variation of the TTP, which means that the AIFs have been extracted from different places with different mean bolus-arrival. Method C yields much more consistent curves, with only the AIF from series 3 being somewhat different

from the rest. The irregularity of the curves extracted with method C seems lower than for methods A and B. Not looking at the outliers for methods A and B, method C also seems to extract AIFs with higher mean PTB.

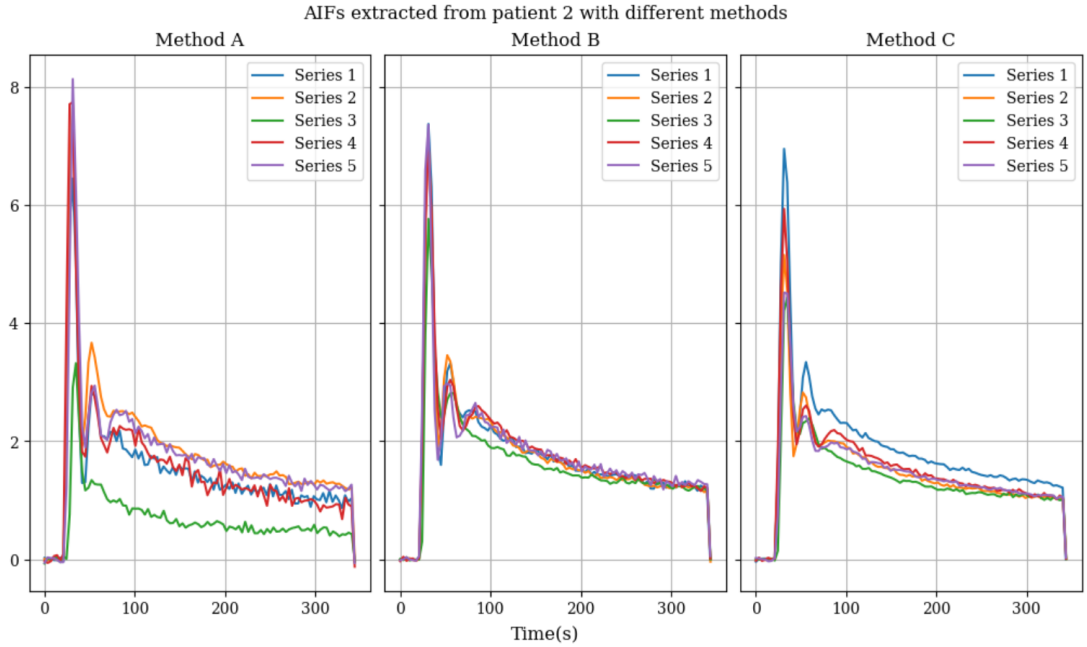

Figure 4.9: The AIFs $[\frac{1}{s}]$ selected from patient 2 with different methods. Each series is a DCE-MRI acquisition of the patient at different times. They are chronologically listed.

For patient 2, method B has the most similar curves. They have a higher mean PTB than the curves extracted with method C and more consistent AUCs. Still, it seems like method A and B produces somewhat more irregular curves than method C, similar to the case of patient 5.

4.2.2 Voxel locations

To investigate further why the curves extracted with methods A, B, and C differ, it will be insightful to see where they are extracted from. Here, it is interesting to see if the voxels lie within the MCA and if they are in the same places for each series.
For patient 5 in Figure 4.10, both methods A and B select voxels from the MCA in series 1 and 2. This changes in the later series, where the voxel selection is inconsistent. Method C selects voxels from the lower middle part of the slice and not the MCA. Here, the selection is more consistent compared to the other methods. However, these areas

do not correspond to large arterial vessels; thus, they might introduce late TTPs to the average function.

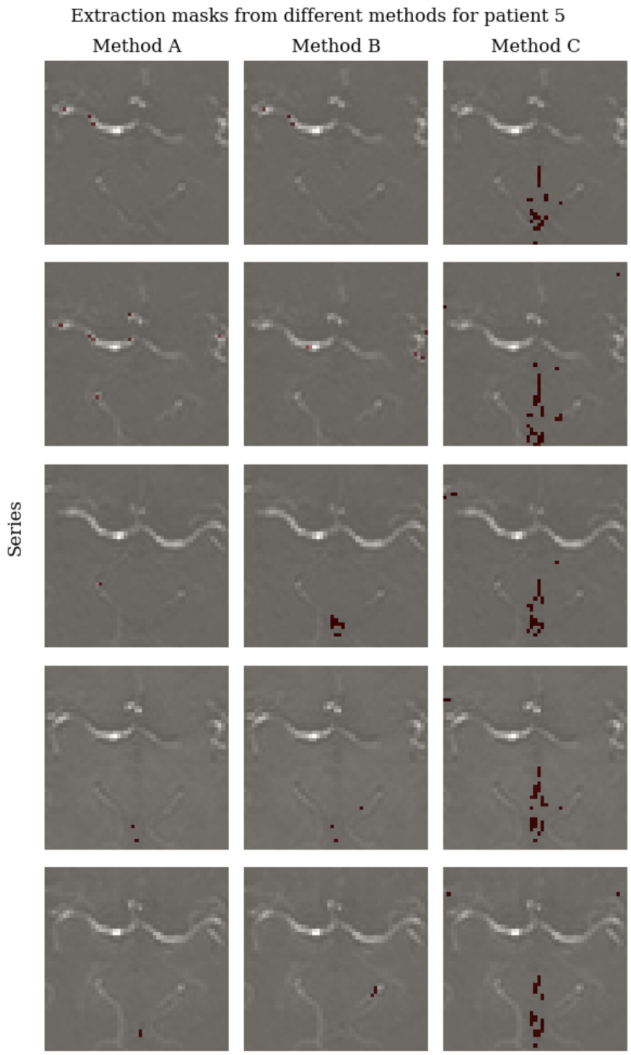

Figure 4.10: Voxels where the AIF is extracted from patient 5. Each row is the same series.

In Figure 4.11, method A still selects the voxels inconsistently. Sometimes from the MCA and other times from what is assumed to be the posterior communicating artery (PCA). Method B and C, conversely, select voxels from both the MCA and the PCA consistently for each series, with the majority of voxels being in the PCA. The only difference between methods B and C is the different amounts of voxels selected. Method C sometimes selects voxels outside of the arterial structures, e.g., the upper part in series 5, the top right, and bottom left areas in series 3, 4, and 5.

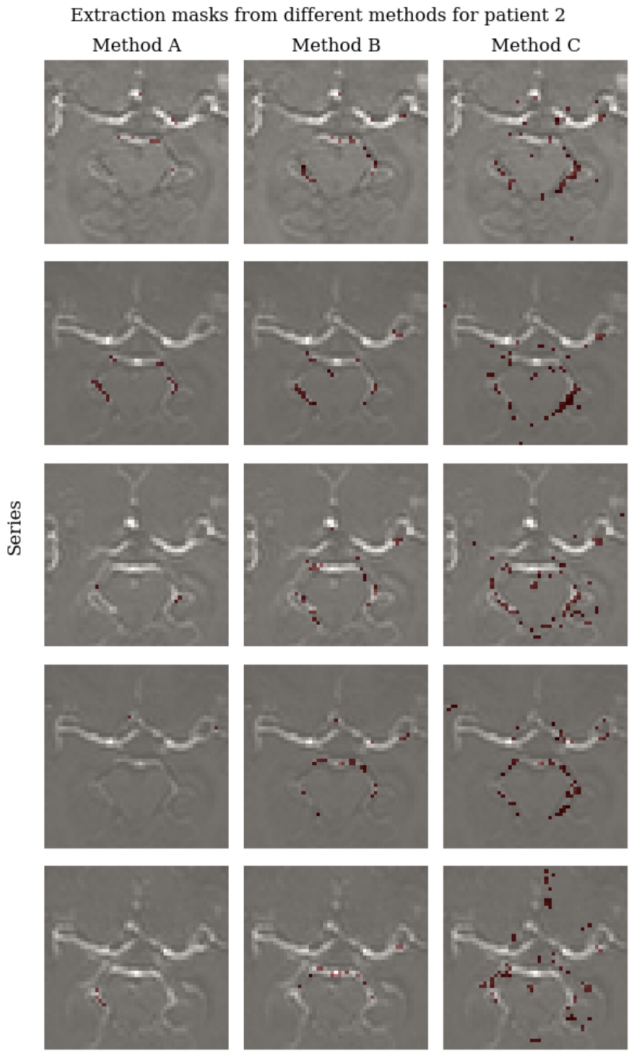

Figure 4.11: Voxels where the AIF is extracted from patient 2. Each row is the same series

There is a clear connection between the consistency of where the AIFs are extracted from and the difference between the resulting AIFs for each series. Since method C has access to more slices to select AIFs from, it also yields many more selected voxels than methods A and B.

Discussion

This thesis is motivated by the need for a reproducible and automatic AIF extraction technique. The work imitated a physician's workflow, placing an ROI approximately on the slices containing an MCA and then using a widely used clustering algorithm to extract the AIFs. A deep neural network was used to select slices with a visible MCA structure to automate the process. This also ensured transparency and simplicity if the method is to be implemented in a clinical environment. An extensive literature review had to be done on previous techniques to understand earlier work in the field. Many had a "black box" problem or were hard but impossible to implement. The learning curve has been steep both in learning about the field and implementing a solution. Most noticeably, the large amounts of data yielded significant obstacles for the model definition and training as the training setup had limited memory and computing resources.

5.1 MCA detection method

Only the Densenet model was implemented and tested, while other models like ResNet, ImageTransformers, and a custom model were not explored due to their higher data requirements and the size limitations of ResNet. It is uncertain whether these models would outperform DenseNet.
The preprocessing step before training the model involved cropping the images to retain only 70% of the original image. This approach yielded good results for the provided data, assuming a visible MCA in the center of the image. However, if images with a different field of view, such as those capturing only the cancer, are used, the detection will not work. The same limitation applies to images taken along the sagittal and frontal planes. The model has been trained on heterogeneous data, including sequences from different

machines, resolutions, and patients with or without visible brain abnormalities, to make it useable in different clinical environments. However, the model has not been trained on data from the wide varieties of machines, sequences, or abnormalities, that it could be exposed to if deployed. Therefore, the reported performance in this thesis is likely higher than what can be expected in a real-world production environment.
Lastly, the model required labeled data, but no experienced clinician was available to label it. The labeling of the data has proven to be a challenge for the model to achieve optimal performance, as evident from the results where at least one contradiction can be seen.

5.1.1 Considerations of results

The resulting MCA detection model has a reasonable performance when used on the unseen testing data. Its precision is below 90%, meaning that it will falsely label a slice not containing an MCA to contain an MCA more than 10% of the time. As method A and B relies on correctly identified MCA slices, this is not desirable. Yet, with the good performance of method C using all slices, method B would most likely not be significantly impacted by a single misclassification.
From the training accuracy, precision, and recall in Figure 4.3, it is evident that the model has not achieved a perfect recall and precision score on the testing data. This is not the case for the training data. Such a difference might indicate an overfitting of the training data or, as previously stated, errors in the MCA labeling. An example can be seen in Figure 4.5. At the bottom, two almost identical slices are labeled differently.

5.2 AIF extractions method

While there are several different methods in literature to extract AIFs from one or more selected slices, only the k-means approach by Mouridsen et al. is tested [1]. This is due to the easy implementation and long history of use. In some papers, the k-means algorithm performs worse than more modern methods like Ncut[22] and HIER[21]. Both methods rely on a selected slice or ROI. Thus, it would be interesting to see what results the application of these methods would produce.
Another part of the method is to use the clustering algorithm on the cropped volumes, similar to the HIER method [21]. This leaves only a reduced amount of voxels to search on and might have impacted the results in various ways. For example, it might have

reduced the number of voxels from where the resulting AIF is averaged. This can be seen in Figure 4.10, where only a single voxel was selected with method A.
The method by Mouridsen et al. was originally developed to be used on a slice of the whole brain and had its parameters (percentage of low-AUC curves and high irregularity curves to discard) tuned to such cases. Since this method is clustering on an ROI, other kinds of parameters might be more optimal.
It must be noted that the implementation of the AIF extraction, although based on a well-established method, has not been validated. This is likely a source of error.

5.2.1 Considerations of results

From Figure 4.6, it is visible that method B produced the least varied metrics with the exception of three outliers. This means that the method has the most consistent inter-patient variability. It might be a good indication that the method has the lowest variability overall.
Further, both models A and B have a lower average TTP than method C and a lower standard deviation. The difference in the mean TTP between methods B and C is 2.79 s. This means that method C selects AIFs with a later bolus arrival and that the extracted AIFs might originate from smaller arteries or veins.
Analyzing the mean differences between the following series for each patient, method C has the lowest mean differences and lowest standard deviations. The means for methods A and B are most likely highly impacted by their outliers. Excepting the outliers, method B seems to have the lowest variance both for the extracted values and the differences. For the correlation between the series, method B yields the highest mean correlation, but method C has the highest median, although with more outliers and a larger variance.
Looking at the selected voxels by each method, it is clear that method C selects its voxels more consistently from the same places and more total voxels. This might have been the reason why it had no outliers. Still, it is evident from the masks that method C rarely selected voxels from the MCA.

5.3 Thesis limitations

As this work presents how an automatic extraction pipeline can be made in the simplest terms, it has some drawbacks.
The patients used in the test set are imaged while undergoing brain cancer treatment.

In turn, it is expected that the vascular properties will change. Analyzing the difference in AIFs between the imaging series is troublesome, as the AIF depends on the vascular structure. It is also assumed that the imaging protocol has been the same for all series of each patient. Changes in the imaging protocol between two series will impact the result and is a source of error.
Further, the intra-patient analysis does not tell whether one method is better for distinguishing cancer or other pathologies than the others. If no intra-patient variance was the goal, a population-averaged curve [16] would yield zero difference between the series.
With this in mind, how the extracted AIFs impact the resulting biomarkers produced by PK modeling could be investigated. A within-subject analysis as described by the QIBA recommendations [13] would indicate if the extracted AIFs were at all useable for clinical applications.

5.4 Further work

This thesis shows how a simple and transparent AIF pipeline can be made to work automatically. Still, as the results show, it is not a finished work.
To continue the development of a reproducible and repeatable automatic AIF extraction pipeline, it would be beneficial to investigate how different AIF extraction methods work with the different slice selection methods described in this thesis. A proper analysis, including PK modeling and analysis according to the recommendations provided by QIBA, would help decide if the AIFs could be used in clinical practice.

Further, the impact of automatically placing an ROI should be investigated further. Only using a static ROI, without MCA detection, could show itself to improve AIF selection as a simple implementation. Thus, an empirical study evaluating the boundaries of the M1 part of the MCA could be used to extract general cropping aspects.
Instead of having a binary MCA detection model, a localization model could be made to place an ROI on the predicted MCA. This could make the static cropping obsolete and, hopefully, produce more precise AIF extractions.

Lastly, as the dataset with patients imaged over time provided to this thesis only had a few samples where the MCA were visible, a dataset should be made of patients imaged over time where the MCA is visible. Only testing the methods on 5 patients yields biased results, and the amount should be greatly increased.

5.5 Conclusions

In this thesis, an automatic MCA detection model has been developed and used together with a k-means clustering algorithm to automatically extract AIFs from DCE-MRI sequences of the human brain. Three different variations of using the slices predicted by the model, methods A, B, and C, were tested. Method C did not use the model at all. They were tested on the repeatability of extractions from patients that had been imaged several times.

The MCA detection model had an accuracy, precision, and recall that should be improved if deployed.

Method C showed the lowest average difference between the series of each patient. Method B and C produced some outliers but had overall a lower TTP than method C. This indicated that their AIFs more likely originate from large arteries than the AIFs produced with method C.

With the exception of a few outliers, the differences between the series for method B had the lowest variance. Method B also had the lowest variance of the AIFs extracted for all patients. Although promising, the pipeline and method should be trained and evaluated on more diverse data.

List of Acronyms and Abbreviations

AIF Arterial Input Function.

ANN Artificial Neural Network.

AUC Area Under the Curve.

BBB Blood Brain Barrier.

CA Contrast agent.

DCE Dynamic Contrast Enhanced.

EES Extravascular Extracellular Space.

FCN Fully Connected Neural Network.

Gd Gadolinium.

ICA Internal Carotid Artery.

MCA Middle Cerebral Artery.

ML Machine Learning.

MLP Multilayered Perceptron.

MRI Magnetic Resonance Imaging.

PK Pharmacokinetical.

PTB Peak To Baseline.

PVE Partial Volume Effects.

QIBA Quantitative Imaging Biomarkers Alliance.

ROI Region of interest.

SNR Signal-to-Noise-Ratio.

TTP Time To Peak.

Bibliography

[1] K. Mouridsen, S. Christensen, L. Gyldensted, and L. Østergaard, "Automatic selection of arterial input function using cluster analysis," *Magnetic Resonance in Medicine*, vol. 55, no. 3, pp. 524–531, 2006.

[2] D. B. Plewes and W. Kucharczyk, "Physics of MRI: A primer," *Journal of Magnetic Resonance Imaging*, vol. 35, no. 5, pp. 1038–1054, may 2012. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/jmri.23642https://onlinelibrary.wiley.com/doi/abs/10.1002/jmri.23642https://onlinelibrary.wiley.com/doi/10.1002/jmri.23642

[3] D. C. Preston. (2006) Magnetic resonance imaging (mri) of the brain and spine: Basics. [Online]. Available: https://case.edu/med/neurology/NR/MRI%20Basics.htm

[4] F. Calamante, "Arterial input function in perfusion MRI: A comprehensive review," *Progress in Nuclear Magnetic Resonance Spectroscopy*, vol. 74, pp. 1–32, oct 2013.

[5] R. Y. Choi, A. S. Coyner, J. Kalpathy-Cramer, M. F. Chiang, and J. Peter Campbell, "Introduction to Machine Learning, Neural Networks, and Deep Learning," *Translational Vision Science Technology*, vol. 9, no. 2, 2020. [Online]. Available: /pmc/articles/PMC7347027//pmc/articles/PMC7347027/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC7347027/

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[7] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, "Dive into deep learning," *arXiv preprint arXiv:2106.11342*, 2021.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-December, pp. 770–778, dec 2015. [Online]. Available: https://arxiv.org/abs/1512.03385v1

[9] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, 2017. [Online]. Available: https://github.com/liuzhuang13/DenseNet.

[10] P. S. Tofts, G. Brix, D. L. Buckley, J. L. Evelhoch, E. Henderson, M. V. Knopp, H. B. Larsson, T.-Y. Lee, N. A. Mayr, G. J. Parker, R. E. Port, J. Taylor, and R. M. Weisskoff, "Estimating Kinetic Parameters From Dynamic Contrast-Enhanced T 1-Weighted MRI of a Diffusable Tracer: Standardized Quantities and Symbols," Tech. Rep., 1999.

[11] G. P. Krestin, W. Steinbrich, and G. Friedmann, "Adrenal masses: evaluation with fast gradient-echo MR imaging and Gd-DTPA-enhanced dynamic studies," *Radiology*, vol. 171, no. 3, pp. 675–680, 1989.

[12] D. R. Hadizadeh, B. Mädler, J. Gieseke, R. Fimmers, E. Hattingen, and H. H. Schild, "Effects of arterial input function selection on kinetic parameters in brain dynamic contrast-enhanced MRI," *Magnetic Resonance Imaging*, vol. 40, pp. 83–90, jul 2017.

[13] A. Shukla-Dave, N. A. Obuchowski, T. L. Chenevert, S. Jambawalikar, L. H. Schwartz, D. Malyarenko, W. Huang, S. M. Noworolski, R. J. Young, M. S. Shiroishi, H. Kim, C. Coolens, H. Laue, C. Chung, M. Rosen, M. Boss, and E. F. Jackson, "Quantitative imaging biomarkers alliance (QIBA) recommendations for improved precision of DWI and DCE-MRI derived biomarkers in multicenter oncology trials," *Journal of Magnetic Resonance Imaging*, vol. 49, no. 7, pp. e101–e121, jun 2019. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/jmri.26518https://onlinelibrary.wiley.com/doi/abs/10.1002/jmri.26518https://onlinelibrary.wiley.com/doi/10.1002/jmri.26518

[14] G. Brix, F. Kiessling, R. Lucht, S. Darai, K. Wasser, S. Delorme, and J. Griebel, "Microcirculation and microvasculature in breast tumors: Pharmacokinetic analysis of dynamic MR image series," *Magnetic Resonance in Medicine*, vol. 52, no. 2, pp. 420–429, 2004.

[15] P. S. Tofts, "Modeling tracer kinetics in dynamic gd-dtpa mr imaging," *Journal of*

Magnetic Resonance Imaging, vol. 7, no. 1, pp. 91–101, 1997. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/jmri.1880070113

[16] G. J. Parker, C. Roberts, A. Macdonald, G. A. Buonaccorsi, S. Cheung, D. L. Buckley, A. Jackson, Y. Watson, K. Davies, and G. C. Jayson, "Experimentally-derived functional form for a population-averaged high-temporal-resolution arterial input function for dynamic contrast-enhanced MRI," *Magnetic Resonance in Medicine*, vol. 56, no. 5, pp. 993–1000, nov 2006. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/mrm.21066https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.21066https://onlinelibrary.wiley.com/doi/10.1002/mrm.21066

[17] "Operator dependency of arterial input function in dynamic contrast-enhanced MRI," *Magnetic Resonance Materials in Physics, Biology and Medicine*, vol. 35, no. 1, pp. 105–112, feb 2022. [Online]. Available: https://link.springer.com/article/10.1007/s10334-021-00926-z

[18] M. N. Gwilliam, D. J. Collins, M. O. Leach, and M. R. Orton, "Quantifying MRI T 1 relaxation in flowing blood: implications for arterial input function measurement in DCE-MRI," Tech. Rep., 2021.

[19] R. T. Woodall, P. Sahoo, Y. Cui, B. T. Chen, M. S. Shiroishi, C. Lavini, P. Frankel, M. Gutova, C. E. Brown, J. M. Munson, and R. C. Rockne, "Repeatability of tumor perfusion kinetics from dynamic contrast-enhanced MRI in glioblastoma," *Neuro-oncology Advances*, vol. 3, no. 1, jan 2021. [Online]. Available: /pmc/articles/PMC8715899//pmc/articles/PMC8715899/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC8715899/

[20] K. Murase, K. Kikuchi, H. Miki, T. Shimizu, and J. Ikezoe, "Determination of arterial input function using fuzzy clustering for quantification of cerebral blood flow with dynamic susceptibility contrast-enhanced MR imaging," *Journal of Magnetic Resonance Imaging*, vol. 13, no. 5, pp. 797–806, may 2001. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/jmri.1111https://onlinelibrary.wiley.com/doi/abs/10.1002/jmri.1111https://onlinelibrary.wiley.com/doi/10.1002/jmri.1111

[21] D. Peruzzo, A. Bertoldo, F. Zanderigo, and C. Cobelli, "Automatic selection of arterial input function on dynamic contrast-enhanced MR images," *Computer Methods and Programs in Biomedicine*, vol. 104, no. 3, pp. e148–e157, dec 2011.

[22] J. Yin, H. Sun, J. Yang, and Q. Guo, "Automated detection of the arterial input function using normalized cut clustering to determine cerebral perfusion by dynamic susceptibility contrast-magnetic resonance imaging," *Journal of Magnetic Resonance Imaging*, vol. 41, no. 4, pp. 1071–1078, apr 2015.

[23] B. M. ter Haar Romenij, Q. Yang, W. Djj, S. Fan, Y. Bian, E. Wang, Y. Kang, D. J. J Wang, and X. Ji, "An Automatic Estimation of Arterial Input Function Based on Multi-Stream 3D CNN," *Front. Neuroinform*, vol. 13, p. 49, 2019. [Online]. Available: www.frontiersin.org

[24] J. Nalepa, P. Ribalta Lorenzo, M. Marcinkiewicz, B. Bobek-Billewicz, P. Wawrzyniak, M. Walczak, M. Kawulok, W. Dudzik, K. Kotowski, I. Burda, B. Machura, G. Mrukwa, P. Ulrych, and M. P. Hayball, "Fully-automated deep learning-powered system for DCE-MRI analysis of brain tumors," *Artificial Intelligence in Medicine*, vol. 102, p. 101769, jan 2020.

[25] C. Tönnes, S. Janssen, A. K. Golla, T. Uhrig, K. Chung, L. R. Schad, and F. G. Zöllner, "Deterministic arterial input function selection in dce-mri for automation of quantitative perfusion calculation of colorectal cancer," *Magnetic Resonance Imaging*, vol. 75, pp. 116–123, 1 2021.

[26] E. de la Rosa, D. M. Sima, B. Menze, J. S. Kirschke, and D. Robben, "AIFNet: Automatic vascular function estimation for perfusion analysis using deep learning," *Medical Image Analysis*, vol. 74, p. 102211, dec 2021.

[27] A. Winder, C. D. D'esterre, B. K. Menon, J. Fiehler, and N. D. Forkert, "Automatic arterial input function selection in CT and MR perfusion datasets using deep convolutional neural networks," 2020. [Online]. Available: https://doi.org/10.1002/mp.14351

[28] E. M. Purcell, H. C. Torrey, and R. V. Pound, "Resonance absorption by nuclear magnetic moments in a solid," *Phys. Rev.*, vol. 69, pp. 37–38, Jan 1946. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRev.69.37

[29] F. Bloch, "Nuclear induction," *Phys. Rev.*, vol. 70, pp. 460–474, Oct 1946. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRev.70.460

[30] M. Levitt, *Spin Dynamics: Basics of Nuclear Magnetic Resonance*. Wiley, 2013. [Online]. Available: https://books.google.no/books?id=bysFAa4MPQcC

[31] L. G. Odeblad E., "Some preliminary observations on the proton magnetic resonance in biologic samples," *Acta Radiologica*, vol. 43:6, pp. 469–476, 1955.

[32] S. L. Thrower, K. A. Al Feghali, D. Luo, I. Paddick, P. Hou, T. Briere, J. Li, M. F. McAleer, S. L. McGovern, K. D. Woodhouse, D. N. Yeboa, K. K. Brock, and C. Chung, "The Effect of Slice Thickness on Contours of Brain Metastases for Stereotactic Radiosurgery," *Advances in Radiation Oncology*, vol. 6, no. 4, jul 2021.

[33] Y. D. Xiao, R. Paudel, J. Liu, C. Ma, Z. S. Zhang, and S. K. Zhou, "MRI contrast agents: Classification and application (Review)," *International Journal of Molecular Medicine*, vol. 38, no. 5, pp. 1319–1326, nov 2016. [Online]. Available: http://www.spandidos-publications.com/10.3892/ijmm.2016.2744/abstracthttps://www.spandidos-publications.com/10.3892/ijmm.2016.2744

[34] A. Bjørnerud, "The Physics of Magnetic Resonance Imaging Department of Physics University of Oslo," *Compendium*, no. March, 2006.

[35] D. Hanahan and R. A. Weinberg, "The Hallmarks of Cancer," *Cell*, vol. 100, no. 1, pp. 57–70, jan 2000. [Online]. Available: http://www.cell.com/article/S0092867400816839/fulltexthttp://www.cell.com/article/S0092867400816839/abstracthttps://www.cell.com/cell/abstract/S0092-8674(00)81683-9

[36] J. A. Detre, J. S. Leigh, D. S. Williams, and A. P. Koretsky, "Perfusion imaging," *Magnetic Resonance in Medicine*, vol. 23, no. 1, pp. 37–45, jan 1992. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/mrm.1910230106https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.1910230106https://onlinelibrary.wiley.com/doi/10.1002/mrm.1910230106

[37] A. Villringer, B. R. Rosen, J. W. Belliveau, J. L. Ackerman, R. B. Lauffer, R. B. Buxton, Y. hao, V. J. Wedeenand, and T. J. Brady, "Dynamic imaging with lanthanide chelates in normal brain: Contrast due to magnetic susceptibility effects," *Magnetic Resonance in Medicine*, vol. 6, no. 2, pp. 164–174, feb 1988. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/mrm.1910060205https://onlinelibrary.wiley.com/doi/abs/10.1002/mrm.1910060205https://onlinelibrary.wiley.com/doi/10.1002/mrm.1910060205

[38] R. M. Mann, C. K. Kuhl, and L. Moy, "Contrast-enhanced MRI for breast cancer screening," *Journal of Magnetic Resonance Imaging*, vol. 50, no. 2, pp. 377–390, aug 2019.

[39] F. Khalifa, A. Soliman, A. El-Baz, M. Abou El-Ghar, T. El-Diasty, G. Gimel'Farb, R. Ouseph, and A. C. Dwyer, "Models and methods for analyzing DCE-MRI: A review," *Medical Physics*, vol. 41, no. 12, dec 2014.

[40] D. Lewis, X. Zhu, D. J. Coope, S. Zhao, A. T. King, T. Cootes, A. Jackson, and K.-l. Li, "Surrogate vascular input function measurements from the superior sagittal sinus are repeatable and provide tissue-validated kinetic parameters in brain DCE-MRI," *Scientific Reports*, vol. 12, no. 1, dec 2022.

[41] T. Koopman, R. M. Martens, C. Lavini, M. Yaqub, J. A. Castelijns, R. Boellaard, and J. T. Marcus, "Repeatability of arterial input functions and kinetic parameters in muscle obtained by dynamic contrast enhanced MR imaging of the head and neck," *Magnetic Resonance Imaging*, vol. 68, pp. 1–8, may 2020.

[42] A. S. Lundervold and A. Lundervold, "An overview of deep learning in medical imaging focusing on MRI," *Zeitschrift für Medizinische Physik*, vol. 29, no. 2, pp. 102–127, may 2019.

[43] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[44] A. Farseev. (2023) Is bigger better? why the chatgpt vs. gpt-3 vs. gpt-4 'battle' is just a family chat. [Online]. Available: https://www.forbes.com/sites/forbestechcouncil/2023/02/17/is-bigger-better-why-the-chatgpt-vs-gpt-3-vs-gpt-4-battle-is-just-a-family-chat/?sh=58403b315b65

[45] M. Lin, Q. Chen, and S. Yan, "Network In Network," *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, dec 2013. [Online]. Available: https://arxiv.org/abs/1312.4400v3

[46] G. Pleiss, D. Chen, G. Huang, T. Li, L. van der Maaten, and K. Q. Weinberger, "Memory-Efficient Implementation of DenseNets," jul 2017. [Online]. Available: https://arxiv.org/abs/1707.06990v1

[47] C. Larsson, M. Kleppestø, I. Grothe, J. Vardal, and A. Bjørnerud, "T1 in high-grade glioma and the influence of different measurement strategies on parameter estimations in DCE-MRI," *Journal of Magnetic Resonance Imaging*, vol. 42, no. 1, pp. 97–104, jul 2015. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/jmri.24772https://onlinelibrary.wiley.com/doi/abs/10.1002/jmri.24772https://onlinelibrary.wiley.com/doi/10.1002/jmri.24772

[48] Siemens. (2023) Siemens skyra 3t magnetom webpage. [Online]. Available: https://www.siemens-healthineers.com/magnetic-resonance-imaging/3t-mri-scanner/magnetom-skyra#06774724

[49] Pytorch. (2023) Pytorch github repo for densenet implementations. [Online]. Available: https://github.com/pytorch/vision/blob/c06d52b1c5f6aee36802661c3ebc6347b97cc59e/torchvision/models/densenet.py

[50] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J. E. Gonzalez, and I. Stoica, "Tune: A research platform for distributed model selection and training," *arXiv preprint arXiv:1807.05118*, 2018.

[51] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, NIPS 2011*, pp. 1–9, 2011.

[52] L. Li, K. Jamieson, A. Rostamizadeh, E. Gonina, M. Hardt, B. Recht, and A. Talwalkar, "A System for Massively Parallel Hyperparameter Tuning," *CoRR*, vol. abs/1810.0, oct 2018. [Online]. Available: http://arxiv.org/abs/1810.05934

Appendix A

Further data

Table A.1: The mean and the standard deviation for the extracted AIF metrics across the series for each patient.

		Mean values for series		
Method	**Patient**	**AUC**$[\frac{1}{s}]$	**TTP** $[s]$	**PTB** $[\frac{1}{s}]$
Single slice				
	1	160.84 (54.35)	33.86 (1.48)	7.60 (2.02)
	2	144.02 (40.53)	31.94 (1.40)	6.67 (1.77)
	3	201.78 (41.07)	38.19 (3.13)	7.72 (1.77)
	4	387.52 (56.53)	28.74 (1.09)	9.14 (1.81)
	5	440.10 (229.34)	32.17 (3.01)	15.56 (6.29)
All predicted slices				
	1	158.66 (31.92)	32.12 (1.01)	6.87 (1.09)
	2	172.20 (7.12)	31.25 (0.02)	6.92 (0.59)
	3	218.59 (34.23)	38.19 (2.20)	8.33 (1.91)
	4	289.05 (64.54)	28.74 (1.09)	6.83 (1.34)
	5	403.97 (205.90)	33.03 (2.94)	13.40 (6.31)
All slices in volume				
	1	251.82 (37.53)	36.46 (2.99)	8.79 (1.30)
	2	152.20 (15.06)	31.94 (1.40)	5.40 (0.95)
	3	210.60 (23.45)	40.97 (3.42)	7.91 (0.70)
	4	431.24 (48.85)	31.74 (1.64)	10.63 (1.47)
	5	378.83 (49.12)	36.46 (1.39)	11.26 (1.64)

Table A.2: The average difference and standard deviation per patient for the AIFs extracted with the different methods. The average correlation between the series for each patient is also provided, with all correlations having a p-value < 0.01. The best average values are in bold, except for the TTP.

		Mean difference between series			
Method	**Patient**	**AUC** $[\frac{1}{s}]$	**TTP** $[s]$	**PTB** $[\frac{1}{s}]$	**Correlation**
A					
	1	-47.60 (36.07)	1.15 (1.60)	-1.73 (0.47)	0.92 (0.08)
	2	6.79 (74.29)	0.00 (2.47)	0.42 (3.15)	0.86 (0.10)
	3	-11.67 (63.20)	-0.02 (4.29)	-0.72 (2.34)	0.91 (0.07)
	4	-17.74 (88.41)	0.00 (2.22)	**-0.35** (0.29)	0.92 (0.03)
	5	8.40 (379.43)	2.15 (2.59)	**0.08** (9.94)	0.87 (0.11)
B					
	1	-16.77 (61.26)	1.15 (1.65)	-0.63 (1.87)	**0.93** (0.02)
	2	**0.51** (12.49)	0.00 (0.02)	**0** (0.93)	**0.97** (0.01)
	3	17.17 (56.49)	0.85 (2.87)	0.80 (3.19)	**0.94** (0.03)
	4	**0.19** (108.03)	0.00 (2.22)	-0.53 (1.65)	0.92 (0.03)
	5	131.22 (215.04)	1.61 (2.88)	3.48 (6.87)	0.85 (0.12)
C					
	1	**11.85** (58.43)	2.31 (3.23)	**0.17** (2.10)	0.81 (0.25)
	2	-8.88 (19.50)	0.00 (2.47)	-0.61 (1.29)	0.95 (0.02)
	3	**2.53** (32.22)	0.02 (4.94)	**0.17** (1.02)	0.90 (0.11)
	4	15.95 (61.90)	0.53 (2.87)	0.49 (1.76)	**0.96** (0.02)
	5	**5.62** (77.77)	0.54 (2.36)	0.10 (2.15)	**0.94** (0.07)

Appendix B

Implementation details

Table B.1: Hardware specification

Hardware	**Name**
CPU	Intel(R) Core(TM) i7-8700K 3.70GHz
GPU	NVIDIA GeForce GTX 1080
RAM	32 GB

Software used:

Table B.2: List of software.

Software	**Version**	Use
Windows 11 Enterprise	21H2	Operating System
Visual Studio Code	1.78.0	Integrated Development Environment
Python	3.10.9	Python compiler
Numpy	1.24.2	Data management
Pandas	1.5.3	Data management
PyTorch	1.13.0	Model definition and training
Scikit-learn	1.2.0	Model definition and metrics
Raytune	2.2.0	Hyperparameter Tuning
Pydicom	2.3.1	DICOM file management
Matplotlib	3.7.0	Visuals