Machine Learning in Automated Segmentation of Small Lesions in Magnetic Resonance Imaging for Multiple Sclerosis

Master Thesis in Medical Technology

by

Maria Mathea Køhler Aarhus

Department of Physics and Technology

University of Bergen

June 01, 2023

Scientific environment

This study was carried out at the Department of Physics and Technology at the University of Bergen, as well as Mohn Medical Imaging and Visualization Centre (MMIV), Department of Radiology, Haukeland University Hospital (HUH).

Acknowledgements

I would like to take this opportunity to express my sincere gratitude to all those who have supported and contributed to the completion of this thesis. Your assistance, guidance, and encouragement have been extremely helpful throughout this journey.

Firstly I will start to thank my supervisors, Frank Riemer and Renate Grüner, for your immense encouragement and ability to motivate. Thank you for not only sharing your extensive expertise but also for providing unwavering support and creating an environment conducive to meaningful discussions. Your guidance and mentorship have been invaluable, serving as steadfast pillars in the life of a stressed-out master's student. I also appreciate you for giving me the chance to participate in the Lund ISMRM Nordic Chapter and the MMIV conference last year. It was a great opportunity for learning and provided me with valuable feedback from prominent researchers in the field.

A special thanks to the entire team at MMIV for your warm welcome and exceptional support, surpassing my initial expectations and making the process of writing this master's thesis truly remarkable. Your kindness and assistance have greatly enhanced my experience. I would also like to thank Ellen Skorve, Astri J Lundervold, Øivind Torkildsen, and Kjell-Morten Myhr from the Department of Neurology, HUH, for providing me access to the data used in the thesis. Knowing that this work serves a practical purpose in aiding patients, has been a great motivation. Also, a huge thank you to Fulvio Zaccagna, who gave valuable insights into the multiple sclerosis disease.

Last but not least I have to thank my friends and fellow students, I would never have gotten to submit this thesis without your help. Thank you for all your support and encouragement throughout the past five years, your love and helping hands have really made a difference. Thank you to my father Jan-Tore and brother Markus, your reassurance has been irreplaceable. And thank you to Sindre for your patience and always being there for me.

Maria Mathea Køhler Aarhus

Bergen, June 2023

Abstract

Introduction: Multiple Sclerosis (MS) is the most prevalent neurological disability in young adults. It is an autoimmune-mediated disorder that involves demyelinating lesions affecting the central nervous system, with magnetic resonance imaging (MRI) as one of the most important diagnostic tools. Small MS lesions are indicative of disease progression and treatment but can be difficult to spot. The use of machine learning (ML) can greatly aid in this process and this thesis aims to test and compare different ML methods applied to structural MRI images, for this purpose.

Methods: The study employed various ML methods to automatically segment MS lesions, with a specific focus on lesions smaller than or equal to 10 mm^3. The models evaluated included the popular lesion segmentation algorithms lesion segmentation tool (LST) and nicMSlesions, as well as the neural networks U-Net and nnU-Net. The neural networks were re-purposed and trained to perform small lesion segmentation to see how they perform against established algorithms.

Results: Evaluation of the segmentations generated by each model was conducted using dice score, sensitivity, and specificity measures. Both overall lesion segmentations and segmentations exclusive to small lesions were assessed for all models. The nnU-Net exhibited the highest dice and sensitivity scores for both types of segmentations, followed by the U-Net. The LPA, LGA, and nicMSlesions ranked subsequently in descending order of scores. This trend was consistent for both overall lesion segmentations and segmentations excluding the larger lesions. The specificity values remained high across all models throughout the assessment.

Conclusion: The nnU-Net and U-Net demonstrated the ability to segment lesions smaller than or equal to 10 mm^3, suggesting the feasibility of this task for future work. Furthermore, the fully automated process of the nnU-Net emphasizes the significance of automation in achieving high segmentation scores, underlining its importance beyond initial expectations.

Contents

Abbreviations

AI	Artificial Intelligence
CNN	Convolutional Neural Network
CNS	Central Nervous System
CSF	Cerebrospinal Fluid
DL	Deep Learning
DNN	Deep Neural Network
DSC	Dice Score Coefficient
DOIT	Determination of Optimal Initial Threshold
FC	Fully Connected
FDR	False Discovery Rate
FN	False Negative
FOR	False Omission Rate
FP	False Positive
FLAIR	Fluid Attenuated Inversion Recovery
GM	Grey Matter
HUH	Haukeland University Hospital
LGA	Lesion Growth Algorithm
LPA	Lesion Prediction Algorithm
LR	Logistic Regression
LST	Lesion Segmentation Tool
MAGNIMS	Magnetic Resonance Imaging in MS
MICCAI	Medical Image Computing and Computer Assisted Intervention
ML	Machine Learning
MNI	Montreal Neurological Institute

MRI	Magnetic Resonance Imaging
MS	Multiple Sclerosis
MSE	Mean Squared Error
NMR	Nuclear Magnetic Resonance
NPV	Negative Predictive Value
PPMS	Primary Progressive MS
PRMS	Progressive Relapsing MS
PVE	Partial Volume Estimate
ReLU	Rectified Linear Unit
RF	Radio Frequency
ROI	Region of Interest
RRMS	Relapsing-Remitting MS
SGD	Stochastic Gradient Descent
SPM	Statistical Parametric Mapping
SPMS	Secondary Progressive MS
TE	Echo Time
TI	Inversion Time
TL	Transfer Learning
TN	True Negative
TP	True Positive
TR	Repetition Time
TPM	Tissue Probability Map
TLV	Total Lesions Volume
WM	White Matter

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Multiple Sclerosis (MS) is the most prevalent neurological disability in young adults. It is an autoimmune-mediated disorder that involves demyelinating lesions affecting the central nervous system (CNS) and can often lead to serious cognitive or physical issues [1]. The prevalence lies around 50–300 per 100.000 people, meaning that about 2-3 million people are estimated to live with multiple sclerosis globally [2].

The introduction of magnetic resonance imaging (MRI) revolutionized the diagnosis and treatment of MS. It allowed for the visualization of disease activity, providing valuable insights. As MRI technology continued to advance over the following decades, it became the most important diagnostic and monitoring tool available. MRI has also gained prominence as a crucial appliance in MS clinical trials and is routinely used for long-term clinical monitoring [3]. By analyzing lesions through MRI sequences, specific criteria have been established to facilitate the diagnosis of multiple sclerosis in patients who exhibit clinical symptoms commonly associated with the condition. The McDonald diagnostic criteria for multiple sclerosis, established in 2001 and revised in 2017, relies on assessing the number, size, and location of brain and spinal cord lesions as key indicators of the disease [4].

Newly formed MS lesions often appear small, meaning detecting them is an important step in early MS diagnosis. The identification of these lesions provides crucial information when evaluating the disease activity and determining the effectiveness of treatment. The emergence of new lesions is also closely associated with disease progression and severity, often accompanied by heightened symptoms. As they can be difficult to spot due to their size and analyzing MRI scans manually can be a time-consuming process, the inclusion of artificial intelligence (AI) to automatically detect and segment

MS lesions can greatly assist in monitoring the disease [5]. Research in the field of MS diagnosis using MRI modalities and deep learning (DL) architectures have been initiated since 2016. The research is comprised of the utilization of conventional machine learning (ML) methods and DL models for segmentation and classification applications [6].

1.2 Aims

The goal of this thesis is to investigate different machine learning methods, both conventional ML algorithms and DL-based models, to segment MS lesions. The focus is on finding how well these models perform on lesion segmentation in general, and small lesions in particular. In this project, small lesions have been defined as lesions smaller than or equal to 10 mm^3.

The models chosen for the project include well-established algorithms such as the lesion segmentation tool (LST), which provides two segmentation algorithms, and nicMSlesions. Both have been used as benchmark methods in MS lesion segmentation publications [7]. For the DL models the U-Net developed by Ronneberger et al. [8] and the more automated nnU-Net [9] were chosen. The U-Net because of its viability through fastMONAI [10], which is an open-source deep learning library chosen because of its accessibility to state-of-the-art DL techniques. The nnU-Net is proposed to handle a wide variety of target structures and image properties. It has produced high performance scores from international segmentation challenges [9] and is, therefore, an interesting algorithm to evaluate and compare for this type of task.

The results will be presented, evaluated, and compared using appropriate performance measures such as sensitivity, specificity, and dice score.

Chapter 2

Background

The background chapter presents the foundation for understanding the concepts and terminology relevant to the research. It delves into the fields of MRI and MS disease, as well as establishing the theory behind the algorithms and neural networks appropriate to the thesis. The current state of knowledge in the field is also briefly overviewed.

2.1 Magnetic resonance imaging

The nuclear magnetic resonance (NMR) phenomenon was first described experimentally in 1946 by Bloch and Purcell [11][12]. It was first clinically utilized in 1981, providing the physical basis for generating images through NMR [13]. Magnetic resonance imaging has emerged as a potent diagnostic tool for assessing and monitoring the treatment of patients. To do this, it utilizes magnetic fields and radio-frequency (RF) signals to generate visual representations of anatomical structures and functions [14].

2.1.1 MRI basics

All atomic nuclei are built up of protons and neutrons, which both are spin-$\frac{1}{2}$ particles. Some nuclei, like the hydrogen nucleus, ^1H, thus have a resultant net spin based on the number of nucleons present. Associated with this spin is a net magnetic moment specific to the nucleus at hand. There must be an odd number of neutrons and/or protons as both are spin-$\frac{1}{2}$ particles. This spin generates a magnetic field with north and south poles and can be compared to the analog dipole bar magnet, with the poles aligning along the axis of rotation (Figure 2.1). When applying a strong external magnetic field ($\overrightarrow{B_0}$) the nucleus will align parallel or antiparallel to $\overrightarrow{B_0}$ [15]. These two orientations correspond to two different energy states. When the spin of the spin-$\frac{1}{2}$ particles is oriented parallel to $\overrightarrow{B_0}$, it is called a low-energy state, and while oriented antiparallel to $\overrightarrow{B_0}$, it is a high-energy state. For all states of matter, the majority of spins are pointed in

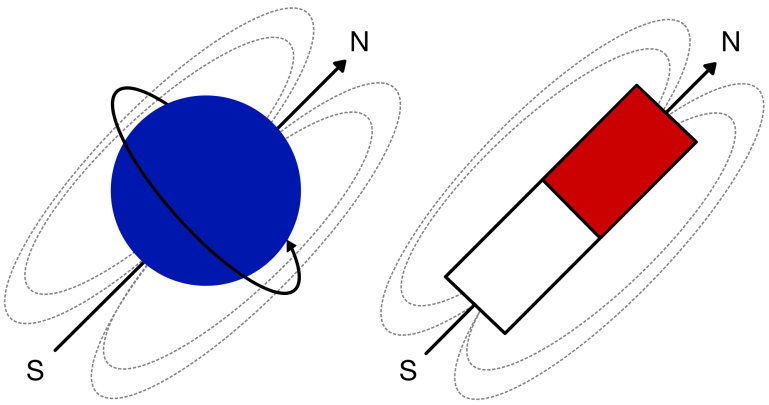


Figure 2.1: Image showing the spin of a hydrogen nucleus with a magnetic moment, to the left. The behavior is similar to that of the bar magnet shown to the right. N and S represent north and south, respectively. The directions of the straight arrows represent the direction of the magnetic field.

the same direction as $\overrightarrow{B_0}$. The magnetic momentum for these energy states is described by Equation 2.1, where the positive sign means that the momentum is pointing in the direction of the magnetic field.

$$\overrightarrow{\mu_z} = \pm\frac{1}{2}\gamma\hbar \tag{2.1}$$

When placed in an external magnetic field, say along the z-direction, the magnitude of the z-component is exactly known and can only assume discrete values. Quantum mechanically, this is because applying an external magnetic field is equivalent to making a measurement or observation in this direction [16].

During MRI, the sum of the magnetic moment of all the spin particles is observed. This is because the individual magnetic moments are too small to be measured directly. This new magnetization ($\overrightarrow{M}$) that arises within an object due to the presence of $\overrightarrow{B_0}$ is seen in equation 2.2. N_s is the number of spins being observed. Considering a coordinate system with its z-axis along $\overrightarrow{B_0}$, at rest, the net magnetization is approximately entirely in the z-direction, making $\overrightarrow{M} = \overrightarrow{M_z}$.

$$\overrightarrow{M} = \sum_{n=1}^{N_s} \overrightarrow{\mu}_n \tag{2.2}$$

Boltzmann statistics (Equation 2.3) show the ratio between the number of spins up and down when influenced by a strong magnetic field.

$$\frac{N_\uparrow}{N_\downarrow} = exp\left(\frac{\Delta E}{KT_s}\right) \tag{2.3}$$

$N_\uparrow$ is the number of spins in the same direction as $\overrightarrow{B_0}$ while $N_\downarrow$ is the number of spins opposing $\overrightarrow{B_0}$. T_s is the absolute temperature of the system in Kelvin and ΔE is the

energy difference between the two spin states, which can be seen in Equation 2.6. K is the Boltzmann constant, K = 1.38x10^{-23}J/K and $\hbar$ is Planck's constant, $\hbar$ = 6.63×10^{-34} m^2 kg/s. The two energy states $E_\uparrow$ and $E_\downarrow$ are the energies of the protons opposing and parallel to $\overrightarrow{B_0}$, respectively. They can be written as Equation 2.4 and 2.5.

$$E_\uparrow = -\frac{1}{2}\gamma\hbar\overrightarrow{B_0} \tag{2.4}$$

$$E_\downarrow = \frac{1}{2}\gamma\hbar\overrightarrow{B_0} \tag{2.5}$$

$$\Delta E = E_\downarrow - E_\uparrow = \gamma\hbar\overrightarrow{B_0} \tag{2.6}$$

Equation 2.6 shows that the difference between the energy states depends on $\overrightarrow{B_0}$. More energy is needed for a nucleus to oppose the magnetic field, and only higher-energy nuclei possess enough energy to do this. As a consequence of this, when the strength of $\overrightarrow{B_0}$ increases, fewer nuclei have enough energy to oppose the magnetic field in the antiparallel direction. In thermal equilibrium, there are always fewer high-energy nuclei than low-energy nuclei. The magnetic moments of the aligned nuclei, therefore, cancel out the magnetic moment of those opposing the field, creating the net magnetization $\overrightarrow{M}$ because of the larger number of aligned nuclei [17]. Both thermal energy and the strength of $\overrightarrow{B_0}$ can affect how many nuclei align in what direction, which is why the MRI signal depends so strongly on these variables. Low temperatures and a large magnitude of $\overrightarrow{B_0}$ make it hard for the nuclei to go to the higher energy state, thus leading to a high positive net magnetization and a strong signal. For clinical use, $\overrightarrow{B_0}$ usually has a value of 1.5T or 3T [14].

While a bar magnet would orient itself completely parallel or antiparallel to the magnetic field, a nucleus would not. Even under the influence of the strong external magnetic field, the spin of the hydrogen proton experiences a small precession around the direction of $\overrightarrow{B_0}$, giving each spin a small magnetic component in the x-y direction ($\overrightarrow{M_{xy}}$) Because of angular momentum, the nucleus adopts a gyroscopic motion (Figure 2.2). The angular frequency around $\overrightarrow{B_0}$ is called the Larmor frequency and is proportional to the field strength. This variable is described by the Larmor equation as seen in Equation 2.7. ω is the angular frequency of the protons, γ is the gyromagnetic ratio which is a constant fixed for a specific nucleus, and $\overrightarrow{B_0}$ is the field strength. The gyromagnetic ratio is denoted in MHz/T [17].

$$\omega = \gamma\overrightarrow{B_0} \tag{2.7}$$

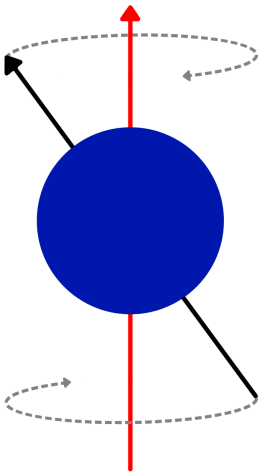


Figure 2.2: Image of how a hydrogen proton experiences gyroscopic motion around the direction of a main magnetic field.

The hydrogen atom has one of the highest gyromagnetic ratios of all the nuclei and has a value of 42.57 MHz/T [17]. Hydrogen is the most abundant atom in the human body as it appears in water and fat molecules. This is why ^1H gives the strongest MR signal and is frequently used in clinical MRI [17][16].

2.1.2 MR signal generation

In the acronym MRI, the "R" denotes resonance. Resonance in MRI occurs when a nucleus is subjected to an oscillating force that has a frequency close to the natural frequency of the nucleus (ω_n). Exposure to this oscillating force transfers energy from the external force to the nucleus [17]. If the frequency of the transferred energy matches the Larmor frequency (ω_0) of the nucleus, resonance occurs, causing the nucleus to become excited. The Larmor frequency of hydrogen lies within the radio frequency (RF) band of the electromagnetic spectrum for all B_0 field strengths used in clinical MRI. Applying an RF pulse with the same frequency as the Larmor frequency of hydrogen allows the hydrogen nuclei to resonate, resulting in an increase in nuclei with a downward spin.

Resonance serves as the fundamental basis for generating signals in MRI. One of the outcomes of resonance is that the magnetic moment rotates away from the alignment with the main magnetic field $\overrightarrow{B_0}$. The magnitude of the angle in which the $\overrightarrow{M}$ misaligns with $\overrightarrow{B_0}$ is called the flip angle. This misalignment also leads to the reduced magnitude of $\overrightarrow{M}$ in the $\overrightarrow{M_z}$ direction and increased magnitude in the $\overrightarrow{M_{xy}}$ direction [18]. The size of the flip angle depends on the amplitude and duration of the RF pulse [17]. A flip angle of 90° would fully transfer the magnetic moment into the transverse plane (x-y plane),

which is perpendicular to the main magnetic field. Another outcome of resonance is that the magnetic moments of the nuclei within $\overrightarrow{M_{xy}}$ begin to move into phase with each other. During the application of the RF pulse, all individual magnetic moments move in phase, leading to a coherent net magnetic moment that precesses in the transverse plane at the Larmor frequency [17].

Faraday's law of induction state that if a conduction coil is placed close to a rotating magnetic field, a voltage will be induced in the coil [19]. A signal is produced when in-phase magnetization cuts across the coil. As $\overrightarrow{M}$ precess in-phase in the traverse plane, the movement causes fluctuations inside this coil and also induces voltage within it. This induced voltage is what constitutes the MR signal. The frequency of the signal is consistent with the Larmor frequency, and the magnitude depends on the amount of transverse magnetization [17].

2.1.3 Relaxation

When an RF pulse is no longer applied, the nuclei give up absorbed RF energy and start to realign with $\overrightarrow{B_0}$. The dephasing is caused by individual magnetic moments releasing energy into the surrounding tissue and interaction with adjacent magnetic nuclei, leading to the occurrence of T1 recovery and T2 decay. This relaxation results in recovery of magnetization in the longitudinal plane, $\overrightarrow{M_z}$ (spin-lattice relaxation) and decay of magnetization in the transverse plane, $\overrightarrow{M_{xy}}$ (and spin-spin relaxation) [17].

T1 recovery

T1 recovery is also called spin-lattice relaxation, where "lattice" refers to the environment surrounding the nucleus. During this relaxation, the energy from the nucleus is dissipated into the surrounding environment which causes the nuclei to recover their longitudinal magnetization [17]. The recovery rate is exponential, and the T1 constant is the amount of time elapsed before 63% of the longitudinal magnetization is recovered (Equation 2.8) [16].

$$M_z(t) = M_0(1 - e^{-\frac{t}{T1}}) \tag{2.8}$$

The value of 63% becomes apparent when setting time t equal to T1 in equation 2.8, leading to equation 2.9.

$$M_z(T1) = M_0(1 - e^{-\frac{T1}{T1}}) = M_0(1 - e^{-1}) = M_0 \cdot 0.63 \tag{2.9}$$

T2 decay

T2 decay or spin-spin relaxation, expresses the process by which protons fall out of phase in the x–y plane and leads to the decay of transverse magnetization. The relaxation happens due to inhomogeneity within the surrounding tissue which causes the nuclei to exchange energy with adjacent nuclei. This happens as a result of the magnetic fields of nuclei affecting the surrounding magnetic fields [17]. The decay rate is exponential and the T2 constant is the time it takes before only 37% of the transverse magnetization is left [16].

$$M_{xy}(t) = M_0 e^{-\frac{t}{T2}} \tag{2.10}$$

The value of 37% becomes apparent when setting time t equal to T2 in equation 2.10, leading to equation 2.11.

$$M_{xy}(t) = M_0 e^{-\frac{T2}{T2}} = M_0 e^{-1} = M_0 \cdot 0.37 \tag{2.11}$$

Both T1 and T2 depend on tissue composition, structure, and surroundings. Fat is hydrogen linked to carbon and consists of large molecules. This structure allows for electrons to work as a protecting cloud, shielding the nucleus from the effects of the magnetic field. Water, on the other hand, is hydrogen linked to oxygen, which has a tendency to pull electrons away from the hydrogen nucleus. This causes the Larmor frequency of hydrogen in water to be higher than that of fat. Hydrogen in fat recovers faster along the longitudinal axis than in water, as well as the transverse magnetization decaying faster in fat than in water (Figure 2.3). These effects cause fat and water to appear in different ways in MR images [17]. Typical values in biological tissues are T1 $\approx$ 300–2000 ms and T2 $\approx$ 30–150 ms [16].

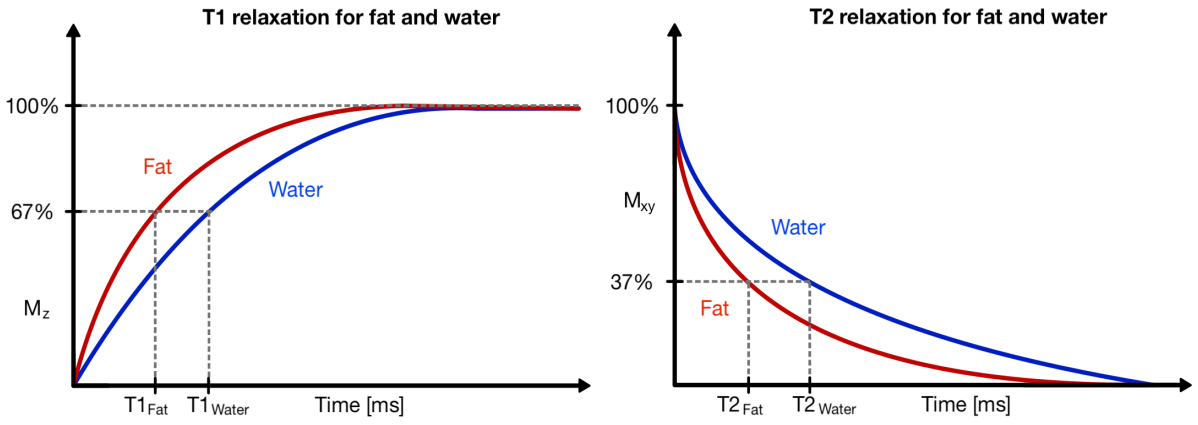


Figure 2.3: To the left is a graph showing the difference in T1 recovery for fat and water. Here the y-axis shows the longitudinal magnetization $\overrightarrow{M_z}$. To the right is the graph showing T2 decay in fat and water, the y-axis refers to the transversal magnetization $\overrightarrow{M_{xy}}$. Fat is shown in red while water is shown in blue, this is true for both graphs.

2.1.4 Image formation

Image contrast

An image has contrast if there are areas of high signal as well as areas of low signal. A high signal can be seen as areas with a large component of transverse magnetization, while the opposite is true for areas of low signal [17]. For the image contrast to be manipulated, four pulse sequence parameters are first introduced. The flip angle α, as mentioned in Section 2.1.2, is the angle between the position of the perturbed magnetization vector immediately after an RF pulse and its equilibrium position along $\overrightarrow{B_0}$. The echo time, TE, is the time delay from the excitation pulse to the signal readout. The repetition time, TR, is the time between two successive RF pulses. The last parameter is the inversion time, TI, which is the time between pulses to selectively zero out the signal contribution of one particular tissue such as fat [16]. Specific values for these sequence parameters lead to the formation of differently weighted images such as T1-weighted or T2-weighted images [17].

T1 contrast

Because the T1 time is shorter for fat than for water, hydrogen residing in fat realigns faster with $\overrightarrow{B_0}$. Thus the longitudinal component of magnetization is also larger in fat than in water. When a 90° RF pulse is applied after a certain TR the longitudinal component of magnetization for both tissues is flipped into the transverse plane. Because there is more longitudinal magnetization in fat before the RF pulse, there is more transverse magnetization in fat following the pulse. This leads to fat having a higher signal than water and appearing bright on a T1-contrast image. For the same reason, water has a low signal and therefore appears dark. This generates a T1-weighted image. The value of TR controls how much T1-recovery can occur between RF pulses and thus also controls the amount of T1-weighting. A short TR makes sure that both water and fat have not recovered all of their longitudinal magnetization before the new RF pulse is applied [17].

T2 contrast

T2-weighted images work similarly to that of the T1-weighted. T2 is shorter in fat than in water, causing the transverse magnetization component to decay faster in fat. Since water has a larger transverse component, it also gets a higher signal and appears bright on a T2-contrast image, in contrast to fat which appears dark. The TE parameter controls the amount of T2 decay that happens before the signal is received. A long TE

results in a T2-weighted image as both water and fat are given enough time to decay [17].

Fluid Attenuated Inversion Recovery

Fluid Attenuated Inversion Recovery (FLAIR) is an inversion recovery pulse sequence with long TR and TE and a TI that effectively nulls out the signal from cerebrospinal fluid (CSF) [20]. It is particularly useful in brain imaging to visualize lesions or abnormalities that the bright signal from CSF may obscure [21]. FLAIR applies a 180-degree inversion pulse to the tissue of interest, nulling the signal from fluids in the tissue. The tissue is then imaged with a standard T2-weighted sequence, highlighting the abnormal areas that may have been hidden in a conventional T2-weighted image [22]. FLAIR imaging is regarded as one of the most sensitive imaging techniques for the detection of white matter lesions, either age-related or in patients with multiple sclerosis [21].

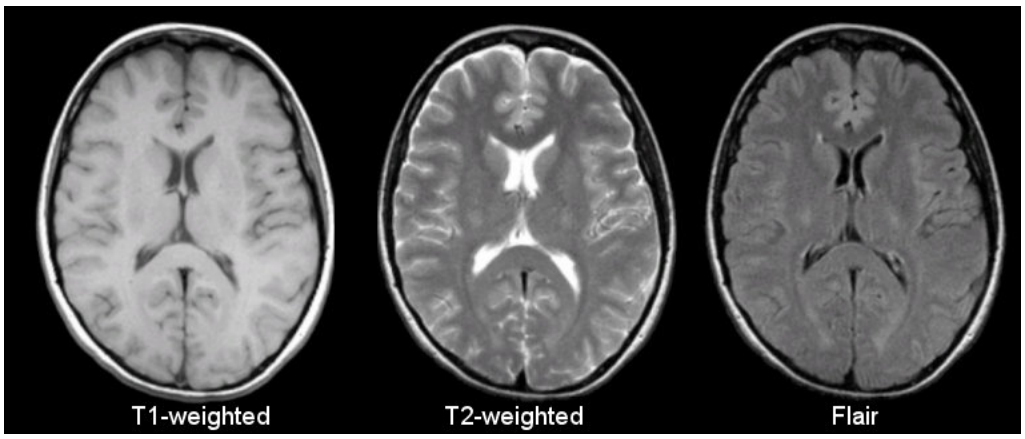


Figure 2.4: Image showing how MR images from T1-weighted, T2-weighted and FLAIR image sequences look. Image reprinted from [23].

2.2 Multiple sclerosis

Multiple Sclerosis is a chronic demyelinating disease affecting the central nervous system. The disease is one of the most common causes of neurological impairment in young adults and can lead to severe physical or cognitive disability as well as neurological defects [1]. MS is defined as the immune system attacking the nerve fibers and the myelin coating around them, resulting in inflammation (Figure 2.5). This inflammation can damage or destroy the nerve cells and myelin, altering the electrical messages in the CNS. These areas of damage caused by demyelination are called lesions. MS-related lesions appear on MRI images as light or dark spots, depending on the MRI technique used [3].

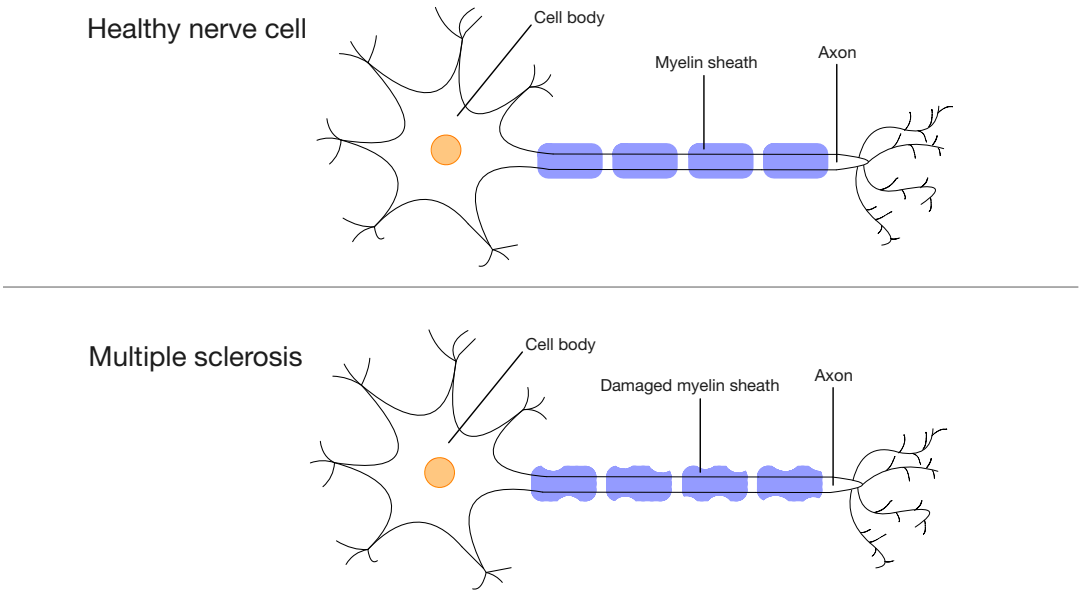


Figure 2.5: Image showing how the myelin sheath of a neuron gets damaged in patients with MS.

2.2.1 Subtypes of multiple sclerosis

Determining the type of MS correctly is essential for prognosis and treatment decisions regarding the patient. There are four subtypes to consider, which are relapsing-remitting MS (RRMS), primary progressive MS (PPMS), secondary progressive MS (SPMS), and progressive relapsing MS (PRMS).

The most common type with a prevalence of approximately 87% is RRMS, which is characterized by alternating periods of inflammatory attacks on myelin and periods of remission. The symptoms vary and may include fatigue, intestinal and urinary disorders, visual impairment, numbness, learning disability, and memory impairment. About 65% of patients with RRMS will, later on, develop SPMS; this is considered to be the second phase of the disease [1]. Only image data from patients diagnosed with RRMS were included in the current thesis.

The subtype PPMS affects approximately 10%–15% and is characterized by a worsening neurological function at the beginning of symptoms without early relapses or remission. Patients with PPMS tend to have more lesions in the spinal cord rather than the brain, leading to symptoms like stiffness, weakness, and problems with walking and balance. The least common type of MS is PRMS and occurs in approximately 5% of patients. This type has symptoms such as dizziness, depression, eye pain, and double vision [1].

2.2.2 Multiple sclerosis diagnosis

MRI is one of the most important diagnostic tools when it comes to early diagnosis of MS. This has led to the incorporation of MRI related criteria into the International Panel criteria for diagnosis (McDonald criteria), which was last revised in 2017 [24]. Guidelines have also been published by the Magnetic Resonance Imaging in MS (MAGNIMS) network in order to optimize the MS diagnosis process [25]. The guidelines include standardized MRI protocols, such as T2-weighted sequences required to be obtained in at least two planes [25].

Multiple sclerosis lesions

In multiple sclerosis, a lesion can be seen as a localized area of increased signal intensity on imaging sequences such as T2-weighted. These lesions are usually round to ovoid in shape and can vary in size from a few millimeters to more than one or two centimeters in diameter. To meet diagnostic criteria, lesions should be at least 3 mm in their long axis, and their location should also be taken into consideration. For example, a lesion smaller than 3 mm located on the floor of the fourth ventricle should still be considered abnormal because lesions and flow-related artifacts rarely occur in this location. To exclude artifacts or small hyperintensities, lesions should be visible on at least two consecutive slices, although in acquisitions with higher slice thickness, smaller lesions may be visible on a single slice [26].

In multiple sclerosis, lesions typically develop in both hemispheres, but their distribution may be mildly asymmetric in the early stages. Compared to other disorders that cause white matter lesions, multiple sclerosis lesions tend to affect specific white matter regions and the spinal cord. The involvement of these areas should be assessed to evaluate dissemination in space in patients suspected of having multiple sclerosis [26].

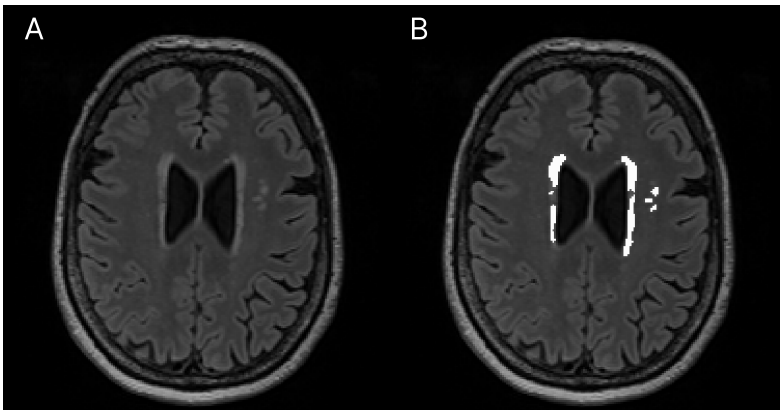


Figure 2.6: Image showing how multiple sclerosis lesions can show up on a FLAIR image (A). In the image to the right (B), the same image is shown but with the lesions colored in white. The images are from this project's dataset.

2.3 Machine learning

Machine learning is a scientific branch that focuses on computers and how they learn from data [27] [28]. It combines statistics, which seeks to understand relationships from data, and computer science, with its efficient computing algorithms [28]. Machine learning can be divided into two sub-types; supervised and unsupervised learning. Supervised learning requires a labeled dataset to predict a known target value and is often used for classifying or predicting an outcome. On the other hand, unsupervised learning does not need labeled data and can be used for analysis and clustering, among other things [29].

2.3.1 Image segmentation

Medical image analysis heavily relies on image segmentation, typically the initial and most critical step in numerous clinical applications. For example, image segmentation is frequently employed in brain MRI analysis to measure and visualize anatomical structures, analyze changes, delineate pathological regions, and plan surgeries and image-guided interventions [30]. Image segmentation aims to split an image into distinct, non-overlapping parts with semantically meaningful and homogeneous attributes such as color, texture, intensity, or depth. The outcome of this process is either an image that labels each homogeneous region or a collection of contours that describe the boundaries between the regions.

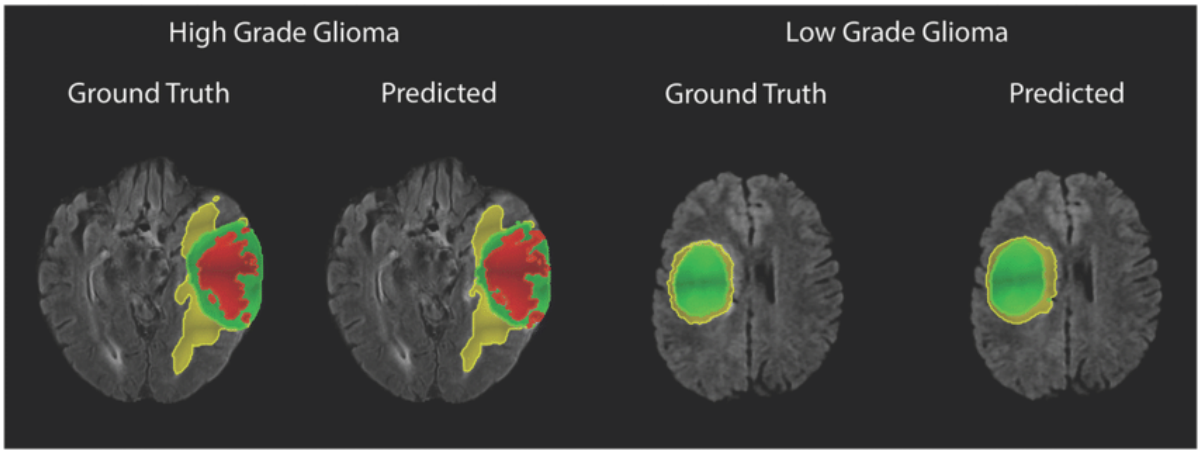


Figure 2.7: Image showing an example of how automatic segmentation can be used to outline a region of interest. Image is reprinted from Di Ieva et al. [31], a study where deep learning was applied to automatically segment brain tumors in magnetic resonance images.

As the medical image segmentation process is demanding [32], computerized methods for image segmentations have become popular to assist with this task. Automatic image segmentation is a supervised learning technique that utilizes annotated datasets to teach a model how objects in an image should be sectioned. An example can be seen in Figure 2.7. It works by assigning each pixel in an image to a label where all the pixels with the same label get assigned to the same object or element. For medical purposes, this could be to detect boundaries or outline and fill a region of interest (ROI), such as a tumor or an MS lesion [30].

2.3.2 Transfer learning

Large amounts of data are essential for most ML training algorithms to perform well [33]. This commonly causes complications for medical studies in this field because of small participant groups and lack of expert-annotated datasets [34]. One approach to this problem is applying transfer learning (TL), a method that aims to improve the performance of a model by using previously learned knowledge from similar tasks. TL can therefore aid in situations with data scarcity and save time and hardware resources [34].

2.3.3 Dataset splitting

ML algorithms are prone to overfitting (Figure 2.8) when they are trained on a particular dataset [35] [36]. Overfitting occurs when a model becomes too focused on the training data and performs poorly on new, unseen data. This happens because the model becomes overly complex and memorizes the training examples instead of learning general patterns. One way to counteract this is to split the data into smaller groups, one for training and one for testing. This ensures a way to test that the model generalizes well enough to get adequate results beyond the training sample [35]. Each segment receives a random proportion of features and their corresponding outcome from the original data to ensure variability within the sets. The training sample is usually the largest; the size of each set depends on the amount of data available. Another common way of splitting is to include a validation set [37] [38]. Validation sets are used to fine-tune model parameters and work as a testing ground for different values and approaches. This is to be able to select the model with the best performance and get a better idea of how the models work on unseen data. Not all algorithms need a validation set, but it can be a valuable tool for optimizing a model [37].

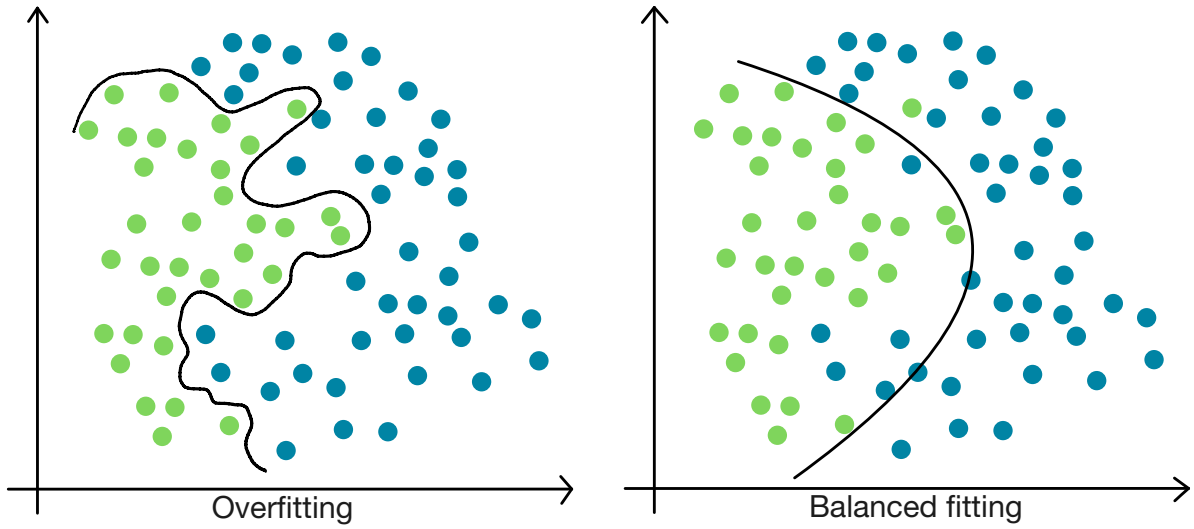


Figure 2.8: Image of an overfitted model to the left and a balanced one to the right. The overfitted model has learned all the nuances of the training set, while the balanced model is able to generalize.

2.3.4 Data augmentation

Data augmentation is another method to address the problems related to a lack of data. The technique can be applied to increase the size and diversity of a dataset and therefore works to reduce the generalization error of a model. Data augmentation applies modifications to the images in the training set to generate additional representative samples that simulate changes in patients' acquisition and anatomical variation. This additional data helps the model avoid learning features that are too specific to the original training data, leading to improved generalization and, ultimately, better performance on the test set. Another common issue when training such a model is a class imbalance, where one or more classes are under-represented in the dataset, leading to a bias towards the over-represented class. Data augmentation can thus overcome this issue by augmenting additional data from the under-represented class [39].

2.3.5 Logistic regression model

Logistic regression (LR) analysis has become an increasingly used tool for statistics in medical research in recent years [40]. LR is a machine learning classification algorithm that is used when wanting to predict the probability of different classes based on the input. The output is always between 0 and 1, where a higher value indicates a higher probability for the input as being class 1 and vice versa. This makes it a good fit for medical research as it is an appropriate model to use involving disease state (disease or healthy) and decision-making (yes or no). More complex forms of LR that can predict more than two classes are referred to as multinomial logistic regression [41]. In logistic regression, each predictor is assigned a number that represents its unique

impact on the variability of the response variable, Y. The predicted probabilities in this model are represented as the natural logarithm (ln) of the odds ratio (Equation 2.12). Here, $ln\left[\frac{P(Y)}{1-P(Y)}\right]$ is the logarithm of the outcomes, Y is either of the two outcomes, $X_1, X_2, ..., X_k$ are predictor variables, $\beta_0, \beta_1, \beta_2, ..., \beta_k$ are regression model coefficients, and β_0 is the intercept [42]. The intercept β_0 is the expected value for the logarithmic odds of the response variable occurring when all predictor variables are equal to zero.

$$ln\left[\frac{P(Y)}{1-P(Y)}\right] = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_k X_k \tag{2.12}$$

Predicting the unknown parameters in β is achieved through maximum likelihood estimation, which involves identifying the set of parameters for which the probability of the observed data is greatest. The regression coefficients indicate the strength of association between each independent variable and the outcome, reflecting the amount of change that would be expected in the response variable with a one-unit change in the predictor variable.

The ultimate objective of LR is to accurately predict the outcome class for individual cases, utilizing the most optimal model. To achieve this goal, a model is constructed that includes all the predictor variables that are relevant in predicting the response variable. LR calculates the odds of success versus the odds of failure, and the results of the analysis are presented in the form of an odds ratio [42].

2.4 Deep learning

Deep learning (DL) is considered a subcategory of ML. It differs from other ML algorithms in that it strives to mimic the human brain through a combination of weights, biases, and data inputs. Working together, these elements manage to recognize, describe and classify entities within the data accurately. DL models are often referred to as deep neural networks (DNNs) as they utilize neural network architectures that consist of several nodes and layers (Figure 2.9). While traditional neural networks contain 2-3 hidden layers, a DNN can have as many as 150. These layers help to learn the features from the data without the need for manual extraction [43]. DL can be superior to classical ML in that it often continues to improve as the size of the dataset increases. This contrasts with some ML methods that plateau at a certain level of performance when adding more training data to the network [43].

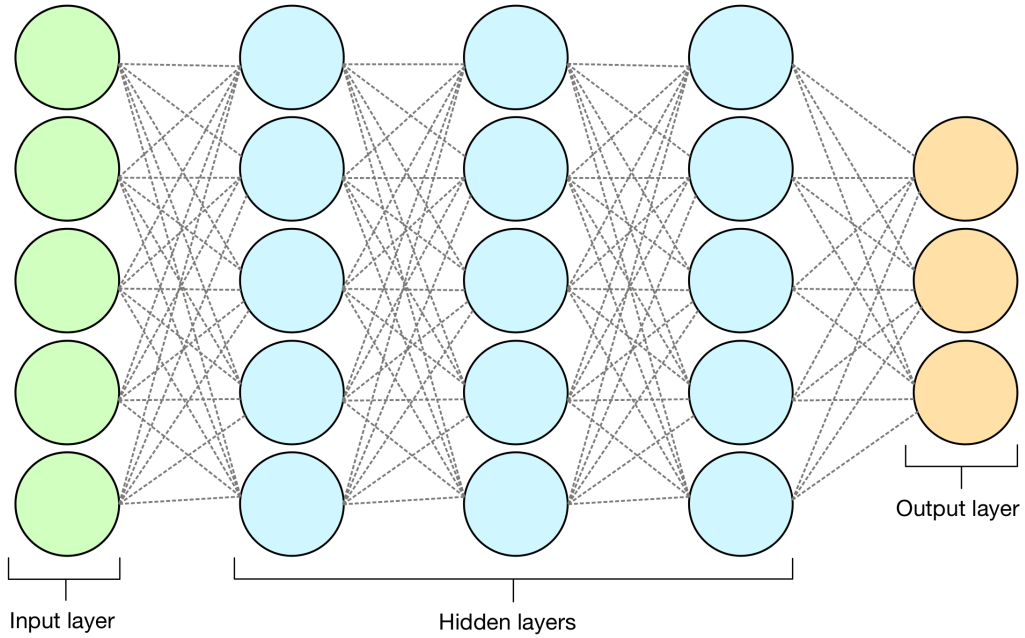


Figure 2.9: Image showing the different layers making up a neural network.

2.4.1 Convolutional neural networks

Convolutional neural networks (CNNs) are one of the most used deep neural networks, and it has made great success in medical image analysis [44] [45]. CNNs have the explicit assumption that the inputs given are image matrices and can assign significance to the different features present. To do this, the CNN architecture comprises several layers that aim to identify, extract or predict features to create new characteristics without losing features that are critical for getting a good prediction. By doing so, the model achieves feature generalization and enhances the network's ability to recognize extracted features [46].

One of the main reasons for CNNs popularity is its weight-sharing ability. This means that instead of having to train each individual parameter in the network, certain parameters are shared between different parts of the network. This reduces the total number of parameters that need to be trained, which in turn helps the network to generalize better and avoid overfitting. Another advantage of CNNs is that they are able to learn both feature extraction and classification simultaneously. This means that the model output is highly organized and efficient, as it is reliant on the extracted features. CNNs are also easier to implement on a large scale than other neural networks, which makes them more accessible and user-friendly [44].

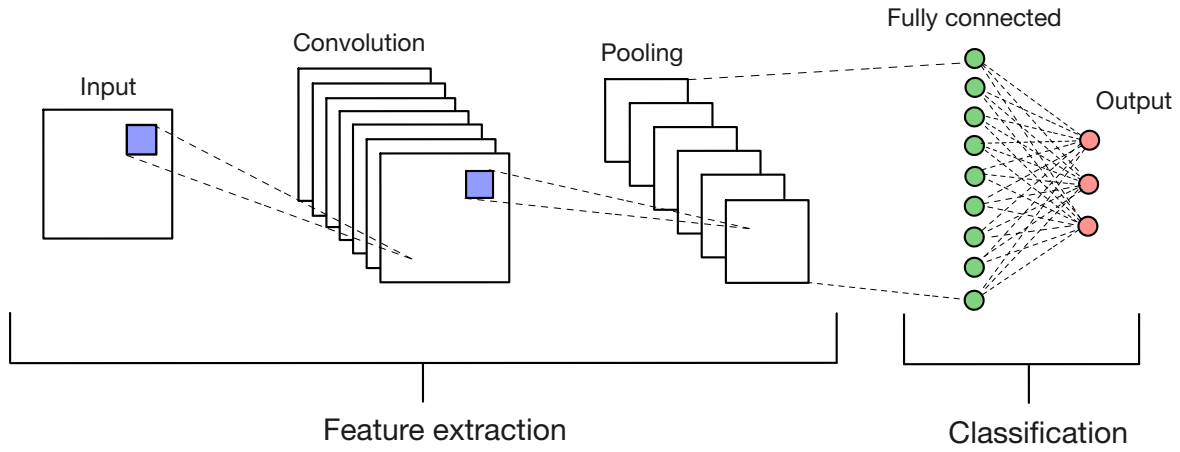


Figure 2.10: Image showing the building blocks of a classic CNN architecture made for classification. It consists of convolutional layers, pooling layers, and fully connected layers. The figure is inspired by an image in [47].

CNN architecture

The CNN architecture is made up of three different types of layers (Figure 2.10). The first layer is a convolutional layer and is used to extract features from the input. A filter of size MxM, also called a kernel (Figure 2.11), is used to traverse the input and perform the elementwise multiplication and summation between the kernel and the corresponding parts of the input based on the filter's dimensions. The output from the multiplication is a feature map that gives information, including the edges and corners at each spatial position on the image. The size of each step for the kernel is called a stride. The map is later fed forward to the other layers to learn other features of the image. Doing this also ensures that the spatial relationship between the pixels is intact.

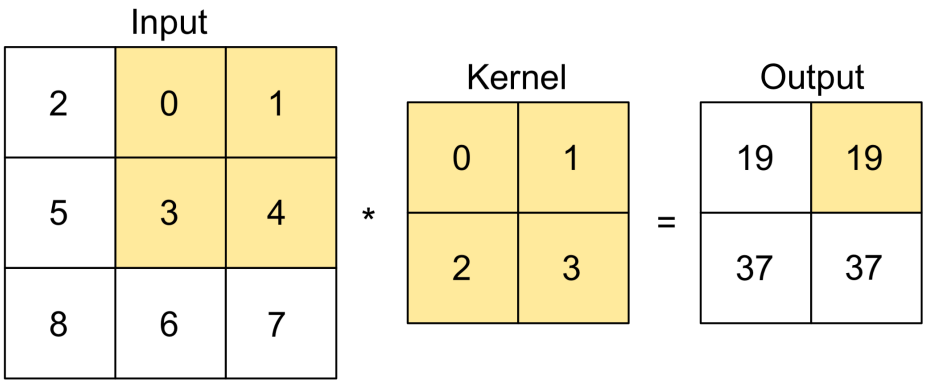


Figure 2.11: Image showing how a kernel of size 2x2 and stride of 1 is used to calculate the convolution of an input.

Following a convolutional layer is usually a pooling layer. This layer intends to reduce the computational cost by decreasing the size of the feature map. To do this, it

summarises the features produced by the convolutional layer. There are two main pooling methods, max pooling, and average pooling. The former (Figure 2.12) extracts the maximum value from the portion of the image that is covered by the kernel, while the latter returns the average value. Max pooling usually performs better than average pooling as it also works as a noise suppressant.

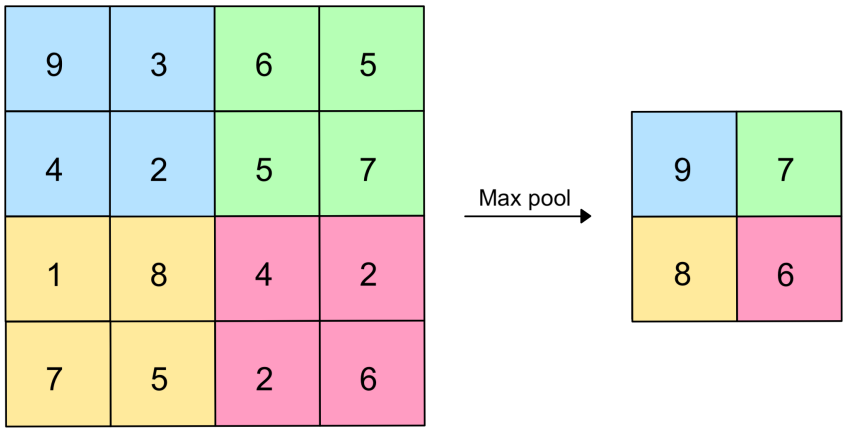


Figure 2.12: Image showing the max pooling operation with size 2x2 and stride 2.

A fully connected (FC) layer is added to learn non-linear combinations of the higher-level features from the output of the convolutional layer. The output is first flattened with a flattening layer. The flattened information is then used to predict the best label to describe the image. It consists of neurons connected to all preceding and succeeding neurons, helping it make a good mapping between input and output. Because all the features are connected to the FC layer, it can be susceptible to overfitting. To counteract this, a Dropout layer [48] can be added to drop an arbitrary number of neurons to reduce the size of the model. If dropout is set to p = 0.5, then 50% of the neurons in the network are dropped randomly.

Since images mostly contain non-linearity, activation functions are often applied to the output of a neuron to introduce non-linearity. It is a mathematical function and is critical for the ability of neural networks to model complex patterns in data. In other words, an activation function decides whether a neuron should be activated or not based on its input and in doing so, enables the network to perform more advanced and sophisticated computations. Some popular activation functions are sigmoid, tanh and ReLu. The sigmoid function maps the real value into a number between 0 and 1, tanh maps the real value into a number between -1 and 1, and ReLu maps the value based on f(x) = max(0,x).

2.4.2 Epochs

In deep learning, an epoch refers to a complete pass through the entire training dataset during the training phase of a neural network. In other words, during an epoch, the neural network processes each training example in the dataset once, adjusts its parameters based on the loss function and updates the model's weights. The number of epochs is an important hyperparameter that can affect the accuracy of the model. Generally, a higher number of epochs can lead to better accuracy, but too many epochs can cause the model to overfit to the training data, and too few can lead to underfitting. Therefore, the number of epochs can have a substantial effect on a model's performance [49].

2.4.3 Loss functions

A loss function is a mathematical function that can be used to measure the difference between the predicted output of a neural network and the actual output. The goal of the loss function is to provide feedback to the optimizer so that it can update the weights and biases of the neural network in a way that minimizes the difference between the predicted and actual output. There are various types of loss functions, each suitable for different types of problems and applications. For example, in classification tasks, the cross-entropy loss function is commonly used to measure the difference between the predicted and actual class labels. In regression tasks, the mean squared error (MSE) loss function is commonly used to measure the difference between the predicted and actual continuous values. Choosing an appropriate loss function is important for training a neural network that achieves good performance on the desired task. The selection of the loss function should be based on the specific characteristics of the problem and the type of data being used [50][51].

2.4.4 Optimizers

An optimizer is an algorithm that updates the parameters (weights and biases) of a neural network during the training process. The objective of the optimizer is to minimize the loss function, which represents the difference between the predicted output and the true output of the model. The optimizer computes the gradients of the loss function with respect to the parameters of the model. It uses them to update the parameters in a way that minimizes the loss. There are various optimization algorithms available, each with their own advantages and disadvantages, such as stochastic gradient descent (SGD), Adam, Adagrad, and RMSprop. The choice of optimizer can often have a significant impact on the training process, and the final performance of the model [52].

2.4.5 Learning rate

The learning rate is a hyperparameter that controls the step size or rate at which the optimizer updates the weights and biases of a neural network during the training process. A high learning rate can cause the optimizer to overshoot the optimal point, leading to instability and divergence of the training process, while a low learning rate can slow down the convergence of the model and increase the time required for training. Therefore, choosing an appropriate learning rate is important for achieving good performance in deep learning models. Generally, the learning rate is set through a process of trial and error, starting with a relatively high value and decreasing it gradually until the model converges to the optimal solution [53].

2.5 Algorithms and neural networks

This section aims to provide an explanation of the theory behind the various algorithms and neural networks utilized in this thesis to automatically segment multiple sclerosis lesions.

2.5.1 Lesion segmentation tool

The Lesion Segmentation Tool (LST) is an open-source toolbox for the Statistical Parametric Mapping (SPM) software [54], that is able to segment T2 hyperintense lesions in FLAIR images. Currently, there are two algorithms implemented for lesion segmentation. The first, a lesion growth algorithm [55], requires a T1 image in addition to the FLAIR image. The second algorithm, a lesion prediction algorithm [56], requires a FLAIR image only. The toolbox was developed in cooperation with multiple organizations and is maintained by Paul Schmidt [55][56].

2.5.2 Lesions growth algorithm

Lesion segmentation by the lesion growth algorithm (LGA) [55] is implemented in the LST toolbox version 3.0.0 for SPM. It is an unsupervised lesion segmentation algorithm and consists of three major steps (Figure 2.13).

Preprocessing

The algorithms perform preprocessing through SPM8/SPM12 and its voxel-based morphometry toolbox (VBM8). To prevent smoothing of the images, the algorithm operates in the space of the original T1-weighted image, also called native space, exclusively.

For tissue classification, the program uses T1-weighted images to estimate a partial volume estimate (PVE) label in native space. Every voxel in this image is then attributed to one of the three integers 1, 2, and 3, representing CSF, grey matter (GM), and white matter (WM), respectively. Numbers in between indicate a partial volume effect, where more than one tissue type occurs in a voxel [57].

Before being coregistered to the T1-weighted image, the FLAIR image is bias-corrected for MRI field inhomogeneity. Because the probability of each voxel belonging to WM, which is where lesions are commonly found, is important for identifying lesions later on, the SPM tissue probability map of white matter (TPM_{WM}) is adjusted to match the T1-weighted image space. The algorithm does this by applying the inverse deformation matrix from the PVE label estimation [55].

Lesion belief maps

After preprocessing, the algorithm calculates the FLAIR intensity distributions for each of the three tissue classes. The distribution is found using the PVE image. This is achieved to detect any FLAIR-hyperintense outliers, which would be considered a lesion voxel. Each voxel, i, is attributed a label z_i based on the estimated PVE label x_i (Equation 2.13). This is done to obtain lesion belief maps of the three different tissues [55].

$$z_i = \begin{cases} CSF, & \text{if } x_i < 1.5 \\ GM, & \text{if } 1.5 \leq x_i < 2.5 \\ WM, & \text{if } x_i \geq 2.5 \end{cases} \tag{2.13}$$

The generation of belief maps to aid in lesion identification involves combining voxel values that provide evidence for the presence of white matter lesions. The lesion belief maps are obtained for each tissue class. The B_{GM} map generates voxel values that increase if there is a high probability of the voxel belonging to WM based on spatial location, medium T1 intensity, and high FLAIR intensity. Higher B_{GM} values indicate stronger evidence for the presence of a WM lesion. Similarly, the B_{CSF} map generates voxel values that increase if there is a high probability of the voxel belonging to WM based on spatial location, low T1 intensity, and high FLAIR intensity. Higher B_{CSF} values support the assumption of the presence of "black hole" lesions. Finally, the B_{WM} map generates voxel values that increase if there is a high probability of the voxel belonging to WM based on spatial location, high T1 intensity, and high FLAIR intensity. Higher B_{WM} values suggest the presence of WM lesions [55].

The lesion belief maps are then summed together into the image B to input into the

lesion growth model later. The B_{GM} after application of a threshold κ (L_{init}) is chosen to initialize the lesion growth model as it requires a seed for the lesions to expand. This is done because the creators of the algorithm [55] found through extensive preliminary experiments that the PVE label would not yield lesions without any part assigned to GM.

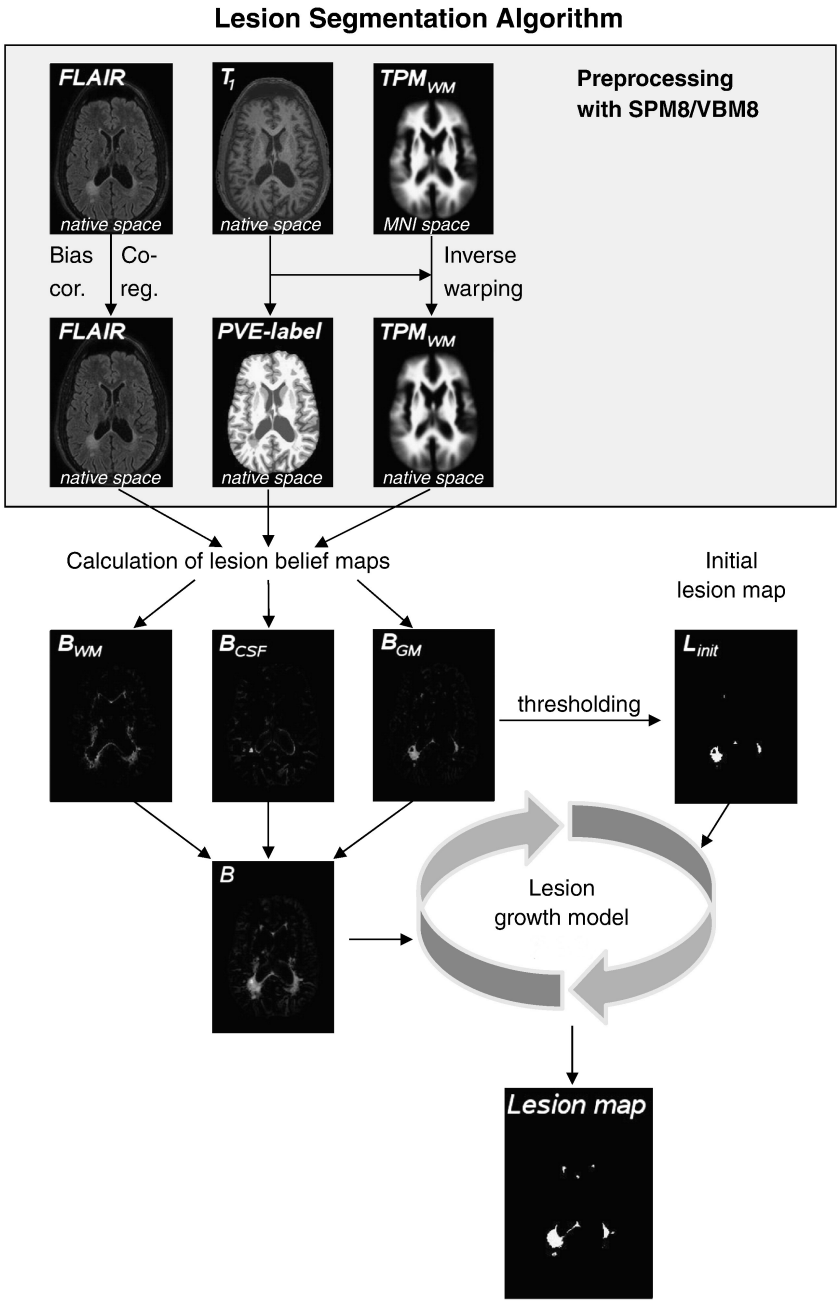


Figure 2.13: Image showing the lesions growth algorithms methodology to produce lesion segmentation maps. Figure reprinted from [55]

Lesion growth model

The growth of lesions is modeled by expanding L_{init} towards the lesion belief map (B). The model achieves this by iteratively analyzing neighboring voxels and assigning

them to lesions based on specific conditions, i.e., a voxel that shares a common border with a lesion voxel is considered to be a lesion, or that a lesion could only grow along those voxels which have a positive lesion belief value. The iteration continues until no further voxels are eligible for lesion assignment. During this process, the probability of belonging to WM or GM is compared to the probability of belonging to lesions [55].

2.5.3 Lesion prediction algorithm

Lesion segmentation by the lesion prediction algorithm (LPA) [56] is implemented in the LST toolbox version 3.0.0 for SPM. The algorithm is a supervised classification algorithm and consists of a binary classifier in the form of a logistic regression model. It is trained on the data of 53 MS patients with severe lesion patterns obtained at the Department of Neurology, Technische Universität München, Munich, Germany. As covariates for this model, a similar lesion belief map was used as for the lesion growth algorithm [55] and a spatial covariate considering voxel-specific changes in lesion probability. Parameters of this model are used to segment lesions in new images by providing an estimate of the lesion probability for each voxel.

Preprocessing and feature extraction

The algorithm extracts two features from the available MR images, beginning with the position of each brain voxel in standard space (MNI) [58]. Then FLAIR images that have been coregistered to a T1-weighted image are normalized to MNI space using the "Normalize" function implemented in SPM. This creates an inverse deformation field that can map MNI coordinates into the subject-specific native space [56].

The second feature is a lesion belief map. To produce this image, the algorithm roughly segments the FLAIR image into three main tissue classes GM, WM, and CSF. Subsequently, FLAIR intensities are standardized by dividing each voxel by the mean of the segmented GM. The mean of the standardized GM voxels is then subtracted from all FLAIR intensities, only keeping the positive differences by setting negative ones to zero. Furthermore, the remaining differences are multiplied by a tissue probability map for WM, which is obtained by applying the inverse deformation field from above to the tissue probability maps included in SPM. The resulting lesion belief map shows voxels that appear hyperintense in the FLAIR image and are likely part of WM in healthy subjects, making them possible lesion candidates [56].

Training

When training the algorithm, the features extracted during preprocessing are combined using a voxel-wise logistic regression model with a spatially varying intercept. The model equation (Equation 2.14) uses y_{ij} as the reference lesion map for the jth voxel and the ith subject. The linear predictor Equation 2.15 has β_0 as the overall intercept, x_{ij} as the value of the lesion belief map for the jth voxel and the ith subject and β_1 as the corresponding effect of the lesion belief map [56].

$$y_{ij} \sim B(\pi_{ij}) \quad \text{with} \quad \pi_{ij} = \frac{exp(\eta_{ij})}{1 + exp(\eta_{ij})} \tag{2.14}$$

$$\eta_{ij} = \beta_0 + \beta_1 x_{ij} + \gamma_j \tag{2.15}$$

Lesion segmentation

In contrast to training, the algorithm does not need a T1-weighted image for segmentation, as this was only necessary to obtain reference lesion maps using the LGA. To extract the relevant features for lesion segmentation, the algorithm first uses estimated inverse deformation fields to map the mean image of the spatially varying intercept from MNI space into the subject-specific native space ($\hat{y}$). Intensities of the FLAIR image are then corrected for bias field inhomogeneity with the "Segment" function in SPM. The lesion segmentation is performed by computing the lesion probability for each voxel using Equation 2.17 with the linear predictor seen in Equation 2.16 [56].

$$\hat{\eta}_j = \hat{\beta}_0 + \hat{\beta}_1 x_j + \hat{\gamma}_j \tag{2.16}$$

$$\hat{\pi}_j = \frac{exp(\hat{\eta}_j)}{1 + exp(\hat{\eta}_j)} \tag{2.17}$$

2.5.4 NicMSlesions

NicMSlesions is an MS lesion segmentation algorithm published in 2018 and was one of the top algorithms to complete the Medical Image Computing and Computer Assisted Intervention (MICCAI) challenge on MS lesion segmentation in 2016. The algorithm utilizes a CNN architecture and MS databases from MICCAI to accurately segment white matter (WM) lesions on T1 and FLAIR images [59]. As MS lesions only make up about $\sim$ 1.7% of the total brain volume of a patient, this reduces the number of positive training samples from the available voxels in the image [60]. This negatively affects CNNs as they often suffer from overfitting when insufficient data is

available for training [61]. NicMSlesions tries to solve this issue by optimizing the cascade of two identical CNNs, where the first network (CNN1) is trained to be more sensitive to revealing possible candidate lesion voxels. In contrast, the second network (CNN2) is trained to reduce the number of false positive outcomes [59][62].

Architecture

The nicMSlesions algorithm uses a 7-layer architecture for both CNN1 and CNN2. Each network comprises two stacks of convolution and max-pooling layers with 32 and 64 filters, respectively. Convolutional layers are followed by a fully-connected layer of size 256 and a soft-max FC layer of size 2 that returns the probability of each voxel belonging to the positive and negative class [62].

Training

The data used for training is acquired by extracting a balanced set of 11×11×11 patches from the FLAIR and T1-weighted input images. This set also includes a random selection of normal-appearing tissue voxels. The training data is then split into training and validation sets to optimize the neural network weights. The training set adjusts the weights, while the validation set measures how well the trained CNN performs after an epoch. Both CNNs are trained individually without parameter sharing, with ReLU [63] applied to all layers. The adaptive learning rate method (ADADELTA) proposed by Zeiler [64] is used to learn the network parameters, with a batch size of 128 and categorical cross-entropy as loss function. To reduce over-fitting, batch-normalization regularization [65] and Dropout with p=0.5 is used after both convolutional layers and before the first fully-connected layer, respectively. Additionally, the CNN model implements early stopping to prevent over-fitting by stopping training after several epochs without a decrease in the validation error. The final network parameters are taken from the epoch with the lowest error before stopping [62].

Data augmentation

The nicMSlesions algorithm performs data augmentation at batch time by multiplying the number of training samples by four. For each mini-batch, all patches are first rotated at 180 degrees in the axial plane. New versions of the patches are generated by horizontally flipping both the original and rotated versions. Other rotations than 180 degrees are avoided to roughly maintain the symmetry of the brain and avoid artificial rotations of brain structures. Valverde et al. found through empirical evaluations that rotated patches increased the segmentation accuracy of the proposed method in $\sim 1.5\%$ when compared to non-rotated patches [62].

Testing

To test the trained model, the algorithm makes the new unseen images undergo the same CNN cascade architecture. Feature patches are obtained for all brain voxels in the input of each new subject and evaluated using the first trained CNN to discard voxels with a low probability of being a lesion. The algorithm then uses the remaining voxels to evaluate using the second CNN to acquire the final probabilistic lesion mask, which is then linearly thresholded to obtain binary output masks. The resulting binary masks are further refined by discarding regions with lesion size below the automatically optimized parameter l_{min}, computed once after training and applied to the unseen testing images. This is done to reduce the number of false positives [62].

2.5.5 U-Net

The 2D U-Net architecture (Figure 2.14) was proposed for biomedical image segmentation at the 2015 MICCAI conference [8]. It is based on a modified CNN architecture to be able to produce more precise segmentations with fewer training images. The name of the network comes from its symmetrical shape that forms a "U" with its downward contracting path and upward expanding path. These paths are also known as the encoder and decoder. The encoder works similarly to CNN in that it aggregates the semantic information at the expense of spatial information. To recover this missing spatial information, the decoder receives information from the deep layers of the U-Net and recombines it with the information saved in previous layers.

The architecture of the network includes operations such as up-convolutions and skip connections [8]. Up-convolution is a trainable technique used to upsample or expand an image, unlike the downsampling max-pooling operation. The upsampled feature mapping is concatenated with the previous feature mapping from the corresponding layer in the contracting path, facilitated through a skip-connection. This allows the concatenated feature maps from the opposite layer to be reused, recovering information from previous layers before downsampling, and preserving feature maps even in a deep network. This approach mitigates the issue of vanishing gradients caused by backpropagation in deeper networks [44]. Many newer networks designed for segmentation tasks have utilized the U-Net architecture as a base, and a 3D U-Net architecture was introduced in [66] by replacing all 2D operations with their 3D counterparts, such as 3D convolutions, 3D max pooling, and 3D up-convolutions.

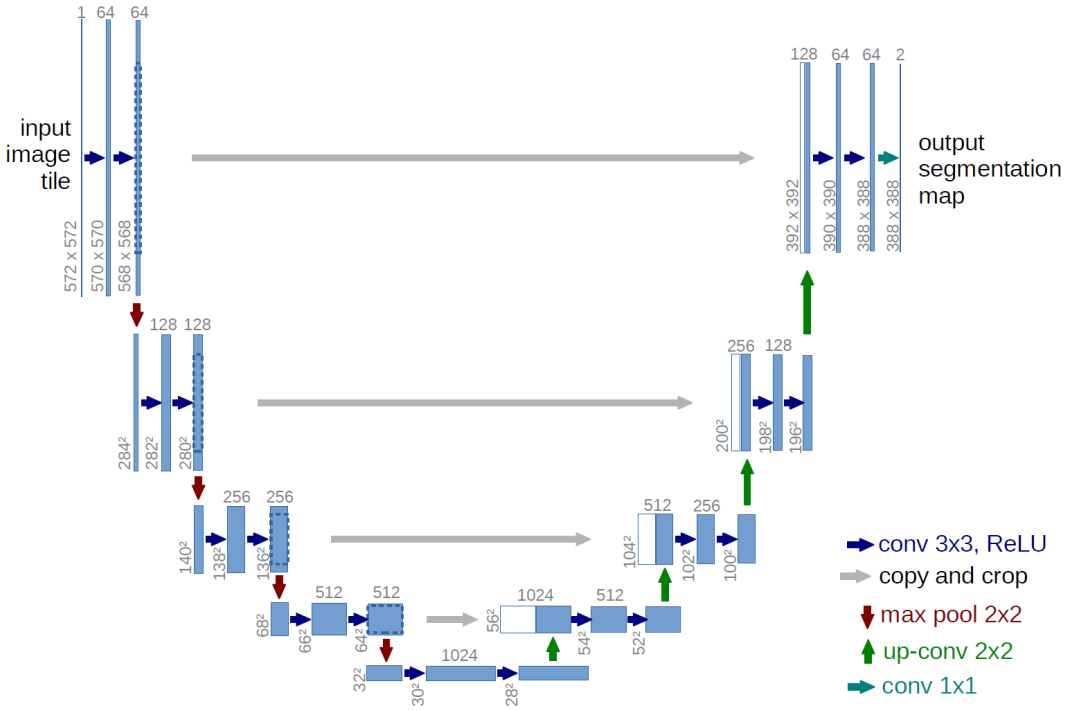


Figure 2.14: Image of the 2D U-Net architecture. It includes blue boxes that represent multi-channel feature maps, with the number of channels indicated at the top of each box. The lower left corner of each box displays the x-y-size. Additionally, white boxes are utilized to depict copied feature maps. The arrows indicate the various operations taking place. Figure is reprinted from Ronneberger et al. [8]

2.5.6 fastMONAI

fastMONAI is a low-code Python-based open-source deep learning library. It is built upon fastAI, MONAI, and TorchIO [10]. The library is created for the use of advanced deep learning techniques in 3D medical image analysis for solving regression, segmentation, and classification tasks. fastMONAI provides functionalities to step through data loading, preprocessing, training, and result interpretations [10].

2.5.7 nnU-Net

The nnU-Net, which is an abbreviation of no-new-UNet, is a deep learning-based segmentation method that automatically configures itself, including preprocessing, network architecture, training, and post-processing for any new task. It was published in 2020 by Isensee et al [9]. Their implementation was developed and validated on the datasets provided by the Medical Segmentation Decathlon and yields a segmentation method that performs the automated configuration for arbitrary new datasets. The configuration is made to be holistic, meaning it covers the entire segmentation pipeline without the need for manual intervention [9].

Architecture

The 3D nnU-Net has an architecture that is similar to that of the U-Net. Two plain convolutional layers are used between pooling layers in the encoder and transposed convolution operations in the decoder. Isensee et al. deviate from the original U-Net architecture in that they have replaced the ReLU activation function with leaky ReLU and use instance normalization instead of batch normalization [67].

Training

The nnU-Net is set to train for 1000 epochs using stochastic gradient descent (SGD) as an optimizer, a momentum of 0.99, and a learning rate of 0.01. The learning rate has been set to decay using $(1 - \frac{epoch}{epoch_{max}})^{0.9}$, and the loss function used is a combination of cross-entropy and dice loss. Isensee et al. use downsampled ground truth segmentation masks to compute the loss for each output of the model. The training objective is to minimize the sum of the losses at all resolutions, so the weights for each resolution are halved from the previous one and are normalized to sum to 1. To prevent class imbalances from affecting the training, the algorithm uses oversampling by randomly selecting 66.7% of the samples from random locations within the training case and guaranteeing that 33.3% of patches contain foreground classes present in the training sample. Data augmentation techniques are also used during training, such as rotations, scaling, noise and brightness simulations, and mirroring [9].

Inference

To predict images, nnU-Net uses a sliding window approach where the window size equals the patch size used during training. The adjacent predictions overlap by half of the size of the window as the segmentation accuracy decreases toward the borders of the window. To suppress stitching artifacts and reduce the influence of positions close to borders, nnU-Net applies a Gaussian importance weighting, increasing the weight of the center voxels in the softmax aggregation [9].

2.6 Performance evaluation

Being able to compare images to evaluate the quality of image segmentation is an important part of measuring processes in this field of research [68]. Evaluation of semantic segmentation can be a complex process because it is required to measure classification accuracy as well as localization correctness. The aim is to quantify the similarity between the predicted segmentation and the annotated ground truth [69].

2.6.1 Dice score

A simple spatial overlap index is the Dice similarity coefficient (DSC) (Figure 2.15), which was first proposed by Lee R. Dice in 1945 [70] and has been adopted to validate the segmentation of white matter lesions in MRIs [71]. DSC is a spatial overlap index between two segmentations, A and B (Equation 2.18) The value of the DSC can have any value between 0 and 1. A value of 1 indicates complete spatial overlap between two sets of binary segmentation results, and 0 indicates no overlap.

$$DSC(A,B) = \frac{2(A \cap B)}{(A+B)} \tag{2.18}$$

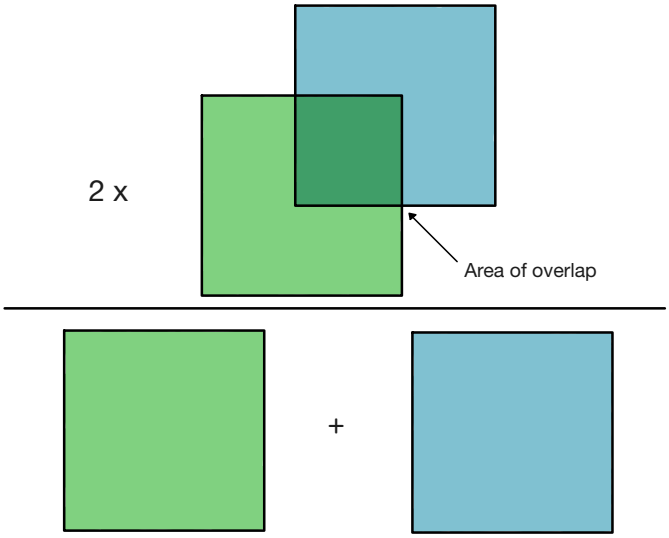


Figure 2.15: Image of how the dice score coefficient (DSC) between two images is calculated. The two images are represented with the green and blue squares.

2.6.2 Sensitivity and specificity

In machine learning, sensitivity, and specificity are commonly used metrics for evaluating the performance of a classification or segmentation model [72]. Sensitivity (Equation 2.19), also known as recall or true positive rate, measures the proportion of actual positive cases that the model correctly identifies as positive. It is calculated as the number of true positives (TP) divided by the sum of true positives and false negatives (FN). Specificity (Equation 2.20) measures the proportion of actual negative cases that the model correctly identifies as negative. It is calculated as the number of true (TN) negatives divided by the sum of true negatives and false positives (FP) [73][72].

$$Sensitivity = \frac{TP}{TP+FN} \tag{2.19}$$

$$Specificity = \frac{TN}{TN+FP} \tag{2.20}$$

Both sensitivity and specificity can be important tools for evaluating a model's performance. A high sensitivity indicates that the model is good at identifying positive cases, while a high specificity indicates that the model is good at identifying negative cases and has few false positives.

2.7 Recent advances

The early stages of multiple sclerosis (MS) can be challenging to diagnose due to its diverse clinical presentation. However, MR images are able to show brain lesions which are effective imaging biomarkers for the disease. This makes MRI crucial to diagnose, estimate disease stage and predict the future outcomes for the patient [74]. Since the detection of these lesions is time-consuming and susceptible to manual errors, image processing methods grounded in machine learning techniques are used to apply automatic segmentation to MR images. Within this field, the computer-aided diagnosis systems (CADS) used can be split into two categories: conventional machine learning methods and techniques that focus on deep learning [6].

Conventional machine learning methods have predominantly been used for MS diagnosis using MRI or clinical data. However, this approach is complex, requiring expertise in multiple fields of AI. Conventional machine learning CADS suffers from several limitations, including high computational costs associated with multiple algorithms and inefficient performance when dealing with large amounts of MRI input data [6].

Deep learning methods have gained popularity in diagnosing diseases using medical data, particularly in the field of MRI data analysis. One of the strengths is their ability to automatically extract latent feature representations, eliminating the need for manual feature extraction. DL also allows for consistent integration of feature extraction and classification [6]. A common problem of some neural networks is that tuning a large number of parameters and initializing the weights are complex processes. Additionally, these networks also often require large amounts of data to perform well [75].

Since 2016, research has focused on the application of DL models using MRI modalities to aid in different cases regarding MS. A large number of studies have been performed, and between the years 2016–2021, 92 articles were published in various journals on the topic of deep learning and MS [6] [75]. These studies are not restricted to

lesion segmentation but also include lesion identification, disease progression prediction, disease classification, and segmentation of other regions of interest (ROIs), such as the spinal cord. Most of this research has utilized convolutional neural networks or other types of neural networks. The performance of models primarily focused on MS lesion segmentation typically ranges between dice scores of 0.50-0.80, sensitivity of 0.50-0.90, and specificity of 0.60-0.90 [6].

When looking specifically into small lesion segmentation, a similar study to this project uses a U-Net architecture that focuses on the segmentation of new MS lesions. The network achieved a DSC of 40.3% [76]. Another similar study has been carried out by Battaglini et al. [77], whose work includes the automatic identification of new lesions in MS. There is also the work of Basaran et al. [5], who proposed a pipeline built upon the nnU-Net framework to segment MRI images containing both new and non-new lesion cases. Because new lesions often appear small, these topics resemble that of this thesis.

A limitation of these studies is that they focus on lesion segmentation in general and not small lesion specifically. It is already known that small lesions are particularly important predictors of disease progression and may provide information on effect of treatment. Furthermore, the studies do not compare different machine learning approaches and how these perform on small lesion segmentation. In the current work, small lesions are a particular target of investigation and several machine learning approaches, both classical approaches and deep learning approaches, are evaluated and compared. As such, the results from the current thesis can provide important novel information in the field of lesion segmentation in MS.

Chapter 3

Methods

This chapter provides an overview of the patient data utilized in this study and discusses the pre-processing steps taken. Additionally, the process of acquiring segmentations from all models is also described, as well as the methodology employed to evaluate the different models on the project data.

3.1 Study population

The longitudinal data used in the thesis include de-identified image data from 54 participants from a completed clinical study in MS at Haukeland University Hospital (HUH). The group of participants included 19 males and 35 females who were diagnosed with RRMS per the McDonald criteria guidelines from 2017 [24]. All participants were between the ages of 23-64 years, with the onset of MS symptoms no more than three years prior to diagnosis. The patients underwent clinical, neuropsychological, and MRI assessments for two years, but only image data were included in the current evaluations. The dataset contains images collected from MRI scans at baseline in the years between 2014-2016 and images acquired after 12 months. Each participant dataset contains the T2-FLAIR and T1-weighted images and masks for the brain and the present lesions. The overall clinical study was approved by the Regional Ethics Committee of Western Norway (registration number 2016/31/REK vest), and inclusion was based on written informed consent [78].

Table 3.1 shows how the data is divided into three sets. For each of these three sets, the mean age, number of participants of both genders, total lesions volume, and total number of lesions are shown. The total lesion volume and the number of lesions are both presented as a mean value calculated from the images in the three sets.

Table 3.1: Table showing the mean age, with standard deviation (SD) of the participants within each dataset. It also shows the number of males (m) and females (f) as well as the mean total lesion volume in mm^3 extracted from the images. The table also includes the mean number of lesions in each set.

	Age, mean ($\pm$ SD)	Gender m/f	Total lesion volume, mean (mm^3)	Number of lesions, mean
Training set	36 ± 9	15/23	2412	17
Validation set	39 ± 14	3/5	2873	20
Test set	49 ± 14	1/7	2632	25

3.2 MRI acquisition

MRI was conducted on a 3T Magnetom Prisma MR scanner (Siemens Healthineers, Erlangen, Germany). The detailed MRI acquisition protocol, which shows the image sequence parameters used for acquiring the images, can be seen in Table 3.2.

Table 3.2: Information about the detailed magnetic resonance imaging acquisition protocol. The sequence parameters are shown for the acquisition of both anatomical T1-weighted volumes and T2-FLAIR volumes.

Sequence parameters	Anatomical T1-weighted volumes	T2-FLAIR volumes
Sequence name	MPRAGE 3D T1-weighted sagittal volume	SPACE 3D T2-weighted sagittal volume
TE/TR/TI	2.28 ms/1.8 s/900 ms	386 ms/5 s/1600 ms
Acquisition matrix	256 x 256 x 192	256 x 256 x 192
Field of view (FOV)	256 x 256 mm^2	256 x 256 mm^2
Slice thickness	1 mm	1 mm
Readout bandwidth	200 Hz/px	751 Hz/px
Total acquisition duration	7.40 min	6.17 min

The images were also processed by Icometrix (Leuven, Belgium) for supervised digital image analyses yielding cross-sectional data on global and regional brain volumes and lesion assessment. Icometrix also made the gold standard reference lesion masks that are used as the ground truth when looking into performance evaluation. The T1 and FLAIR volumes, as well as the reference lesion mask from each participant scan, were saved in folders only recognizable by the number assigned to them by Icometrix.

3.3 Image processing pipelines

3.3.1 Data splitting

The whole dataset consisted of 102 volumes from 54 participants. There are more volumes than participants as most participants were scanned two times with 12 months in between. For the purpose of training and testing, the dataset has been split into training, validation, and testing sets. Instead of splitting randomly, the volumes chosen

for the validation and testing sets were chosen based on if they contained smaller lesions or not. To be able to present the results from the models more accurately, all volumes were checked in order to see which of them contained smaller lesions. A script written in MATLAB (MATLAB V9.12 (R2022a), The Mathworks Inc., Massachusetts, United States) was used on the binary lesion masks in the dataset to ascertain the number of voxels in each lesion. The result from this corresponded well with what was shown when viewing the lesion masks with FSLeyes (FMRIB v6.0, Analysis Group, Oxford, UK). Sixteen volumes containing small lesions were then put in the validation and testing sets, eight in each. The corresponding sixteen volumes coming from the same participants were not used in the validation or test set as they had no small lesions. To avoid the models getting any biases they were also excluded from being put in the training set and were therefore not used for the project. The remaining 70 volumes were then used for training, maximizing the amount of data used for the project without using the same participants in different datasets. The training set was made sure to also contain small lesions, but not exclusively.

3.3.2 Localization of small lesions

The function *bwconncomp()* in MATLAB was used for locating the volumes that contain the small lesions. The function used a binary image to find and count the number of connected components. As the lesion masks are binary, all the lesions in the image are numbered as 1 and will therefore become a connected component when using this method. Since the images used as input have three dimensions, 26-connectivity is used in order to locate all the lesions in the mask as well as see how many voxels are involved in each of the connected components (Figure 3.1). For this project, the small lesions were considered to consist of ten voxels or fewer. The *bwconncomp()* function outputs the number of objects in the image as well as a list made up of all the voxels for each component. The length of this list is used to check the number of voxels in each lesion and thereby ascertain if they were small or not.

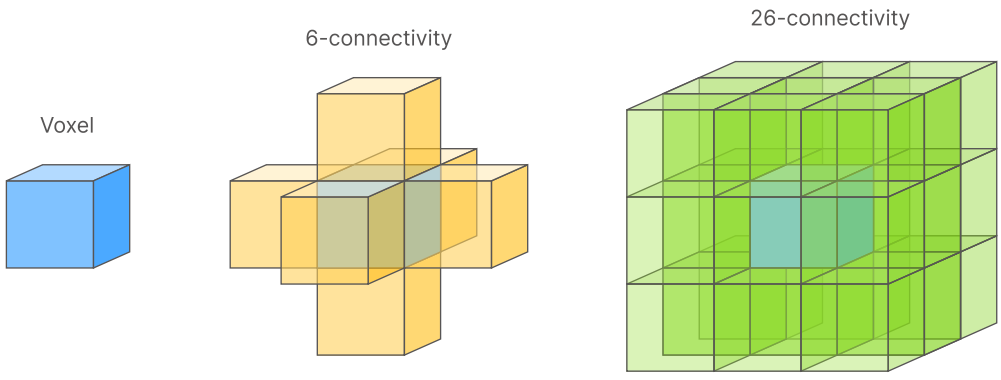


Figure 3.1: Image showing how connectivity looks in three dimensions. Figure inspired by [79].

3.4 Obtaining lesion segmentations

This section explains how the different algorithms and neural networks were utilized in order to output lesion segmentations to use for evaluation. In total, five different models were used to produce comparable segmentations. Two computers were used during this part. A DELL Precision 7250 with Ubuntu 20.04.1 LTS and Intel Core i7-6820HQ CPU @ 2.70Ghz x 8 processor was used to train and test the nicMSlesions algorithm. The rest of the training and testing was performed on an Alienware Area-51 R4 computer with Ubuntu 20.04.5 LST and Ryzen threadripper 1050x 16-core processor.

3.4.1 Lesion segmentation tool

LST is accessed through MATLAB and the SPM12 academic software toolkit (SPM12 revision 7771, Functional Imaging Laboratory, London, UK).

Lesion growth algorithm

The LGA requires a T1 and a FLAIR image in order to produce lesion segmentations. For both the validation and test datasets, the corresponding T1 and FLAIR NIfTI images were chosen as input. As stated in the documentation, different values for the initial threshold κ should be tested before requiring the final segmentation. To do this, the built-in tool Determination of optimal initial threshold (DOIT) was used following an original run-through of the algorithm. After obtaining the optimal initial threshold, the LGA was run on the T1 and FLAIR images from the test set. κ was set to the value acquired by DOIT. The rest of the values, such as the Markov random field parameter and the maximum number of iterations, was chosen to stay at their standard values, 1 and 50, respectively. All the resulting segmentations were saved in their corresponding folders and were summarised in an HTML report for viewing of the results.

Determination of optimal initial threshold

In order to use DOIT, the first run-through of the LGA was done on the images from the validation test set with κ values between 0.05 and 1 and an increment of 0.05, as suggested by the LST documentation. Each probability map is saved within each patient folder in the validation set as ples_lga_[κ]_rm[FLAIR].nii where κ is the number chosen between 0.05 and 1.

The determination of the optimal initial threshold tool requires real binary segmentations to use as a reference image as well as probability maps obtained by LGA. The binarised ground true lesion masks from the validation set were chosen as input, while

the LGA probability maps inside each patient folder were automatically searched for when running the algorithm. Furthermore, a threshold value for the LGA probability maps also needs to be specified. Here the standard value of 0.5 was chosen as it was suggested by the toolbox. After running DOIT, a CSV file is returned. The CSV file contains columns for the folder of the reference images, the name of the FLAIR images, the value of κ, the corresponding DSC, as well as values for sensitivity and specificity. The optimal κ was chosen based on which κ-value yielded the best dice score for all the images inputted.

Lesion prediction algorithm

The LPA only needs FLAIR images to produce lesion segmentation probability maps. However, it is possible to choose an additional image that is used during co-registration before the main lesion segmentation. For each participant in the test set the FLAIR and corresponding T1 NIfTI images were chosen. The algorithm was run two times, one for the validation set and one for the test set. The output from LPA is a probability map saved in each of the folders as ples_lpa_m[r][FLAIR].nii, and an HTML report to review the results.

3.4.2 NicMSlesions

The program was accessed from a public repository made by the authors of the algorithm. The functionalities of the program were enabled through its GUI, this is where all the images and options were inputted.

Training

For training, the method requires FLAIR and T1-weighted NIfTI images as inputs, as well as lesions as binary masks. The folder containing the training set is chosen and the identification tags for the two images were inputted, "FLAIR" and "T1", respectively. For the MASK tag, binarised lesion masks obtained from Icometrix were chosen since they were used as the gold standard for segmentation during this project. The "use pre-trained" button is also toggled, as this could help to improve the performance of the model after training. For our study, the pre-trained model "baseline_2ch" from the publication was used. Because changes to the other settings led to the nicMSlesions algorithm not producing any results, all parameters remained as suggested by the publication.

Inference

The beginning steps of the inference task were similar to when training. It was run two times in total, one for the validation set and one time for the test set. The same identification tags were used for the FLAIR and T1 images when training, as well as the same parameter options. The model outputted from the training algorithm was chosen before running the inference. Once completed, a new folder named after the trained model is created within each participant's directory, containing both the probabilistic and final hard segmentations.

3.4.3 U-Net employed trough fastMONAI

The implementation of U-Net through fastMONAI (fastMONAI v0.3.0) was done using the "Binary semantic segmentation" tutorial from the fastMONAI website, as baseline [80]. The data was read into a data frame using a CSV file containing the paths to the FLAIR image and lesion mask. As the training and testing set was already decided, the *train_test_split()* function in the tutorial is not used. The data is passed to the *MedDataset()* function as a way to extract and present information about the data. Multiple data augmentations were added such as normalization, random flipping, and affine rotations set to five degrees. To get the training data in the correct format to use in the learner function, the training data frame was inputted into the fastMONAI function *MedDataBlock.dataloaders()*.

Training

The network used for training was 3D U-Net supplied by MONAI (MONAI v1.1.0). The number of in_channels is set to one, as only the FLAIR NIfTI images were used for training. The learning rate was decided based on the plot resulting from the function *lr_find()*. The learning rate was thereafter set to a value that corresponded to one of the areas on the plot close to a steep decline. To train the model the fastai (fastai v2.2.0) function *fit_flat_cos()* is utilized, and it was set to run for 400 epochs with a batch size of four. The loss function and optimizer used were dice loss and Ranger21, respectively. Both were accessed through fastMONAI.

Inference

Inference was performed using the fastMONAI function *inference()* with the model trained in the previous step as input. The path to where the test images were located was also inputted, as well as the path to where the predictions were going to be saved.

3.4.4 nnU-Net

nnU-Net (nnU-Net V1) and its functionalities were downloaded from the public repository of Isensee et al [9]. This algorithm requires the data to be presented in another format than used for the other algorithms. To convert the data, Python code (Python 3.8.10, Microsoft (VSCode), Washington, USA) was written to copy images to the correct nnU-Net folder as well as rename them to fit the format. Once this was done, all the files were compressed using the *gzip* terminal command on Linux, as this is required for running the algorithm.

Training

The nnU-Net performs training using 5-fold cross-validation. The training command, therefore, had to be run five times in order to acquire all the folds needed before inference. The configuration chosen for training was the 3D full resolution U-Net, including the *–npz* command to save the lesion probabilities for use during inference. Because the other trained models were set to train for a maximum number of 400 epochs, the training class nnUnetTrainerV2 that was included in the download was changed to also fit this requirement.

Inference

Inference was performed using the nnUNet_predict command on the trained 3D full-resolution U-Net using all 5 folds. The *–save_npz* was specified as well to save the softmax probabilities alongside the predicted segmentation masks. The command *nnUNet_find_best_configuration* was omitted ahead of inference as only one configuration was used for training.

Conversion

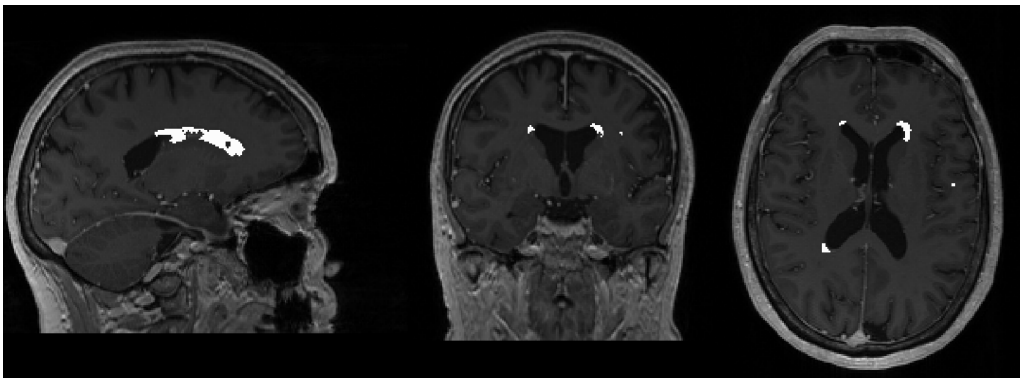
Because the lesion probabilites from the nnU-Net were outputted as numpy arrays as well as a pickle file containing the information to reconstruct the image, a function was made to convert the data from these files into images. The SimpleITK (SimpleITK v2.2.1, Apache License) interface was imported together with the pickle module in Python. Applying the information stored in the pickle file together with the numpy arrays stored in the *.npz* file the segmentations containing the lesion probabilities were reconstructed and saved as NIfTI images.

3.5 Postprocessing

3.5.1 Validation pipeline

To calculate the performance metrics for each model, the probability maps first had to be binarised so as to compare them to the reference lesion masks (Figure 3.1). A validation pipeline was therefore made to ascertain what value the probability maps should be thresholded at to make the comparable segmentations. To do this, the FSL terminal command *fslmaths -thr* was used. The command uses the inputted *thr* value to turn all the values alike or above the threshold input into ones and the values below to zero. The data from the validation dataset was used during this process. This is to make sure that the data that is to be used for testing remained untouched until it was time for performance evaluation. This allows for the evaluation to be unbiased and therefore provides a result that is representative of how well the models perform on unseen data.

To ensure a fair comparison between the segmentations produced by different models, the probability maps were initially thresholded at 0.5. The resulting binary lesion masks were then evaluated using the dice score metric to identify the model with the highest score. Once the model with the highest dice score was determined, the other models were thresholded again at various values until their dice score was as close as viable to the highest score. This was done to ensure that the resulting binary lesion masks were as similar as possible before proceeding to evaluate the models. Ultimately, the binary lesion masks generated by this procedure were used to evaluate the performance of the different models. This approach allowed for a standardized and objective comparison of the models based on their ability to accurately segment the lesions of interest.



(a) T1-weighted image with binary lesion mask

(b) Binary lesion mask

Figure 3.1: Image (a) shows how a binary lesion mask looks on top of its corresponding T1-weighted image, it shows the same image from three different perspectives. Image (b) shows the same binary lesion mask, without the T1 image. These images were used as the ground truth lesion masks. Here all the black areas are annotated with 0, while the lesion masks are annotated with 1. This image is also shown from three different perspectives, corresponding to the ones in (a). The images were generated through FSLeyes.

3.5.2 Extraction of small lesion predictions

All the models were trained to output a probability map for lesions of all sizes. However, to focus the evaluation specifically on small lesions, additional steps were incorporated into the process. This involved utilizing a MATLAB script that made use of the function *bwconncomp()* to identify and remove larger lesions from the final binary segmentations. A for-loop was implemented to iterate through each of the binary segmentations generated from the test set of each model. The length of the *pixelIdList*, obtained from the *bwconncomp()* function, was used to determine the size of each lesion in the segmentation. Any lesions larger than the specified size were removed by setting the corresponding list values to 0, effectively erasing them from the binary image. Once all the lesions in the image were checked, a binary image containing only small lesions remained. This image was saved using the *niftiwrite()* function and added to a new folder for later evaluation.

During the dice score calculation, the segmentations of each model were compared to the reference lesion masks from the test set that had undergone the same procedure to remove larger lesions. This allowed for a direct comparison of the model's ability to accurately segment small lesions.

3.6 Evaluation

3.6.1 Metrics

Three different performance metrics were used to evaluate the segmentations of the different models; DSC, sensitivity, and specificity. All the metrics were implemented in Python. As the segmentations are NIfTI images, NiBabel (Nibabel V5.0.0, Zenodo, Switzerland) was downloaded and imported into the code. Image loading was done using the function *nibabel.load(nifti_image).getfdata()*, which was used to turn the NIfTI image data into binary NumPy arrays. The function *calculate_dice_score(true_label, predicted_label)* was made to calculate the DSC between the two binary arrays representing the true and predicted lesion masks. The Numpy functions *NumPy.logical_and()* and *NumPy.logical_or()* imported from NumPy (Numpy v1.24, Numpy, Texas, United States) were used to calculate the arrays' intersection and union, respectively. The DSC was then found using Equation 2.18.

The same procedure of turning NIfTI image data into binary NumPy arrays was used before calculating the sensitivity and specificity of the predictions. Then the values TP, TN, FP, and FN from section 2.6.2 were computed using the function *NumPy.where()* to find where the values coincide and where they differ from each other. The metrics were then calculated using the Equations 2.19 and 2.20.

3.6.2 Visualizations

Three types of plots were made for easier comparison of the models and to better visualize the results of the different evaluation metrics.

Segmentations

The first type of plot made was a plot containing the T1 images for all eight patients in the test set, with the true lesion mask and predicted lesion mask on top. The two different masks are shown in different colors, and a third color is added to indicate the overlap between them. As the images are three-dimensional, in order to visualize the segmentations in a reasonable way, only two-dimensional slices from the z-direction were used in the plot. The slices chosen for each image are the slices that contain the highest number of true lesion masks. For each segmentation of the eight participants, the three metrics are shown above each image. The values shown are the results of the whole segmentation, not just the results from the specific slice shown. The matplotlib

(matplotlib V3.6.0, Michael Droettboom, et al.) functions *imshow()* and *show()* were utilized to produce the side-by-side segmentation plots.

Bar plots

For easier comparison of the different segmentations, bar plots were made for the three different metrics showing the result from each model side by side. Each bar represents one of the models, and the height of each bar is the mean value of the eight participants for the specified metric shown. Because it is a mean value that is shown, it also includes the standard deviation represented by an error bar. To produce the plots the matplotlib function *bar()* was used.

Line charts

Line charts were made to visualize how the scores of the segmentations vary with different lesion sizes. Three line charts were made, one for each of the performance metrics. Although it is the mean value that is shown, the standard deviations for each data point were left out to not obscure the different lines. The charts were made using the *plot()* function from matplotlib, with each of the models being represented with a specific marker and color.

Chapter 4

Results

This section will present the results of the automatically segmented multiple sclerosis lesions using the different approaches described in the previous chapters. The results will be presented with dice scores, sensitivity scores, and specificity scores for the distinct segmentations. The first two are presented with three decimals as the uncertainties occasionally get very small, and this will make way for a better comparison of the models. The specificity is shown with four decimals as this score is rather high for all models and to express the small differences between them. Both segmentations of the whole brain and segmentations which are only directed at the small lesions are included. Even though the thesis mainly focuses on lesions smaller than or equal to 10 voxels ($\leq$ 10 mm^3), lesion sizes between 100 mm^3 - 10 mm^3 have been included to see how the decreasing lesion sizes affect the performance of the models. When looking at the image segmentations, the figures will be shown using two-dimensional slices, despite the fact that the real segmentations are three-dimensional.

4.1 Lesion growth algorithm

The first results were acquired by running the T1 and FLAIR images from each subject in the test set through LGA. This yielded segmentations of lesions of all sizes. The κ value determined through the DOIT function was 0.2, which was used when acquiring the segmentations. Large lesions were removed when looking at the small lesion results.

4.1.1 Segmentation of all lesions

Table 4.1 shows the dice score, sensitivity, and specificity of all the lesions resulting from the LGA. Individually the automatic segmentation varies, having a dice score of anything between 0.048 and 0.781 giving the mean dice 0.479 $\pm$ 0.232. The dice score

and sensitivity seem to be positively correlated, while the specificity stays similar.

Table 4.1: Table showing the results from using the lesion growth algorithm (LGA) to predict segmentations of the eight subjects in the test set. The table shows individual results as well as the mean scores.

Individual scores for LGA								
	Subject 1	**Subject 2**	**Subject 3**	**Subject 4**	**Subject 5**	**Subject 6**	**Subject 7**	**Subject 8**
Dice Score	0.408	0.632	0.582	0.781	0.530	0.195	0.652	0.048
Sensitivity	0.275	0.704	0.516	0.823	0.581	0.148	0.709	0.030
Specificity	0.9999	0.9997	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
Overall scores								
Mean Dice Score	0.479 ± 0.232							
Mean Sensitivity	0.473 ± 0.271							
Mean Specificity	0.9999 ± 0.0001							

Figure 4.1 shows the segmentations of the eight subjects in the test set. The slice chosen for each subject is the slice containing the most ground true lesions. LGA performs the best on subjects S2, S4, and S7, where there are large areas of overlap, seen in purple. Subjects S6 and S8 show low dice and sensitivity scores corresponding with the little overlap seen in these images.

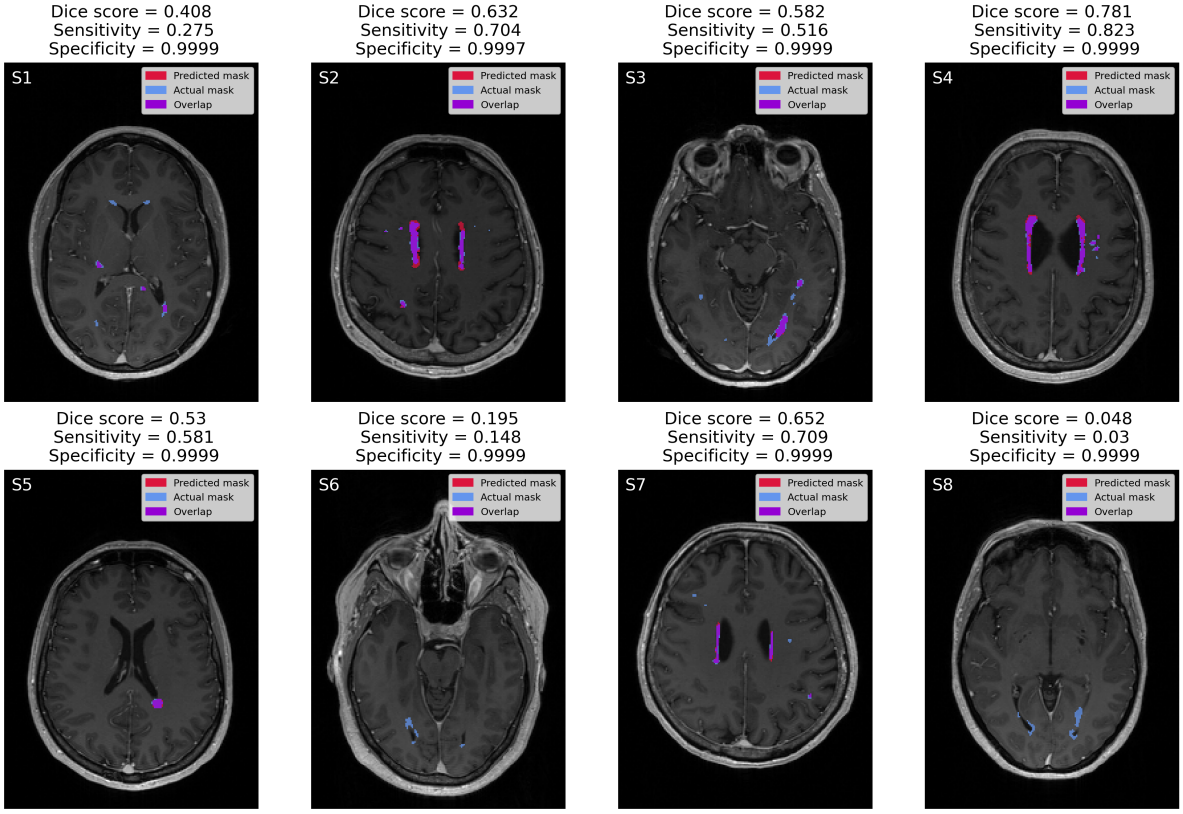


Figure 4.1: Image showing the lesion segmentations from the lesion growth algorithm (LGA) of the eight subjects in the test set. The figures are labeled between S1-S8, corresponding to the subjects 1-8. Above each subject are the associated dice score, sensitivity, and specificity. The ground true lesion masks are shown in blue, the predicted masks from LGA are shown in red, and their overlap is in purple.

4.1.2 Segmentation of small lesions

Table 4.2 shows the mean dice score, mean sensitivity and mean specificity of different lesion sizes. For each of the sizes, 100 mm^3 down to 10 mm^3, all lesions larger than the specified size were removed from both the predicted segmentation and the ground true one before calculating the results. Table 4.2 shows the dice score and sensitivity decreases with the smaller lesion sizes while the specificity stays the same up until lesions smaller than 10 mm^3, where there are no overlapping segmentations.

Table 4.2: Table showing the results from using the lesion growth algorithm (LGA) to predict segmentations of the eight subjects in the test set. The results shown are after larger lesions are removed from both the ground truth and predicted lesions. For each of the lesion sizes shown, all lesions above the mentioned volume are removed. The table shows the mean dice score, mean sensitivity, and mean specificity for lesion sizes smaller than 100mm^3 to 10mm^3 with 10 mm^3 increment.

Mean scores for LGA on decreasing lesion sizes

	Mean dice score	**Mean sensitivity**	**Mean specificity**
≤100 mm^3	0.233 ± 0.152	0.195 ± 0.131	0.9999 ± 0.0001
≤90 mm^3	0.228 ± 0.147	0.197 ± 0.125	0.9999 ± 0.0001
≤80 mm^3	0.180 ± 0.168	0.152 ± 0.142	0.9999 ± 0.0001
≤70 mm^3	0.182 ± 0.162	0.152 ± 0.137	0.9999 ± 0.0001
≤60 mm^3	0.161 ± 0.141	0.138 ± 0.120	0.9999 ± 0.0001
≤50 mm^3	0.155 ± 0.135	0.130 ± 0.110	0.9999 ± 0.0001
≤40 mm^3	0.188 ± 0.119	0.099 ± 0.090	0.9999 ± 0.0001
≤30 mm^3	0.089 ± 0.120	0.074 ± 0.093	0.9999 ± 0.0001
≤20 mm^3	0.089 ± 0.104	0.071 ± 0.078	0.9999 ± 0.0001
≤10mm^3	0.0 ± 0.0	0.0 ± 0.0	1.0 ± 0.0

4.2 Lesion prediction algorithm

The predicted segmentations from LPA were acquired by running the FLAIR and T1 NIfTI images from each subject in the test set through the LPA algorithm in MATLAB. The results on the smaller lesions were calculated after larger lesions were stripped from the segmentations.

4.2.1 Segmentation of all lesions

Table 4.3 shows the dice score, sensitivity, and specificity scores of all the lesions in the segmentations predicted by LPA. The dice scores of the eight subjects vary between 0.384 and 0.789 giving a mean dice score of 0.562 ± 0.117. The same trend can be seen for sensitivity that varies between 0.402 to 0.825 giving a mean sensitivity of 0.605 ± 0.123. The specificity stays a similar value lying between the numbers 0.9996 and 0.9999 giving a mean of 0.9998 ± 0.0001.

Table 4.3: Table showing the results from using the lesion prediction algorithm (LPA) to predict segmentations of the eight subjects in the test set. The table shows individual results as well as the mean scores.

Individual scores for LPA								
	Subject 1	**Subject 2**	**Subject 3**	**Subject 4**	**Subject 5**	**Subject 6**	**Subject 7**	**Subject 8**
Dice Score	0.458	0.607	0.578	0.789	0.580	0.384	0.630	0.470
Sensitivity	0.402	0.697	0.575	0.825	0.602	0.525	0.698	0.518
Specificity	0.9999	0.9996	0.9999	0.9999	0.9999	0.9999	0.9998	0.9999
Overall scores								
Mean Dice Score	0.562 ± 0.117							
Mean Sensitivity	0.605 ± 0.123							
Mean Specificity	0.9998 ± 0.0001							

The individual segmentations and their corresponding scores can be seen in Figure 4.2. LPA achieves a greater mean dice and sensitivity score than the LGA on the segmentations. This corresponds to the individual images S1-S8, which show fewer blue areas than the LGA for most subjects. It has a similar trend to LGA, with the greatest scores belonging to S2, S4, and S7, and the lowest scores belonging to S6 and S1.

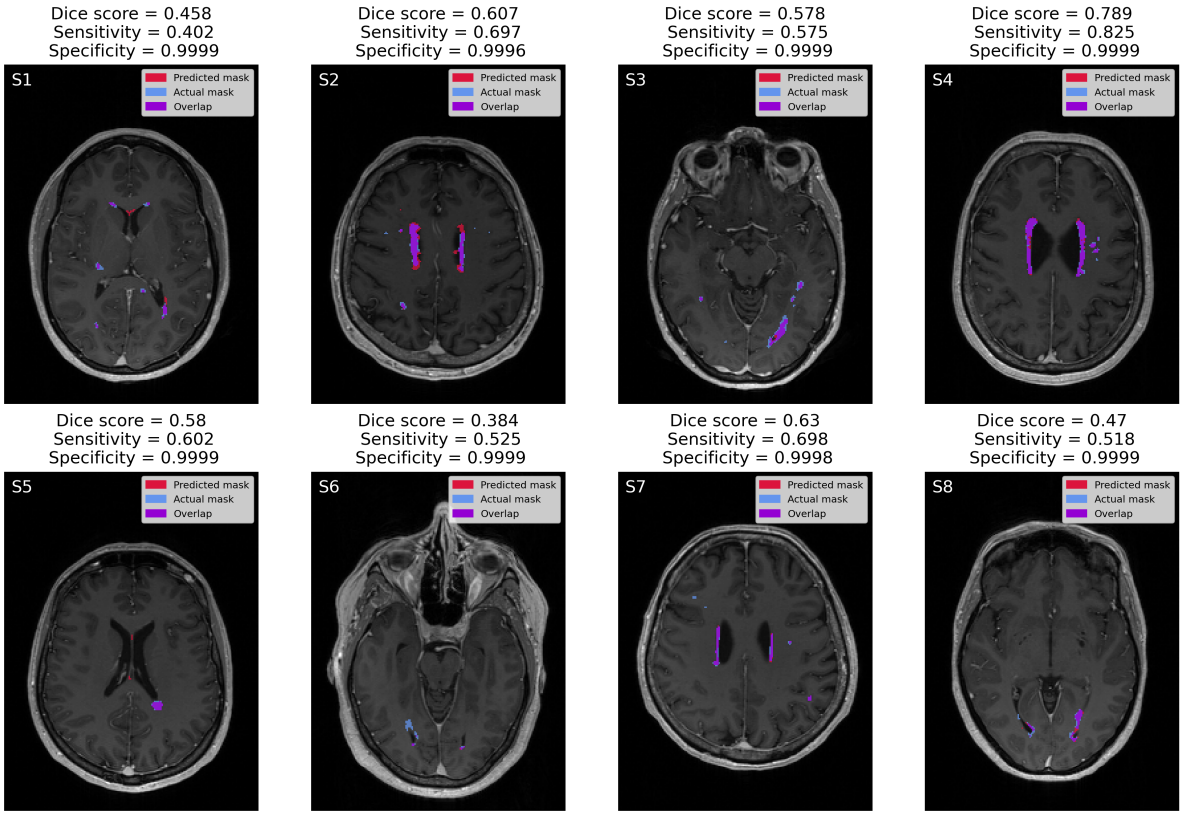


Figure 4.2: Image showing the lesion segmentations from the lesion prediction algorithm (LPA) of the eight subjects in the test set. The figures are labeled between S1-S8, corresponding to the subjects 1-8. Above each subject are the associated dice score, sensitivity, and specificity. The ground true lesion masks are shown in blue, the predicted masks from LPA are shown in red, and their overlap is in purple.

4.2.2 Segmentation of small lesions

Table 4.4 shows the mean dice score, mean sensitivity and mean specificity of segmentations with decreasing lesion sizes. Looking at lesions smaller than 100 mm^3, the mean dice score is 0.281 $\pm$ 0.144, and the mean sensitivity is 0.278 $\pm$ 0.143. At lesions $\leq$ 30 mm^3, the scores diminish below 0.200 and become 0.0 at lesions $\leq$ 10 mm^3. The specificity stays the same at 0.9999 $\pm$ 0.0001 up until lesions $\leq$ 10 mm^3 where it becomes 1.0 $\pm$ 0.0 as there are no overlapping predictions here.

Table 4.4: Table showing the results from using the lesion prediction algorithm (LPA) to predict segmentations of the eight subjects in the test set. The results shown are after larger lesions are removed from both the ground truth and predicted lesions. For each of the lesion sizes shown, all lesions above the mentioned volume are removed. The table shows the mean dice score, mean sensitivity, and mean specificity for lesion sizes smaller than 100mm^3 to 10mm^3 with 10 mm^3 increment.

Mean scores for LPA on decreasing lesion sizes

	Mean dice score	**Mean sensitivity**	**Mean specificity**
$\leq$100 mm^3	0.281 $\pm$ 0.144	0.278 $\pm$ 0.143	0.9999 $\pm$ 0.0001
$\leq$90 mm^3	0.289 $\pm$ 0.139	0.279 $\pm$ 0.134	0.9999 $\pm$ 0.0001
$\leq$80 mm^3	0.267 $\pm$ 0.148	0.254 $\pm$ 0.129	0.9999 $\pm$ 0.0001
$\leq$70 mm^3	0.280 $\pm$ 0.134	0.276 $\pm$ 0.115	0.9999 $\pm$ 0.0001
$\leq$60 mm^3	0.271 $\pm$ 0.145	0.246 $\pm$ 0.126	0.9999 $\pm$ 0.0001
$\leq$50 mm^3	0.247 $\pm$ 0.097	0.244 $\pm$ 0.083	0.9999 $\pm$ 0.0001
$\leq$40 mm^3	0.215 $\pm$ 0.075	0.215 $\pm$ 0.063	0.9999 $\pm$ 0.0001
$\leq$30 mm^3	0.158 $\pm$ 0.112	0.149 $\pm$ 0.104	0.9999 $\pm$ 0.0001
$\leq$20 mm^3	0.135 $\pm$ 0.126	0.104 $\pm$ 0.095	0.9999 $\pm$ 0.0001
$\leq$10mm^3	0.0 $\pm$ 0.0	0.0 $\pm$ 0.0	1.0 $\pm$ 0.0

4.3 NicMSlesions

The segmentations from the nicMSlesions algorithm were acquired by running the FLAIR and T1 NIfTI images from each subject in the test set through the downloaded nicMSlesions program. No options in the program were changed before acquiring the test results. The results on the smaller lesions were calculated after larger lesions were removed from the segmentations.

4.3.1 Segmentation of all lesions

Table 4.5 shows the dice sore, sensitivity, and specificity scores of the segmentations made by the nicMSlesions algorithm. The dice scores vary between 0.0 and 0.526, giving a mean dice score of 0.337 $\pm$ 0.146. The sensitivity varies between 0.0 and 0.439, leading to a mean sensitivity of 0.258 $\pm$ 0.123. The specificity lies between 0.9998 and 1.0, having a mean score of 0.9999 $\pm$ 0.0001.

Table 4.5: Table showing the results from using the nicMSlesions algorithm to predict segmentations of the eight subjects in the test set. The table shows individual results as well as the mean scores for each performance measure.

Individual scores for nicMSlesions								
	Subject 1	**Subject 2**	**Subject 3**	**Subject 4**	**Subject 5**	**Subject 6**	**Subject 7**	**Subject 8**
Dice Score	0.291	0.443	0.526	0.396	0.0	0.306	0.385	0.353
Sensitivity	0.185	0.379	0.439	0.277	0.0	0.227	0.263	0.293
Specificity	0.9999	0.9998	0.9999	0.9999	1.0	0.9999	0.9999	0.9999
Overall scores								
Mean Dice Score	0.337 ± 0.146							
Mean Sensitivity	0.258 ± 0.123							
Mean Specificity	0.9999 ± 0.0001							

Figure 4.3 shows the lesion segmentations made by the nicMSlesions algorithm on the eight subjects in the test set. Most notable is the segmentation for S5, where there is no overlap present throughout the brain. This can also be seen in Table 4.5, where the dice and sensitivity scores are 0 for subject 5. Additionally, nicMSlesions struggle when segmenting lesions across various brain regions, as evident from the presence of substantial blue patches observed in nearly all subjects S1-S8.

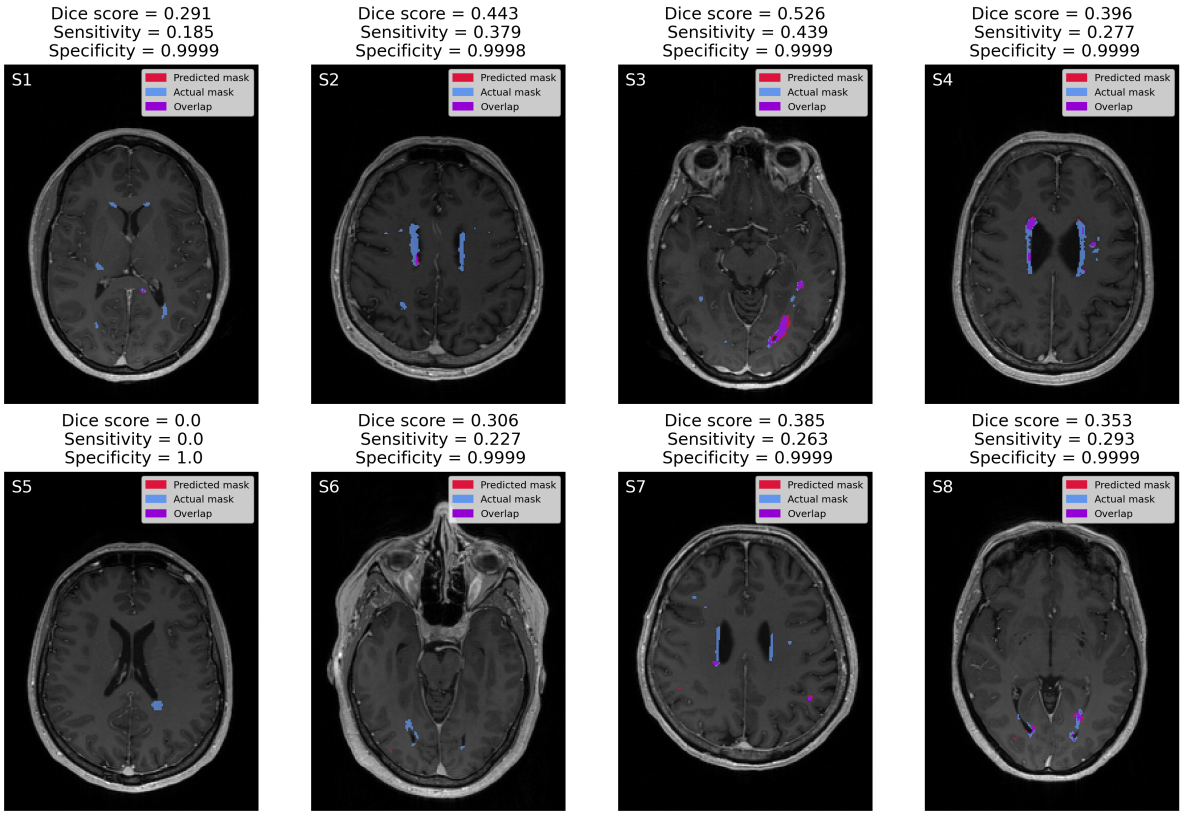


Figure 4.3: Image showing the lesion segmentations from the nicMSlesions algorithm of the eight subjects in the test set. The figures are labeled between S1-S8, corresponding to subjects 1-8. Above each subject are the associated dice score, sensitivity, and specificity. The ground true lesion masks are shown in blue, the predicted masks from the algorithm are shown in red, and their overlap is in purple.

4.3.2 Segmentation of small lesions

Table 4.6 presents the mean dice score, sensitivity, and specificity for segmentations of decreasing lesion sizes. The segmentations containing lesions $\leq$ 100 mm^3 got a mean dice score of 0.190 $\pm$ 0.136, mean sensitivity of 0.155 $\pm$ 0.107, and mean specificity of 0.9999 $\pm$ 0.0001. The mean dice score and sensitivity decrease with the decreasing lesion sizes. For segmentations with lesions $\leq$ 10 mm^3, the dice score is 0.015 $\pm$ 0.032, the sensitivity is 0.013 $\pm$ 0.028, and the specificity is 1.0 $\pm$ 0.0. An example of a nicMSlesions prediction on small lesions can be seen in supplementary Figure B.1.

Table 4.6: Table showing the results from using the nicMSlesions algorithm to predict segmentations of the eight subjects in the test set. The results shown are after larger lesions are removed from both the ground truth and predicted lesions. For each of the lesion sizes shown, all lesions above the mentioned volume are removed. The table shows the mean dice score, mean sensitivity, and mean specificity for lesion sizes smaller than 100mm^3 to 10mm^3 with 10 mm^3 increment.

Mean scores for nicMSlesions on decreasing lesion sizes

	Mean dice score	**Mean sensitivity**	**Mean specificity**
$\leq$100 mm^3	0.190 $\pm$ 0.136	0.155 $\pm$ 0.107	0.9999 $\pm$ 0.0001
$\leq$90 mm^3	0.182 $\pm$ 0.126	0.149 $\pm$ 0.100	0.9999 $\pm$ 0.0001
$\leq$80 mm^3	0.131 $\pm$ 0.114	0.104 $\pm$ 0.085	0.9999 $\pm$ 0.0001
$\leq$70 mm^3	0.123 $\pm$ 0.086	0.099 $\pm$ 0.070	0.9999 $\pm$ 0.0001
$\leq$60 mm^3	0.089 $\pm$ 0.082	0.070 $\pm$ 0.065	0.9999 $\pm$ 0.0001
$\leq$50 mm^3	0.072 $\pm$ 0.082	0.055 $\pm$ 0.062	0.9999 $\pm$ 0.0001
$\leq$40 mm^3	0.064 $\pm$ 0.071	0.049 $\pm$ 0.054	0.9999 $\pm$ 0.0001
$\leq$30 mm^3	0.057 $\pm$ 0.084	0.046 $\pm$ 0.065	0.9999 $\pm$ 0.0001
$\leq$20 mm^3	0.051 $\pm$ 0.082	0.040 $\pm$ 0.064	0.9999 $\pm$ 0.0001
$\leq$10mm^3	0.015 $\pm$ 0.032	0.013 $\pm$ 0.028	1.0 $\pm$ 0.0

4.4 U-Net employed through fastMONAI

The U-Net segmentations were acquired from running a Python pipeline using functions from the fastMONAI library. The model used to predict the segmentations were trained on FLAIR NIfTI images. The results on the smaller lesions were calculated after the larger lesions were stripped from the segmentation.

4.4.1 Segmentation of all lesions

Table 4.7 shows the dice score, sensitivity, and specificity for the segmentations made by the fastMONAI U-Net. The dice scores for each of the subjects in the test set vary between 0.708 and 0.805, giving a mean score of 0.764 $\pm$ 0.039. The sensitivity values among the subjects range from 0.620 to 0.777, getting a mean score of 0.708 $\pm$ 0.047. The specificity has a mean score of 0.9999 $\pm$ 0.0001.

Table 4.7: Table showing the results from using the U-net architecture from fastMONAI in order to predict segmentations of the eight subjects in the test set. The table shows individual results as well as the mean scores for each performance measure.

Individual scores for U-Net								
	Subject 1	**Subject 2**	**Subject 3**	**Subject 4**	**Subject 5**	**Subject 6**	**Subject 7**	**Subject 8**
Dice Score	0.722	0.756	0.785	0.805	0.816	0.708	0.724	0.795
Sensitivity	0.657	0.712	0.731	0.719	0.777	0.697	0.620	0.748
Specificity	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
Overall scores								
Mean Dice Score	0.764 ± 0.039							
Mean Sensitivity	0.708 ± 0.047							
Mean Specificity	0.9999 ± 0.0001							

Figure 4.4 shows the lesion segmentations made by U-Net on the eight subjects in the test set. Notably, the U-Net outperforms the previously mentioned models, as evident by its high mean dice and sensitivity scores reported in Table 4.7. It seems to perform well across various brain regions, as indicated by the significant overlap observed in all subjects S1-S8. The network's lowest score can be seen in subject 1, where there are multiple smaller lesions present.

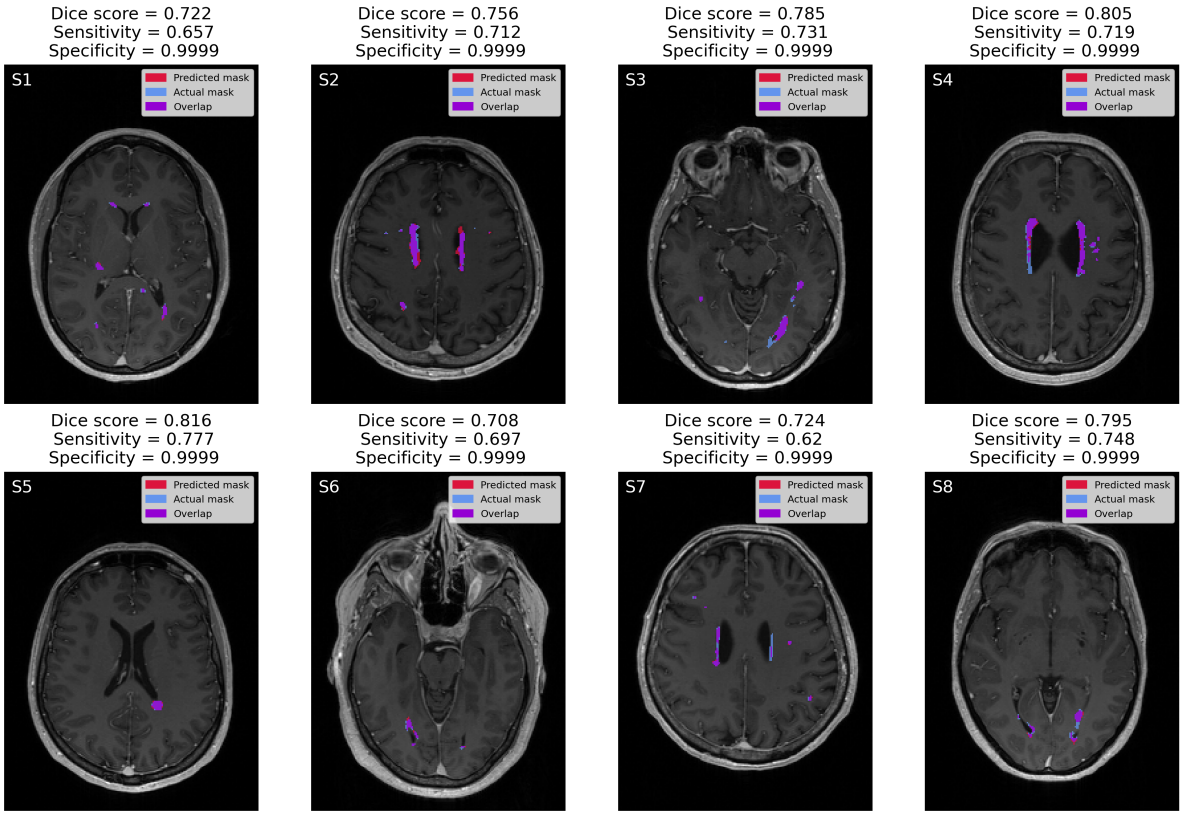


Figure 4.4: Image showing the lesion segmentations from the U-Net neural network of the eight subjects in the test set. The figures are labeled between S1-S8, corresponding to subjects 1-8. Above each subject are the associated dice score, sensitivity, and specificity. The ground true lesion masks are shown in blue, the predicted masks from the algorithm are shown in red, and their overlap is in purple.

4.4.2 Segmentation of small lesions

Table 4.8 shows the mean dice score, mean sensitivity and mean specificity of segmentations of decreasing lesion sizes. The segmentations containing lesions up to 100 mm^3 got a mean dice score of 0.579 $\pm$ 0.084, mean sensitivity of 0.532 $\pm$ 0.078, and mean specificity of 0.9999 $\pm$ 0.0001. The mean dice score and mean sensitivity decrease for all segmentations containing decreasing lesion sizes. Looking at the segmentations having lesions up to a size of 10 mm^3, the mean dice score is 0.129 $\pm$ 0.116, the mean sensitivity is 0.122 $\pm$ 0.109, and the mean specificity is 1.0 $\pm$ 0.0. An example of a U-Net prediction on small lesions can be seen in supplementary Figure B.2.

Table 4.8: Table showing the results from using the U-Net to predict segmentations of the eight subjects in the test set. The results shown are after larger lesions are removed from both the ground truth and predicted lesions. For each of the lesion sizes shown, all lesions above the mentioned volume are removed. The table shows the mean dice score, mean sensitivity, and mean specificity for lesion sizes smaller than 100mm^3 to 10mm^3 with 10 mm^3 increment.

Mean scores for U-Net on decreasing lesion sizes

	Mean dice score	**Mean sensitivity**	**Mean specificity**
≤100 mm^3	0.579 $\pm$ 0.084	0.531 $\pm$ 0.078	0.9999 $\pm$ 0.0001
≤90 mm^3	0.570 $\pm$ 0.109	0.532 $\pm$ 0.087	0.9999 $\pm$ 0.0001
≤80 mm^3	0.516 $\pm$ 0.153	0.495 $\pm$ 0.135	0.9999 $\pm$ 0.0001
≤70 mm^3	0.512 $\pm$ 0.125	0.503 $\pm$ 0.106	0.9999 $\pm$ 0.0001
≤60 mm^3	0.477 $\pm$ 0.123	0.477 $\pm$ 0.098	0.9999 $\pm$ 0.0001
≤50 mm^3	0.432 $\pm$ 0.169	0.446 $\pm$ 0.174	0.9999 $\pm$ 0.0001
≤40 mm^3	0.444 $\pm$ 0.158	0.433 $\pm$ 0.162	0.9999 $\pm$ 0.0001
≤30 mm^3	0.353 $\pm$ 0.155	0.342 $\pm$ 0.158	0.9999 $\pm$ 0.0001
≤20 mm^3	0.348 $\pm$ 0.144	0.281 $\pm$ 0.125	1.0 $\pm$ 0.0
≤10 mm^3	0.129 $\pm$ 0.116	0.122 $\pm$ 0.109	1.0 $\pm$ 0.0

4.5 nnU-Net

The nnU-Net segmentations were acquired by running the nnU-Net predict terminal command. The model used to predict the segmentations was trained on both T1 and FLAIR NIfTI images. The results on the smaller lesions were calculated after the larger lesions were removed from the segmentation.

4.5.1 Segmentation of all lesions

Table 4.9 shows the dice score, sensitivity, and specificity scores of the segmentations made by the nnU-Net algorithm. The dice scores vary between 0.567 and 0.916, having a mean score of 0.817 $\pm$ 0.099. The sensitivity varies between 0.545 and 0.949 giving the mean sensitivity 0.822 $\pm$ 0.114. Specificity has a constant value of 0.9999.

Table 4.9: Table showing the results from using the nnU-Net in order to predict segmentations of the eight subjects in the test set. The table shows individual results as well as the mean scores for each performance measure.

Individual scores for nnU-Net								
	Subject 1	**Subject 2**	**Subject 3**	**Subject 4**	**Subject 5**	**Subject 6**	**Subject 7**	**Subject 8**
Dice Score	0.567	0.842	0.850	0.916	0.832	0.821	0.870	0.841
Sensitivity	0.545	0.888	0.882	0.949	0.828	0.859	0.826	0.795
Specificity	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
Overall scores								
Mean Dice Score	0.817 ± 0.099							
Mean Sensitivity	0.822 ± 0.114							
Mean Specificity	0.9999 ± 0.0001							

Figure 4.5 shows the lesion segmentations made by nnU-Net on the eight subjects in the test set. The nnU-Net achieves its lowest score on subject S1, similar to that of the U-Net. For all the other subjects, there are large areas of overlap seen in purple. Subject S2 shows areas around the ventricles that are red, which means the model has predicted lesions where there are none. The opposite can be seen for subject 7, where the right ventricle is blue, meaning the model was not able to predict this lesion.

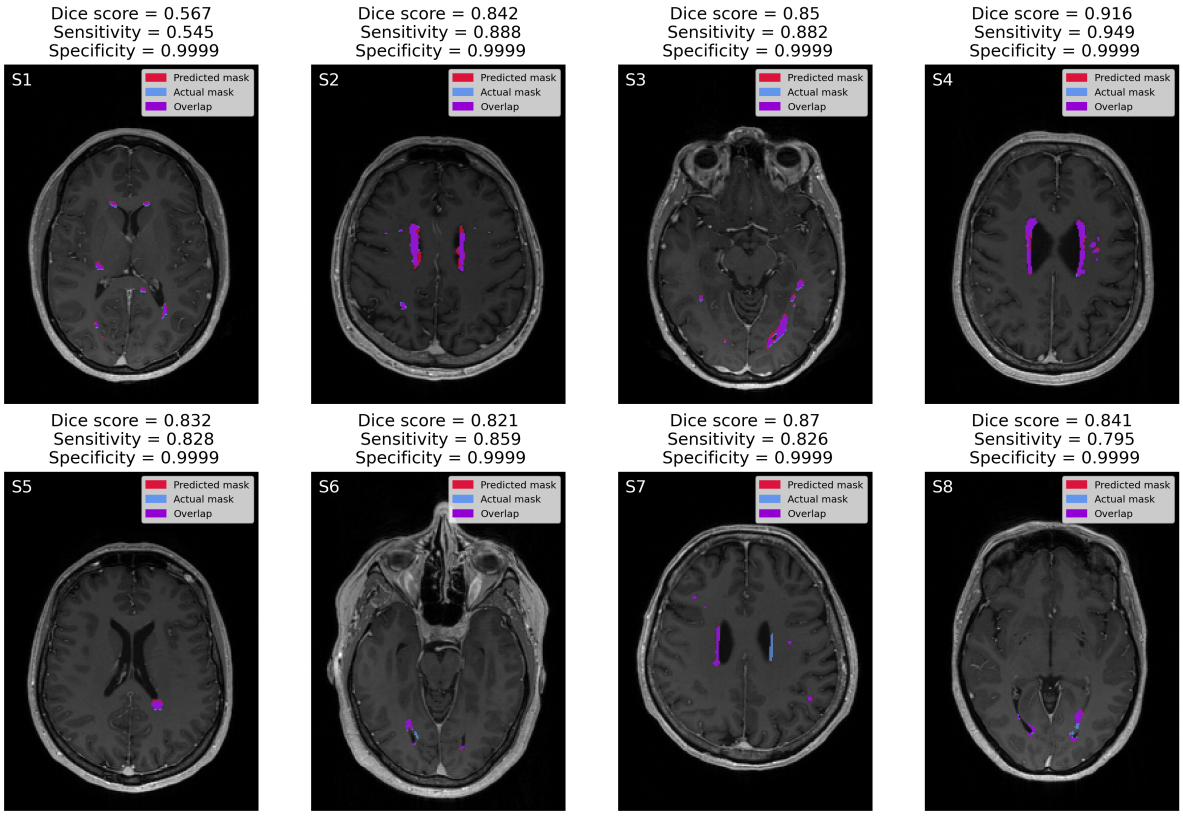


Figure 4.5: Image showing the lesion segmentations from the nnU-Net neural network of the eight subjects in the test set. The figures are labeled between S1-S8, corresponding to the subjects 1-8. Above each subject are the associated dice score, sensitivity, and specificity. The ground true lesion masks are shown in blue, the predicted masks from the algorithm are shown in red, and their overlap is in purple.

4.5.2 Segmentation of small lesions

Table 4.10 shows the mean dice, mean sensitivity, and mean specificity for decreasing lesion sizes. For the segmentations containing lesions up to 100 mm^3, the mean dice score is 0.686 ± 0.127, the mean sensitivity is 0.710 ± 0.143, and the mean specificity is 0.9999 ± 0.0001. For the decreasing lesion sizes, the different scores follow a decreasing trend as well. When looking at the segmentations containing lesions up to 10 mm^3, the mean dice score is 0.389 ± 0.153, the mean sensitivity is 0.395 ± 0.176, and the mean specificity is 1.0 ± 0.0. An example of a nnU-Net prediction on small lesions can be seen in supplementary Figure B.3.

Table 4.10: Table showing the results from using the nnU-Net to predict segmentations of the eight subjects in the test set. The results shown are after larger lesions are removed from both the ground truth and predicted lesions. For each of the lesion sizes shown, all lesions above the mentioned volume are removed. The table shows the mean dice score, mean sensitivity, and mean specificity for lesion sizes smaller than 100mm^3 to 10mm^3 with 10 mm^3 increment.

Mean scores for nnUnet on decreasing lesion sizes

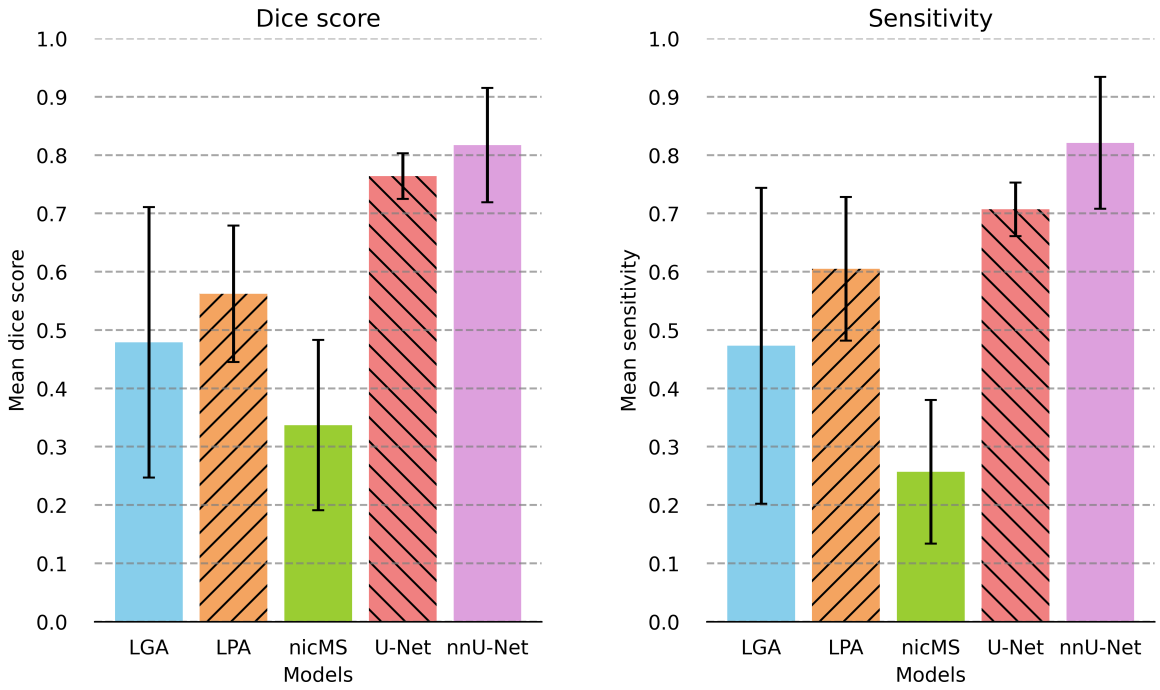
	Mean dice score	**Mean sensitivity**	**Mean specificity**
≤100 mm^3	0.686 ± 0.127	0.710 ± 0.143	0.9999 ± 0.0001
≤90 mm^3	0.666 ± 0.154	0.668 ± 0.157	0.9999 ± 0.0001
≤80 mm^3	0.667 ± 0.142	0.682 ± 0.156	0.9999 ± 0.0001
≤70 mm^3	0.663 ± 0.153	0.671 ± 0.184	0.9999 ± 0.0001
≤60 mm^3	0.609 ± 0.188	0.605 ± 0.230	0.9999 ± 0.0001
≤50 mm^3	0.586 ± 0.200	0.611 ± 0.253	0.9999 ± 0.0001
≤40 mm^3	0.565 ± 0.198	0.572 ± 0.240	0.9999 ± 0.0001
≤30 mm^3	0.548 ± 0.204	0.564 ± 0.223	0.9999 ± 0.0001
≤20 mm^3	0.494 ± 0.230	0.511 ± 0.275	0.9999 ± 0.0001
≤10 mm^3	0.389 ± 0.153	0.395 ± 0.176	1.0 ± 0.0

4.6 All model segmentations

This section will present the results of all the models side by side in order to compare them. Both results from the segmentations containing all lesions and the segmentations where larger lesions were removed will be shown.

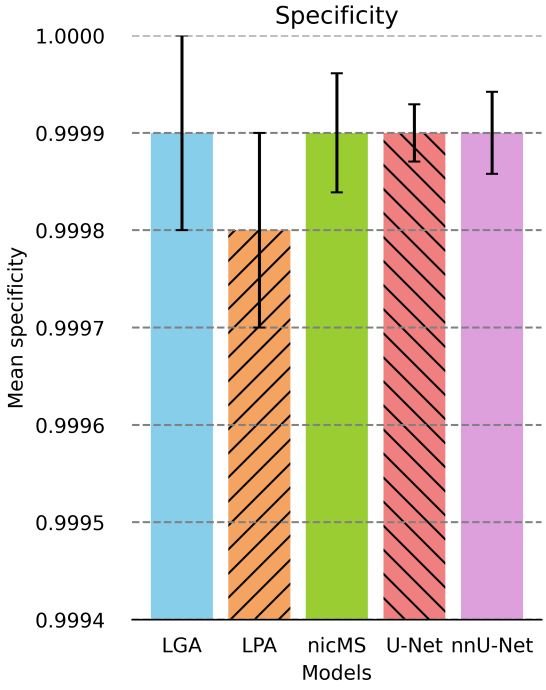
4.6.1 Segmentation of all lesions

The results showing the segmentations containing all lesions are visualized by bar plots seen in Figure 4.6. One bar plot was made for each of the evaluation metrics dice score, sensitivity, and specificity. The slanted lines on the bars of LPA and U-Net have been added for easier separation of the colored bars.



(a) Mean dice score

(b) Mean sensitivity



(c) Mean specificity

Figure 4.6: Figure containing three bar plots. Figure (a) shows the mean dice scores, Figure (b) shows the mean sensitivity, and Figure (c) shows the mean specificity of all five models used for producing image segmentations. The height of each bar represents the model's mean score and the error bar is the standard deviation. The lesion growth algorithm (LGA) is shown in blue, the lesion prediction algorithm (LPA) is shown in orange, the nicMSlesions algorithm is in green, the U-Net in red, and the nnU-Net in purple. The slanted lines seen in two of the bars is for better separation of the colors.

Dice score

The mean dice score of the five different models is displayed in Figure 4.6a. From the figure, it is evident that nicMSlesions has the lowest mean dice score of 0.337 $\pm$ 0.146 compared to the other model segmentations. It is also noteworthy that nicMSlesions has the second-largest error bar. LGA achieves a mean dice score of 0.479 $\pm$ 0.232, exhibiting the largest error bar among the models. LPA got a mean dice score of 0.562 $\pm$ 0.117, making it the model with the third-highest dice score. The U-Net model demonstrates the second-highest mean dice score of 0.764 $\pm$ 0.039, accompanied by the lowest standard deviation. The model with the highest mean dice score is nnU-Net, scoring 0.817 $\pm$ 0.099.

Sensitivity

The mean sensitivity of the five models can be seen in Figure 4.6b. The sensitivity bar plot follows the same trend and pattern as the DSC bar plot in Figure 4.6a. The nicMSlesions model has the lowest mean sensitivity of 0.258 $\pm$ 0.123. Next is LGA with a mean sensitivity of 0.473 $\pm$ 0.271, followed by LPA with a mean score of 0.605 $\pm$ 0.123. The U-Net has the second-highest mean sensitivity 0.708 $\pm$ 0.047, while the nnU-Net has the highest mean score of 0.822 $\pm$ 0.114.

Specificity

The mean specificities of the five models are depicted in Figure 4.6c. LGA, nicMSlesions, nnU-net, and U-Net all achieve a mean specificity of 0.9999, indicating a high level of specificity. On the other hand, LPA exhibits a slightly lower mean specificity of 0.9998, but has a large error bar falling within the range of the other models. U-Net has the lowest error bar of the five models, meaning it has the least variability in its results.

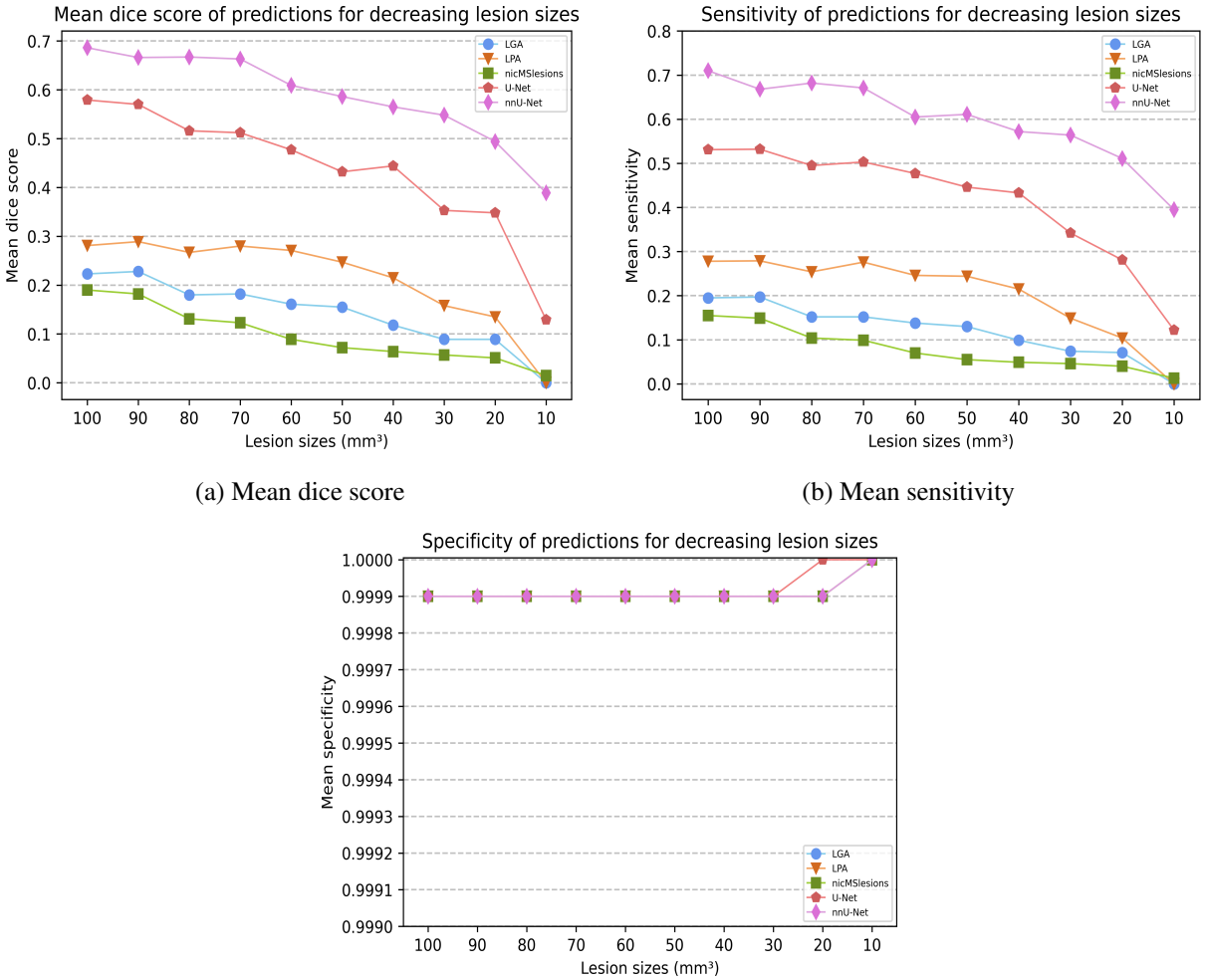
4.6.2 Segmentation of small lesions

This section presents the results from the segmentations on the small lesions. To get a better understanding of how the performance of the models changes when the size of the lesions in the segmentations decreases, the corresponding scores are shown in line plots.

Dice score

Figure 4.7a illustrates the mean dice scores computed for each model based on the decreasing lesion sizes. The figure demonstrates the impact of diminishing lesion sizes

on the evaluation metric. Notably, nnU-Net consistently achieves the highest dice score across all lesion sizes. U-Net attains the second-highest score and, alongside nnU-Net, is the only model that manages to have a prediction above 0.1 when looking at the segmentations containing lesions up to 10 mm^3. The models LGA, LPA, and nicMSlesions also have a general decrease with the decreasing lesion sizes, but when reaching the segmentations containing only small lesions, the dice score is ≈ 0.



(a) Mean dice score

(b) Mean sensitivity

(c) Mean specificity

Figure 4.7: Plot showing the mean performance measures calculated for the lesion sizes 100 mm^3 to 10 mm^3 with 10 mm^3 increment. Figure (a) shows the mean dice score, Figure (b) shows the mean sensitivity, and Figure (c) shows the mean specificity. The metrics were calculated for each of the five models and the result can be seen as the five lines in the plot. The result from LGA is shown in blue circles, the LPA is shown in yellow triangles, nicMSlesions are shown in green squares, U-Net is shown in red diamonds and nnU-Net is shown in purple pentagons.

Sensitivity

Figure 4.7b shows the sensitivity calculated for each model for each of the segmentations containing lesions of decreasing sizes. The sensitivity for each of the models

can be seen decreasing as the lesion sizes in the segmentations decrease as well. nnU-Net has the largest sensitivity across all lesion sizes, together with the U-Net, they are the only models that get a larger score for the segmentations containing lesions ≤ 10 mm^3. The LGA, LPA, and nicMSlesions show a gradual decrease in performance with respect to the decreasing lesion sizes and get a score of ≈ 0 for segmentations with lesions ≤ 10 mm^3.

Specificity

The mean specificity of the five models for each of the lesion sizes can be seen in Figure 4.7c. The figure shows that the performance of the models stays the same across the different lesion sizes. The U-Net gets a specificity of 1.0 when the segmentations contain lesions up to 20 mm^3, while this happens for the nnU-Net, LGA, LPA, and nicMSlesion models at 10 mm^3.

4.6.3 U-Net optimization

The plot produced in order to determine the learning rate to use for training the U-Net can be seen in Figure 4.8. The figure shows how the loss varies with the learning rate. The valley shown as an orange circle in the image indicates an area on the graph that includes a steep decline, and its corresponding value could therefore be used as a learning rate. This plot was produced before initiating training of the U-Net.

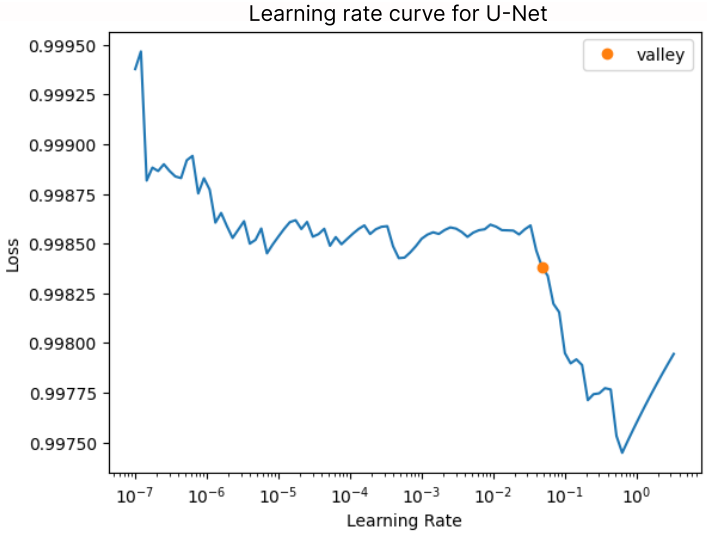


Figure 4.8: Plot visualizing how the loss depends on the learning rate. The orange circle indicates an area of steep decline on the graph. Since the plot was produced by an

Figure 4.9 shows how the training and validation loss for the U-Net drops as the number of epochs increases. Even though there seems to be more noise in the validation loss,

both losses follow the same trend, which is an indicator of the absence of over- and underfitting. The plot was produced after training as a way to indicate how well the training went.

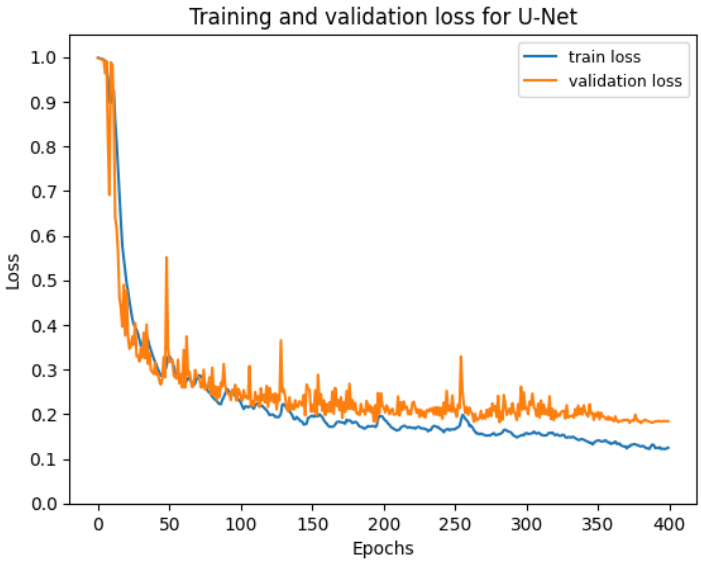


Figure 4.9: Plot showing how the training and validation loss changes as the number of epochs increases. The training loss is shown in blue, while the validation loss is in orange.

4.6.4 Training duration

The models that were trained on the project data all took different amounts of training time. This includes the nicMSlesions, U-Net, and nnU-Net and can be seen in Table 4.11. The times shown are in seconds per epoch and are knowledgeable estimates. As all models were trained for 400 epochs, the total amount in seconds is also shown.

Table 4.11: Table shows how long each of the trainable models used per epoch as well as total training time. Time per epoch is in seconds, while the total training time is in hours. Both measures are estimates as the exact time per epoch has some variation. The nnU-Net total training time is multiplied by 5 as the model had to be trained five times.

Model training times

	Time / epoch (s)	Total training time (h)
nicMSlesions	229	25
U-Net	26	3
nnU-Net	440	49 x 5

Chapter 5

Discussion

The primary objective of this project was to employ machine learning techniques for the automatic segmentation of brain lesions in patients with multiple sclerosis. Specifically, the focus was to assess the performance of these methods when dealing with small multiple sclerosis lesions. To achieve this, differently established algorithms and neural networks were utilized, namely LGA, LPA, nicMSlesions, and nnU-Net. Additionally, the U-Net architecture was employed using fastMONAI to train the segmentation model. The main findings reveal that both nnU-Net and U-Net demonstrated the ability to effectively segment small multiple sclerosis lesions, and highlight the potential of these models in identifying and delineating such lesions. In this chapter, the results obtained from Chapter 4 are examined, along with a discussion on potential sources of error or limitations in the methodology.

5.1 Small lesions

In the current thesis, small lesions were defined as those with a volume less than or equal to 10 mm^3. The primary emphasis of the thesis was to assess the performance of different models in segmenting such lesions. As demonstrated in the preceding chapter, not all models were capable of accurately segmenting these small lesions. Therefore, the evaluation of lesions ranging in size from 10 mm^3 to 100 mm^3 was added to reveal how the size of the lesions affects model performance.

It is also important to note that when dealing with lesion segmentations, the affected areas in the image are relatively small compared to the rest of the image. This is especially true for the small lesions. Consequently, any overlap in the segmentations carries significant weight in determining the performance. Random overlap can lead to misleading indications of a model's performance, necessitating visual inspection of the results to assess whether a model accurately predicts the correct lesion locations or

appears more random.

5.2 Model comparison

When working on getting the segmentations from each of the aforementioned models, an important focus was to try to make the process as equal as possible to have a fair comparison of the models. However, it is important to acknowledge that each method has its limitations in terms of customization. For instance, not all models were able to be trained on the specific data acquired for this project, and some did not accommodate the preferred input or yield the desired output. Nevertheless, all models successfully produced segmentations of MS lesions that could be utilized for calculating the performance measures.

5.2.1 Lesion growth algortihm

The LGA was one of the algorithms where training on the project data was impossible before running inference. This could be one of the reasons for its overall poor performance in segmenting lesions of all sizes, including small lesions (Table 4.1 and 4.2). Additionally, the LGA model was trained on images obtained from a different MRI scanner. Although both scanners were 3T, the specific sequence parameters, such as TE, TR, TI, and field of view, differed between the Philips scanner used for LGA and the Siemens Healthineers scanner used in this project. Consequently, the T1 and FLAIR images utilized for training and inference were dissimilar, impacting the algorithm's performance. The LGA algorithm was trained to recognize voxel patterns in a specific set of images, which may not generalize well with other image sets.

An important aspect of the LGA algorithm is the choice of the initial threshold value, denoted as κ, which enables adjustment for different image protocols. As different κ values yield different segmentation results, it becomes clear that the choice has much to say about the algorithm's performance. The Determination of optimal initial Threshold function was used to determine this value, and the κ chosen resulted in the highest dice score for each participant in the validation set. Because the κ was chosen based on what value had the highest frequency to give the best dice score on validation data only, it is still possible that another κ value could be best for the test set. Despite this, the κ was reasonably chosen with the approach recommended by the authors. The test set was not used for this process as it should remain unseen until the model inference phase to get a realistic performance measure.

The LGA could not find any small lesions in the segmentations containing lesions up to 10 mm^3, it also has the highest standard deviation meaning it is very inconsistent with its predictions (Table 4.2). The reasoning behind the lack of small lesion predictions could be a consequence of the LGA not being trained on the preferred data and there not being any certainty that the MRI images it was trained on contained small lesions. It is also possible that the algorithm used is unsuitable when wanting to detect smaller lesions, as the authors of the algorithm point out that they got lower DSC in patients with lower lesion volume [55].

The LGA was thresholded at a value of 0.1. Since the thresholded image was a probability map, the value is a number of how confident the algorithm is that there is a lesion present in the specified voxel. The range for these numbers is between zero and one, making 0.1 relatively low. It is an interesting observation that to get the highest dice score possible for the algorithm, voxels considered to have a low probability of being a lesion had to be included.

5.2.2 Lesion prediction algorithm

LPA is the other algorithm that was not readily trainable, which could have affected its performance on the data provided for the project. Correspondence with the author made it clear that it is possible to have the model be trained on other datasets. Because this required the data to be sent to the author, it was decided against on account of the General Data Protection Regulation (GDPR) [81]. There is also the possibility that training on new data won't change that much about the models' performance because the author has included a wide range of images for model training [56].

The machine and sequence used for the MRI image acquisition were the same as for the LGA. Still, it performs better than the LGA both when looking at all lesions and the small lesions (Figure 4.6 and 4.7). One reason for this can be that the LPA is a newer algorithm made by the same authors and build upon the knowledge acquired from working on their first algorithm. It also does not require an initial threshold that could be a source of error. LPA does have the lowest specificity, something that could be correlated to its possibility to overestimate lesion sizes [82]. A lot of overestimation would have a significant impact on the specificity value, so it could not have made that much of a difference as the standard deviation falls in between the mean specificities of the other models. Still, it is something to take notice of.

LPA also got a dice score and sensitivity of zero when looking for lesions ≤ 10 mm^3 (Table 4.4). One reason for this poor quality could be that the set used for training

only consists of images corresponding to patients with high total lesion volume (TLV), which the author has defined as TLV > 10 ml [56]. This further suggests that the training data may not have included an adequate number of small lesions. Additionally, the authors acknowledge that their segmentations may miss smaller lesions in the area between white matter and cortical gray matter [56]. In comparison to the other models, LGA and LPA were trained on fewer participants (52 and 53 patients) compared to nicMSlesions, nnU-Net, and U-Net, which were trained on at least 70 patients, indicating a difference of $\approx 26\%$ in training sample size.

5.2.3 nicMSlesions

The nicMSlesions algorithm was trained on top of their pre-trained model called baseline2ch. Doing this was expected to yield better performance of the model since the model used for inference would have been trained on more images, resulting in more lesions to learn from (Section 2.3.2). However, from Table 4.5, it can be seen that this was not the case, Figures 4.6a and 4.6b also show that the nicMSlesions algorithm has both the lowest DSC and the lowest sensitivity of the models tested during this project. One reason for this large difference could be that the algorithm was unable to train using different settings than the default on the computer available. This means that the setting that assumes the training data contains images of both healthy and sick brains was left unchanged. Because the model assumed the test set contained both healthy and sick MRI images, one of the images in the test set was considered healthy and was therefore not segmented. This can be seen in Table 4.5 for subject 5, where the dice score and sensitivity are zero. Visual inspection also showed that the probability map for this participant contained no segmentations. Since it was the mean scores of the eight participants that were used when plotting the Figures 4.6a and 4.6b, it is possible that this outlier value had an effect on the mean scores and likely portraits nicMSlesions worse than it would if all segmentations contained probabilities. This probably also had an effect on the large error bar.

For the small lesions, it can be seen from Table 4.6 that the algorithm has some overlapping voxels even for the ≤ 10 mm^3 segmentations. This was rather surprising as one of the default settings that could not be changed was the value deciding the smallest lesion output size, which was set to 10. Through visual inspection, it was shown that most of the ≤ 10 mm^3 segmentations contain no overlap, but since even the slightest overlap is enough to generate a score, the segmentations where this happens are enough to give a higher mean score. Even though there is an overlapping prediction there, it seems through visual inspection that this score is a result of two voxels randomly overlapping

instead of the algorithm actually predicting where the small lesions are located.

For a more fair comparison of the models, it would have been optimal to be able to train nicMSlesions on only the project data. It would also have been preferable to reduce the minimum lesion output size and to train the model to assume that all input images consist of MS lesions. Despite the inability to implement the necessary adjustments to customize the algorithm for the project's specific needs, it was still fascinating to observe how this widely recognized algorithm would perform.

5.2.4 U-Net

Tables 4.7 and 4.8 show the U-Net utilized through fastMONAI has good results for both segmentations containing all lesion sizes and for the small. It can also be seen from Figures 4.6a, 4.6b, and 4.6c that it has the smallest error bar. U-Net was possible to train on the project data, but could only be trained on either T1 or FLAIR images. The FLAIR images were chosen to use for training the U-Net because of their sensitivity for detecting white matter lesions (Section 2.1.4). Despite the model only being trained on one type of image, it still performed well on unseen data.

There were several values that had to be set, like the batch size, learning rate, loss function, and the number of epochs. As hyperparameters can affect a model's performance [83], it would have been interesting to do a hyperparameter search in order to find which assembly of hyperparameters would give the best score for this particular task. This was omitted as it would be a time-consuming and performance-heavy task. The hyperparameters were therefore selected manually with the knowledge available in order to produce the best possible result without the hyperparameter search. The learning rate was found using a function to plot the learning rate curve. The most optimal choice is to pick a value from where the curve has a steep decline. Figure 4.8 shows how the plot can be used to choose what learning rate to use for training. Even though there are several steep areas on the graph that could work as potential learning rates, the functions valley point, which is highlighted by an orange circle in the plot, was used as the developers found this to generally yield good results.

The other hyperparameters that were not as easily visualized as the learning rate, were chosen based on similar research within the field. A batch size of 4 was used as this was implemented for a similar task performed in the tutorial that the U-Net was built upon [10]. The loss function applied was dice loss because it is widely used in medical image segmentation [84] [85]. Dice loss also corresponds well the DSC evaluation metric, and it addresses the data imbalance problem which can occur when the voxels

that are supposed to be segmented constitute a small percentage of all the voxels in the image. This is especially useful for the small lesions, as they only make up a few voxels per image. Nonetheless, there is one type of imbalance that the dice loss does not consider, the imbalance between easy and hard samples [85]. Since the images used for the project contain varying amounts of lesions, and the size of the lesions varies greatly as well, it is difficult to make sure that the model is able to weigh the importance of the small lesions the same as the larger ones. Here it could have been interesting to test other loss functions such as focal dice loss which is supposed to counteract this issue [85] or Tversky loss which was shown to perform better than dice loss in this research article by Shruti Jadon [86]. Still, the U-Net performed relatively well on the small lesions (Table 4.8), meaning that the possible imbalance has not had a too negative impact.

The optimizer used for training was the ranger optimizer from fastAI. This optimizer was used in the fastMONAI tutorial [80]. It was kept in because of its ability to both accelerate the learning and to achieve a high validation accuracy without compromising generalization [87]. It also builds upon the Adam optimizer, which is among the most used optimizers in deep learning [88]. Here it would have been interesting to also try the Adadelta optimizer, as this was used for training the nicMSlesions model.

The U-Net was trained for 400 epochs in order for the trainable models to have been given the same foundation with regard to training time. Even though this was done, 400 is not necessarily a number that is going to fit all the models tested during this project. Since fastMONAI allows for customization of the code, the losses found during the training of the model were easily accessed. Figure 4.9 shows that both the training loss and validation loss decrease when the number of epochs increases. There is not a large gap between them and they don't show an increasing trend, they also flatten out after a number of epochs. This indicates that the model generalizes well and is neither overfitted nor underfitted [89], which is a good sign that the model is able to make good predictions on unseen data.

This method did not allow for probability maps as output. Because of this, the resulting segmentations did not go through the same validation pipeline as the rest. Being able to do this would most likely have altered the U-Net result for the better as all the other models except nnU-Net were thresholded at the values that gave the largest dice score.

Ashtari et al. [76] proposed a U-Net pipeline to perform the task of segmenting new MS lesions. Comparing the results from the small lesion segmentations made by the U-Net trained for this project to the one in the publication is difficult as they have not

defined a size for their new lesion segmentations. Ashtari et al. received an average dice score of 0.403 on lesions that developed between two time points. The U-Net trained for this project produces a similar mean dice score of 0.348 ± 0.144 when looking at the segmentation containing lesions ≤ 20 mm^3 (Table 4.8).

5.2.5 nnU-Net

The nnU-Net was the model that got the highest dice score and sensitivity, both when looking at all lesion sizes and only the small ones. This can be seen in Figure 4.6a, 4.6b, 4.7a, and 4.7b. The method is closely related to that of the U-Net, but because the whole process of training and testing is automated, it is less susceptible to user error. The pipeline also automates pre-processing, such as re-sampling and normalization, which can have a large effect on a model's performance. These are some of the reasons why the nnU-Net performed better than its predecessor as well as the other algorithms.

When looking at similar research regarding the nnU-Net and lesion segmentation it seems to perform better than expected. Looking at McKinley et al. [7] who got a mean dice score of 0.562, it was surprising to have the same network get such a larger score for the same task. Some reasons for this discrepancy could be that McKinley et al. [7] has trained and tested their model using images acquired from three different scanners. Their datasets are also split differently, with the training set containing 50 participants and 30 in the test set, as opposed to 70 and 8, which is used for this project. There is also no information given on if McKinley et al. [7] used the segmentations from thresholding the resulting probability maps or if they used the hard segmentation outputted from the model. This has an impact on performance as it decides what lesions are kept in when ultimately testing the model. Basaral et al. [5] who proposed a pipeline including the nnU-Net to segment new MS lesions, got an average dice score of 0.510. The only specification mentioned about size is that lesions below 3 mm^3 were excluded from the dataset used. This score is similar to how the nnU-Net trained for this project performed when looking at the segmentations containing lesions ≤ 20 mm^3 where the nnU-Net got a dice score of 0.494 ± 0.230 (Table 4.10).

When reviewing the results from the author of the nnU-Net, it seems that a large dice score is to be expected [9]. It is ranked high for all the segmentation challenges mentioned in the paper and gets a large dice score segmenting all types of organs. Looking specifically into the brain the nnU-Net got mean dice scores above 0.8 when segmenting brain tumors [90]. For MS lesion segmentation Isensee et al. [9] got a large score though it is not clear from the article what performance method was used or what competition this result is from.

One thing that sets the nnU-Net aside from the other models is the fact that it had to be trained for five folds to perform cross-validation. This could be a reason why the nnU-Net outperforms the other models. Cross-validation can be used to better generalize a model, and hence have an effect on the model's performance [91]. However, for this particular method, it seems that the nnU-Net performs cross-validation to carry out model selection between the possible configurations to use for model training [9]. Since the nnU-Net for this project was only trained with one configuration, this nullifies the need for model selection and therefore acts as a reason why the cross-validation could, in fact, not have had a large effect on the nnU-Nets performance.

The nnU-Net was the model that required the most time to train (Table 4.11), and training five folds for 400 epochs took slightly above two days each. Originally the nnU-Net was set to train for 1000 epochs as this is the default value recommended by the authors. Training the nnU-Net for 1000 epochs was estimated to take around 25 days in total, and was changed both with respect to time and to train the models for the same amount of epochs. The long training time is most likely an outcome due to the whole process being automated, which is dissimilar from the other models. The nicMSlesions also perform some automated pre-processing before training, which explains why the time per epoch was larger when compared to the U-Net.

5.3 Dataset

The dataset is what sets the foundation of what the trained models learn and how it interprets which features are important and not in each image. It also directly affects the performance as it is the images from the dataset that is used to evaluate, even for the models that were not trained for this specific project. The number of images, type of images, and content of these images are all factors that have a large effect on how the five models perform.

The main concern with this is that most models were not able to predict small lesions. Even though not all the evaluated models were made for this task in particular, it is possible that the poor performance was due to the dataset as well. Although all the images in the test and validation set were made sure to include small lesions, the ratio and number of these vary within each set. The images in the training set are also set to include small lesions, but not all of them. This could contribute to the trained model not considering the small lesions favorably, and training on another dataset specifically made to include small lesions could possibly skew this the other way. In spite of this, the necessary steps were taken to utilize the data in the best way possible for this project.

Steps such as having no participants appear in multiple datasets to avoid biased models, and having different variations of small lesions in the test and validation set means that the models were tested on a realistic variety of cases. Regardless of the fact that the dataset was not put together in a way that promotes getting a large score on small lesions, it is one that is representative of the MS disease and was used thereafter.

5.4 Validation pipeline

The validation pipeline was made as a way to comparatively threshold the probability maps outputted from most of the models. It was also a way to be more comprehensive about what lesions were included in the final segmentations. This was meant to hopefully also keep more small lesions in the model's predictions. This led to some of the probability maps being thresholded at low values, such as LPA, which was thresholded at 0.05. Working within the field of medicine, it is possible that it would have been more interesting to look at the segmentations that included more certain predictions. It would also have been fascinating to use the same threshold value across the models instead of optimizing them to better see the difference between the performances. A similar study [92], which compares LGA, LPA, and nicMSlesions among other models, used both optimization of threshold and the model's suggested default thresholds when comparing. This could have been a possible approach for this project as well, but with the focus being on small lesions, the choice of optimizing was, in the end, the one that seemed the best for this particular case.

5.5 Performance measure

5.5.1 Specificity

When looking at the specificity in Figures 4.6c and 4.7c, it is apparent that the specificity among the models had little variation despite the fact that the other scores showed larger differences. Since specificity describes the rate of true negatives divided by both true negatives and false positives (Equation 2.20), it can be misleading if the range between these numbers is large. The images used for calculating the performance measures are binary, meaning they only consist of zeros where there are no lesions and ones where there are. Because each segmentation image consists of a much larger quantity of zeros compared to ones, this contributes to a more skewed perception of how many false positives appear in the model predictions, from Figures 4.1, 4.2, 4.3, 4.4 and 4.5 there are some of the colored areas that are red. This signifies the presence of false positives, and from looking at these figures, it should constitute a larger difference when

compared to the actual specificity scores.

More interesting and less misleading performance measures could, in this case, be the negative predictive value (NPV), the false discovery rate (FDR), or the false omission rate (FOR). The NPV and FOR both indicate the rate between true negatives and false negatives, while the FDR shows how a large portion of both true and false positives are actually false.

5.5.2 Mean

In this thesis, the mean was chosen as a way to present the results of the different models. However, it was observed in Chapter 4 that certain models, namely LGA (Table 4.1), nicMSlesions (Table 4.5), and nnU-Net (Table 4.9), produced outliers in their predictions. The mean and standard deviation metrics are strongly impacted by outliers, and the smaller the sample size of the dataset, the more an outlier has the potential to affect the mean. To counteract this, the median and median absolute deviation could have been utilized to diminish the effect of the outliers on the performance of the models [93]. Nevertheless, the decision was made to use the mean, as it is a commonly used metric for presenting the arithmetic average of a dataset, and incorporating the standard deviation provides insight into the variability within the results. As presented in the previous chapter, the outliers were not numerous nor extreme compared to the other values. Although the median was evaluated, it did not significantly alter the results. Thus, the mean and standard deviation metrics were retained, as they effectively represent the acquired results.

Chapter 6

Conclusions and future work

During this thesis, five different models were tested for their ability to segment MS lesions automatically. While most of the focus has been on finding how well these models perform when looking at small lesions ($\leq$ 10 mm^3), performance on larger lesions has also been evaluated. Being able to detect small lesions is an important task to look into as they are a marker of disease activity. In addition, new multiple sclerosis lesions, which often appear small, are also the primary outcome measure for MS drug trials. Because the size of these lesions can make them difficult to observe, having models that automatically segment these types of lesions can greatly aid with both the diagnosis and treatment of MS.

The results obtained from this project indicate that small lesion segmentation is possible. Of the five models that were tested, two of them successfully predicted some of the lesions smaller than 10 mm^3, both having the U-Net architecture. The nnU-Net performed the best overall, showing the significance and effect of optimizing every aspect of the training and testing procedure. The other models LPA, LGA, and nicMSlesions, could not predict the small lesions and performed worse than U-Net and nnU-Net for all sizes. Reasons for this possibly include LPA and LGA being trained on images from different scanners and not being able to change the default settings of the nicMSlesions.

The study shows that the new and innovative methods within automatic segmentations work just as competently as the more well-established ones. It is clear that not only is the choice of algorithm or neural network important, but also the processes performed in advance. This can be seen with the nnU-Net, which outperforms the other models. Going forward, the focus should shift to not only optimizing the parameters of the model but also optimizing the work done for pre-processing. It was also discussed how the dataset is a factor in how well a model can perform, and how performance measures can paint the wrong picture if not looked into correctly.

For future work, one of the aspects that would be interesting to look more into would be how a dataset made to contain a larger amount of small lesions could affect the performance of the different models. It could also be interesting to include both healthy and sick participants to see if the models are able to distinguish between them, and at the same time, be able to predict small lesions. Additionally, it would be important to correlate the findings of small lesions to clinical symptoms in order to fully map the significance of finding the lesion load also of small lesions.

There is a trade-off with this sort of work, fully automated processes often take away from the understanding and freedom to make changes to the procedure. When merging the fields of artificial intelligence and medicine, a good algorithm can be brushed off because of its inaccessibility and it being incomprehensible to the user. More advanced technology should be usable, understandable, and open to change while still being able to perform well. Going forward, the gap between making something accessible and making something perform well ought to close. That way, people without much programming experience can still benefit from technological discoveries.

Appendix A

Source code

The source code for the project is publicly available on GitHub:

https://github.com/MariaMathea/Masterthesis_ML_MS

The repository contains all the relevant code created during this project. The code is made up of both Python code (.ipynb) and MATLAB scripts (.m). All code segments are explained with comments, and the GitHub link also contains a README file to further elaborate its contents. The repository has the following contents and structure:

- **Validation pipeline**
- **Testing pipeline**
- **U-Net**
- **nnU-Net utils**

  convert: Converts project dataset into nnU-Net format

  generate_json: Generates JSON file necessary for nnU-Net

  reorient_images: Makes the header of two NIfTY images the same

  save_npz_to_nifti: Generates a NIftI image from .npz and .pkl files

- **MATLAB scripts**

  counting_small_lesions: Counts the small lesions in a binary image

  iteration_lesion_removal: Removes lesions from a binary image

  lesion_volume: Returns mean TLV and mean number of lesions in a dataset

- **Visualizations**

  plot_scores: Produces barplot and lineplot

  plot_slices: Produce segmentation plots that show overlap and scores

Appendix B

Small lesion visualization

B.1 nicMSlesions

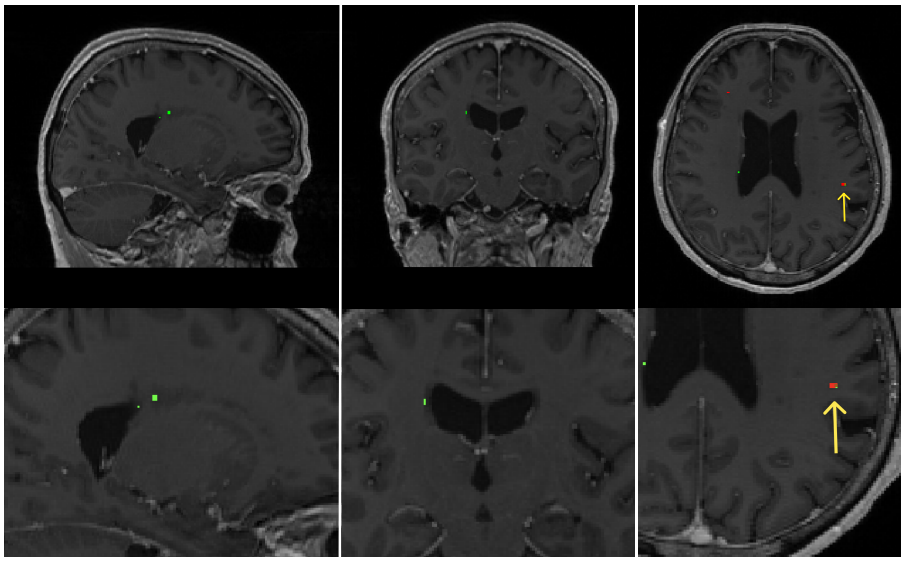


Figure B.1: Figure that shows the presence of overlap in one of the subjects in the test set. The green-colored prediction represents the segmentations made by nicMSlesions for lesions $\leq$ 10 mm^3. The figure displays a T1-weighted image from a single subject, viewed from three different perspectives. The first row presents the segmentation without any zoom, while the second row provides a closer view to highlight the overlapping region. The ground truth mask is depicted in red, and the area of overlap is represented with orange, which is also indicated with an arrow for clarity.

B.2 U-Net

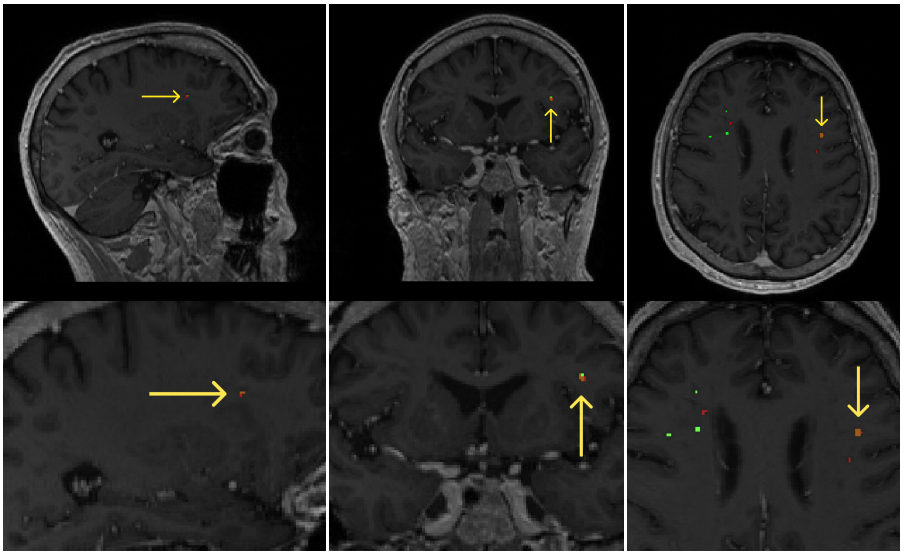


Figure B.2: Figure that shows the presence of overlap in one of the subjects in the test set. The green-colored prediction represents the segmentations made by U-Net for lesions $\leq$ 10 mm^3. The figure displays a T1-weighted image from a single subject, viewed from three different perspectives. The first row presents the segmentation without any zoom, while the second row provides a closer view to highlight the overlapping region. The ground truth mask is depicted in red, and the area of overlap is represented with orange, which is also indicated with an arrow for clarity.

B.3 nnU-Net

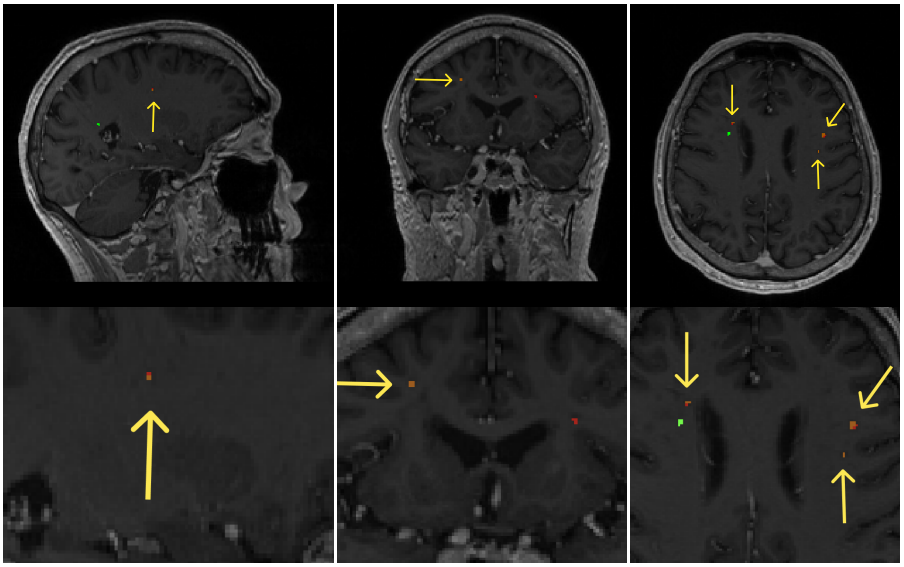


Figure B.3: Figure that shows the presence of overlap in one of the subjects in the test set. The green-colored prediction represents the segmentations made by nnU-Net for lesions $\leq$ 10 mm^3. The figure displays a T1-weighted image from a single subject, viewed from three different perspectives. The first row presents the segmentation without any zoom, while the second row provides a closer view to highlight the overlapping region. The ground truth mask is depicted in red, and the area of overlap is represented with orange, which is also indicated with an arrow for clarity.

Bibliography

[1] N. Ghasemi, S. Razavi, and E. Nikzad, "Multiple sclerosis: Pathogenesis, symptoms, diagnoses and cell-based therapy." *Cell J*, vol. 19, no. 1, pp. 1–10, Apr-Jun 2017. 1.1, 2.2, 2.2.1

[2] A. J. Thompson, S. E. Baranzini, J. Geurts, B. Hemmer, and O. Ciccarelli, "Multiple sclerosis," *The Lancet*, vol. 391, no. 10130, pp. 1622–1636, 2018. 1.1

[3] C. C. Hemond and R. Bakshi, "Magnetic resonance imaging in multiple sclerosis." *Cold Spring Harb Perspect Med*, vol. 8, no. 5, May 2018. 1.1, 2.2

[4] A. J. Thompson, B. L. Banwell, F. Barkhof, W. M. Carroll, T. Coetzee, G. Comi, J. Correale, F. Fazekas, M. Filippi, M. S. Freedman, K. Fujihara, S. L. Galetta, H. P. Hartung, L. Kappos, F. D. Lublin, R. A. Marrie, A. E. Miller, D. H. Miller, X. Montalban, E. M. Mowry, P. S. Sorensen, M. Tintoré, A. L. Traboulsee, M. Trojano, B. M. Uitdehaag, S. Vukusic, E. Waubant, B. G. Weinshenker, S. C. Reingold, and J. A. Cohen, "Diagnosis of multiple sclerosis: 2017 revisions of the mcdonald criteria," *The Lancet. Neurology*, vol. 17, pp. 162–173, 2 2018. 1.1

[5] B. D. Basaran, P. M. Matthews, and W. Bai, "New lesion segmentation for multiple sclerosis brain images with imaging and lesion-aware augmentation." *Front Neurosci*, vol. 16, p. 1007453, 2022. 1.1, 2.7, 5.2.5

[6] A. Shoeibi, M. Khodatars, M. Jafari, P. Moridian, M. Rezaei, R. Alizadehsani, F. Khozeimeh, J. M. Gorriz, J. Heras, M. Panahiazar, S. Nahavandi, and U. R. Acharya, "Applications of deep learning techniques for automated multiple sclerosis detection using magnetic resonance imaging: A review," *Computers in Biology and Medicine*, vol. 136, p. 104697, 2021. 1.1, 2.7

[7] R. McKinley, R. Wepfer, F. Aschwanden, L. Grunder, R. Muri, C. Rummel, R. Verma, C. Weisstanner, M. Reyes, A. Salmen, A. Chan, F. Wagner, and R. Wiest, "Simultaneous lesion and brain segmentation in multiple sclerosis using deep neural networks," *Scientific Reports*, vol. 11, p. 1087, 01 2021. 1.2, 5.2.5

[8] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241. 1.2, 2.5.5, 2.14

[9] F. Isensee, P. F. Jaeger, S. A. A. Kohl, J. Petersen, and K. H. Maier-Hein, "nnu-net: a self-configuring method for deep learning-based biomedical image segmentation," *Nature Methods*, vol. 18, no. 2, pp. 203–211, 2021. 1.2, 2.5.7, 2.5.7, 2.5.7, 3.4.4, 5.2.5

[10] S. Kaliyugarasan and A. S. Lundervold, "fastmonai: a low-code deep learning library for medical image analysis," 2022. 1.2, 2.5.6, 5.2.4

[11] F. Bloch, "Nuclear induction," *Phys. Rev.*, vol. 70, pp. 460–474, Oct 1946. 2.1

[12] 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. 2.1

[13] F. W. Smith, J. M. Hutchison, J. R. Mallard, G. Johnson, T. W. Redpath, R. D. Selbie, A. Reid, and C. C. Smith, "Oesophageal carcinoma demonstrated by whole-body nuclear magnetic resonance imaging." *British Medical Journal (Clin Res Ed)*, vol. 282, no. 6263, pp. 510–512, Feb 1981. 2.1

[14] V. P. B. Grover, J. M. Tognarelli, M. M. E. Crossey, I. J. Cox, S. D. Taylor-Robinson, and M. J. W. McPhail, "Magnetic resonance imaging: Principles and techniques: Lessons for clinicians." *J Clin Exp Hepatol*, vol. 5, no. 3, pp. 246–255, Sep 2015. 2.1, 2.1.1

[15] L. Loued-Khenissi, O. Döll, and K. Preuschoff, "An overview of functional magnetic resonance imaging techniques for organizational research," *Organizational Research Methods*, vol. 22, no. 1, pp. 17–45, 2019. 2.1.1

[16] E. R. Grüner, "Compendium phys212: Medical physics and technology, version 1.0," 2012, department of Physics and Technology, University of Bergen. 2.1.1, 2.1.1, 2.1.3, 2.1.3, 2.1.3, 2.1.4

[17] C. Westbrook and C. Kaut, *MRI in practice*, 2nd ed. Blackwell Science, 1998. 2.1.1, 2.1.1, 2.1.1, 2.1.2, 2.1.3, 2.1.3, 2.1.3, 2.1.3, 2.1.4, 2.1.4, 2.1.4

[18] A. Bjørnerud, "The physics of magnetic resonance imaging," 2006, department of Physics, University of Oslo. 2.1.2

[19] N. Ida, *Faraday's Law and Induction*. New York, NY: Springer New York, 2000, pp. 629–686. 2.1.2

[20] E. K. Lee, E. J. Lee, S. Kim, and Y. S. Lee, "Importance of contrast-enhanced fluid-attenuated inversion recovery magnetic resonance imaging in various intracranial pathologic conditions." *Korean J Radiol*, vol. 17, no. 1, pp. 127–141, Jan-Feb 2016. 2.1.4

[21] J. J. M. Zwanenburg, J. Hendrikse, F. Visser, T. Takahara, and P. R. Luijten, "Fluid attenuated inversion recovery (flair) mri at 7.0 tesla: comparison with 1.5 and 3.0 tesla." *Eur Radiol*, vol. 20, no. 4, pp. 915–922, Apr 2010. 2.1.4

[22] M. Brant-Zawadzki, D. Atkinson, M. Detrick, W. G. Bradley, and G. Scidmore, "Fluid-attenuated inversion recovery (flair) for assessment of cerebral infarction," *Stroke*, vol. 27, no. 7, pp. 1187–1191, 1996. 2.1.4

[23] D. Preston, "Magnetic resonance imaging (mri) of the brain and spine: Basics," Apr 2016, accessed 18.05.23. [Online]. Available: https://case.edu/med/neurology/NR/MRI%20Basics.htm 2.4

[24] A. Thompson, B. Banwell, F. Barkhof, W. Carroll, T. Coetzee, G. Comi, J. Correale, F. Fazekas, M. Filippi, M. Freedman, K. Fujihara, S. Galetta, H.-P. Hartung, L. Kappos, F. Lublin, R. Marrie, A. Miller, D. Miller, X. Montalban, and J. Cohen, "Diagnosis of multiple sclerosis: 2017 revisions of the mcdonald criteria," *The Lancet Neurology*, vol. 17, 12 2017. 2.2.2, 3.1

[25] A. Rovira, M. Wattjes, M. Tintorè, C. Tur, T. Yousry, M. P. Sormani, N. De Stefano, M. Filippi, C. Auger, M. Rocca, F. Barkhof, F. Fazekas, L. Kappos, C. Polman, D. Miller, X. Montalban, and J. Frederiksen, "Evidence-based guidelines: Magnims consensus guidelines on the use of mri in multiple sclerosis - clinical implementation in the diagnostic process," *Nature reviews. Neurology*, vol. 11, 07 2015. 2.2.2

[26] M. Filippi, P. Preziosa, B. L. Banwell, F. Barkhof, O. Ciccarelli, N. De Stefano, J. J. G. Geurts, F. Paul, D. S. Reich, A. T. Toosy, A. Traboulsee, M. P. Wattjes, T. A. Yousry, A. Gass, C. Lubetzki, B. G. Weinshenker, and M. A. Rocca, "Assessment of lesions on magnetic resonance imaging in multiple sclerosis: practical guidelines," *Brain*, vol. 142, no. 7, pp. 1858–1875, 06 2019. 2.2.2

[27] T. Hastie, R. Tibshirani, and J. Friedman, "The elements of statistical learning: data mining, inference, and prediction. springer series in statistics," *Springer New York*, 2009. 2.3

[28] R. C. Deo, "Machine learning in medicine," *Circulation*, vol. 132, no. 20, pp. 1920–1930, 2015. 2.3

[29] R. Sathya and A. Abraham, "Comparison of supervised and unsupervised learning algorithms for pattern classification," *International Journal of Advanced Research in Artificial Intelligence*, vol. 2, 02 2013. 2.3

[30] I. Despotovi, B. Goossens, and W. Philips, "Mri segmentation of the human brain: challenges, methods, and applications." *Comput Math Methods Med*, vol. 2015, p. 450341, 2015. 2.3.1, 2.3.1

[31] A. Di Ieva, C. Russo, S. Liu, A. Jian, M. Bai, Y. Qian, and J. Magnussen, "Application of deep learning for automatic segmentation of brain tumors on magnetic resonance imaging: a heuristic approach in the clinical scenario," *Neuroradiology*, vol. 63, 08 2021. 2.7

[32] S. D. Olabarriaga and A. W. Smeulders, "Interaction in the segmentation of medical images: a survey." *Med Image Anal*, vol. 5, no. 2, pp. 127–142, Jun 2001. 2.3.1

[33] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010. 2.3.2

[34] H. E. Kim, A. Cosa-Linan, N. Santhanam, M. Jannesari, M. E. Maros, and T. Ganslandt, "Transfer learning for medical image classification: a literature review," *BMC Medical Imaging*, vol. 22, no. 1, p. 69, 2022. 2.3.2

[35] J. A. M. Sidey-Gibbons and C. J. Sidey-Gibbons, "Machine learning in medicine: a practical introduction," *BMC Medical Research Methodology*, vol. 19, no. 1, p. 64, 2019. 2.3.3

[36] J. M. Kernbach and V. E. Staartjes, "Foundations of machine learning-based clinical prediction modeling: Part ii—generalization and overfitting," in *Machine Learning in Clinical Neuroscience*, V. E. Staartjes, L. Regli, and C. Serra, Eds. Cham: Springer International Publishing, 2022, pp. 15–21. 2.3.3

[37] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. 2.3.3

[38] A. N. Henderson, S. K. Kauwe, and T. D. Sparks, "Benchmark datasets incorporating diverse tasks, sample sizes, material systems, and data heterogeneity for materials informatics," *Data in Brief*, vol. 37, p. 107262, 2021. 2.3.3

[39] P. Chlap, H. Min, N. Vandenberg, J. Dowling, L. Holloway, and A. Haworth, "A review of medical image data augmentation techniques for deep learning applications," *Journal of Medical Imaging and Radiation Oncology*, vol. 65, no. 5, pp. 545–563, 2021. 2.3.4

[40] T. Oommen, L. G. Baise, and R. M. Vogel, "Sampling bias and class imbalance in maximum-likelihood logistic regression," *Mathematical Geosciences*, vol. 43, no. 1, pp. 99–120, 2011. 2.3.5

[41] E. Boateng and F. Oduro, "Predicting microfinance credit default: A study of nsoatreman rural bank ghana," *Advances in Mathematics and Computer Science*, no. 26, pp. 1–9, 2018. 2.3.5

[42] E. Boateng and D. Abaye, "A review of the logistic regression model with emphasis on medical research," *Data analysis and Information Processing*, no. 7, pp. 190–207, 2019. 2.3.5, 2.3.5

[43] "What is deep learning?: How it works, techniques, applications," May 2017, accessed: 05.05.23 from https://se.mathworks.com/discovery/deep-learning.html. [Online]. Available: https://se.mathworks.com/discovery/deep-learning.html 2.4

[44] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: concepts, cnn architectures, challenges, applications, future directions," *Journal of Big Data*, vol. 8, no. 1, p. 53, 2021. 2.4.1, 2.5.5

[45] H. Yu, L. T. Yang, Q. Zhang, D. Armstrong, and M. J. Deen, "Convolutional neural networks for medical image analysis: State-of-the-art, comparisons, improvement and perspectives," *Neurocomputing*, vol. 444, pp. 92–110, 2021. 2.4.1

[46] J.-G. Lee, S. Jun, Y.-W. Cho, H. Lee, G. B. Kim, J. B. Seo, and N. Kim, "Deep learning in medical imaging: General overview," *Korean Journal of Radiology*, vol. 18, no. 4, p. 570, May 2017. 2.4.1

[47] S. Balaji, "Binary image classifier cnn using tensorflow," Aug 2020, accessed: 01.02.23. [Online]. Available: https://medium.com/techiepedia/binary-image-classifier-cnn-using-tensorflow-a3f5d6746697 2.10

[48] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014. 2.4.1

[49] S. Afaq and S. Rao, "Significance of epochs on training a neural network," *International Journal of Scientific & Technology Research*, vol. 9, pp. 485–488, 2020. 2.4.2

[50] Q. Wang, Y. Ma, K. Zhao, and Y. Tian, "A comprehensive survey of loss functions in machine learning," *Annals of Data Science*, vol. 9, no. 2, pp. 187–212, 2022. 2.4.3

[51] Y. Tian, D. Su, S. Lauria, and X. Liu, "Recent advances on loss functions in deep learning for computer vision," *Neurocomputing*, vol. 497, pp. 129–158, 2022. 2.4.3

[52] R. Zaheer and H. Shaziya, "A study of the optimization algorithms in deep learning," in *2019 Third International Conference on Inventive Systems and Control (ICISC)*, 2019, pp. 536–539. 2.4.4

[53] D. Wilson and T. Martinez, "The need for small learning rates on large problems," in *IJCNN'01. International Joint Conference on Neural Networks. Proceedings (Cat. No.01CH37222)*, vol. 1, 2001, pp. 115–119 vol.1. 2.4.5

[54] "Spm12 software - statistical parametric mapping," Jan 2020, accessed: 01.09.22. [Online]. Available: https://www.fil.ion.ucl.ac.uk/spm/software/spm12/ 2.5.1

[55] P. Schmidt, "An automated tool for detection of flair-hyperintense white-matter lesions in multiple sclerosis," *NeuroImage*, vol. 59, pp. 3774–3783, 2 2012. 2.5.1, 2.5.2, 2.5.2, 2.5.2, 2.5.2, 2.13, 2.5.2, 2.5.3, 5.2.1

[56] ——, "Bayesian inference for structured additive regression models for large-scale problems with applications to medical imaging," Ph.D. dissertation, Ludwig-Maximilians-Universität München, January 2017. 2.5.1, 2.5.3, 2.5.3, 2.5.3, 2.5.3, 5.2.2

[57] M. Á. González Ballester, A. P. Zisserman, and M. Brady, "Estimation of the partial volume effect in mri," *Medical Image Analysis*, vol. 6, no. 4, pp. 389–405, 2002. 2.5.2

[58] A. Evans, D. Collins, S. Mills, E. Brown, R. Kelly, and T. Peters, "3d statistical neuroanatomical models from 305 mri volumes," in *1993 IEEE Conference Record Nuclear Science Symposium and Medical Imaging Conference*, 1993, pp. 1813–1817 vol.3. 2.5.3

[59] S. Valverde, M. Salem, M. Cabezas, D. Pareto, J. C. Vilanova, L. Ramió-Torrentà, Àlex Rovira, J. Salvi, A. Oliver, and X. Lladó, "One-shot domain adaptation in

multiple sclerosis lesion segmentation using convolutional neural networks," *NeuroImage: Clinical*, p. 101638, 2018. 2.5.4

[60] M. L. Gawne-Cain, S. Webb, P. Tofts, and D. H. Miller, "Lesion volume measurement in multiple sclerosis: How important is accurate repositioning?" *Journal of Magnetic Resonance Imaging*, vol. 6, no. 5, pp. 705–713, 1996. 2.5.4

[61] L. Nanni, M. Paci, S. Brahnam, and A. Lumini, "Comparison of different image data augmentation approaches." *J Imaging*, vol. 7, no. 12, Nov 2021. 2.5.4

[62] S. Valverde, M. Cabezas, E. Roura, S. González-Villà, D. Pareto, J. C. Vilanova, L. Ramió-Torrentà, À. Rovira, A. Oliver, and X. Lladó, "Improving automated multiple sclerosis lesion segmentation with a cascaded 3d convolutional neural network approach," *NeuroImage*, vol. 155, pp. 159–168, 2017. 2.5.4, 2.5.4, 2.5.4, 2.5.4, 2.5.4

[63] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814. 2.5.4

[64] M. D. Zeiler, "Adadelta: An adaptive learning rate method," *ArXiv*, vol. abs/1212.5701, 2012. 2.5.4

[65] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ser. ICML'15. JMLR.org, 2015, p. 448–456. 2.5.4

[66] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: Learning dense volumetric segmentation from sparse annotation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells, Eds. Cham: Springer International Publishing, 2016, pp. 424–432. 2.5.5

[67] F. Isensee, J. Petersen, A. Klein, D. Zimmerer, P. F. Jaeger, S. Kohl, J. Wasserthal, G. Koehler, T. Norajitra, S. Wirkert, and K. H. Maier-Hein, "nnu-net: Self-adapting framework for u-net-based medical image segmentation," 2018. 2.5.7

[68] A. A. Taha and A. Hanbury, "Metrics for evaluating 3d medical image segmentation: analysis, selection, and tool." *BMC Med Imaging*, vol. 15, p. 29, Aug 2015. 2.6

[69] D. Müller, I. Soto-Rey, and F. Kramer, "Towards a guideline for evaluation metrics in medical image segmentation." *BMC Res Notes*, vol. 15, no. 1, p. 210, Jun 2022. 2.6

[70] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945. 2.6.1

[71] A. Zijdenbos, B. Dawant, R. Margolin, and A. Palmer, "Morphometric analysis of white matter lesions in mr images: method and validation," *IEEE Transactions on Medical Imaging*, vol. 13, no. 4, pp. 716–724, 1994. 2.6.1

[72] K. M. Ting, *Sensitivity and Specificity*. Boston, MA: Springer US, 2010, pp. 901–902. 2.6.2

[73] R. Parikh, A. Mathai, S. Parikh, G. Chandra Sekhar, and R. Thomas, "Understanding and using sensitivity, specificity and predictive values." *Indian J Ophthalmol*, vol. 56, no. 1, pp. 45–50, Jan-Feb 2008. 2.6.2

[74] N. Aslam, I. U. Khan, A. Bashamakh, F. A. Alghool, M. Aboulnour, N. M. Alsuwayan, R. K. Alturaif, S. Brahimi, S. S. Aljameel, and K. Al Ghamdi, "Multiple sclerosis diagnosis using machine learning and deep learning: Challenges and opportunities," *Sensors*, vol. 22, no. 20, 2022. 2.7

[75] F. Moazami, A. Lefevre-Utile, C. Papaloukas, and V. Soumelis, "Machine learning approaches in study of multiple sclerosis disease through magnetic resonance images," *Frontiers in Immunology*, vol. 12, 2021. 2.7

[76] P. Ashtari, B. Barile, S. Van Huffel, and D. Sappey-Marinier, "New multiple sclerosis lesion segmentation and detection using pre-activation u-net," *Frontiers in Neuroscience*, vol. 16, 2022. 2.7, 5.2.4

[77] M. Battaglini, F. Rossi, R. A. Grove, M. L. Stromillo, B. Whitcher, P. M. Matthews, and N. De Stefano, "Automated identification of brain new lesions in multiple sclerosis using subtraction images," *Journal of Magnetic Resonance Imaging*, vol. 39, no. 6, pp. 1543–1549, 2014. 2.7

[78] E. Skorve, A. J. Lundervold, Ø. Torkildsen, F. Riemer, R. Grüner, and K.-M. Myhr, "Brief international cognitive assessment for ms (bicams) and global brain volumes in early stages of ms – a longitudinal correlation study," *Multiple Sclerosis and Related Disorders*, vol. 69, p. 104398, 2023. 3.1

[79] Neubias, "Connected component labeling," 2023, accessed: 01.05.23. [Online]. Available: https://neubias.github.io/training-resources/connected_components/index.html 3.1

[80] S. Kaliyugarasan and A. S. Lundervold, "Binary semantic segmentation," 2022, accessed: 02.11.22. [Online]. Available: https://fastmonai.no/tutorial_binary_segmentation.html 3.4.3, 5.2.4

[81] European Parliament and Council of the European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council. [Online]. Available: https://data.europa.eu/eli/reg/2016/679/oj 5.2.2

[82] J. M. Waymont, C. Petsa, C. J. McNeil, A. D. Murray, and G. D. Waiter, "Validation and comparison of two automated methods for quantifying brain white matter hyperintensities of presumed vascular origin," *Journal of International Medical Research*, vol. 48, no. 2, p. 0300060519880053, 2020, pMID: 31612759. 5.2.2

[83] X. Jiang and C. Xu, "Deep learning and machine learning with grid search to predict later occurrence of breast cancer metastasis using clinical data," *Journal of Clinical Medicine*, vol. 11, no. 19, 2022. 5.2.4

[84] J. Ma, J. Chen, M. Ng, R. Huang, Y. Li, C. Li, X. Yang, and A. L. Martel, "Loss odyssey in medical image segmentation," *Medical Image Analysis*, vol. 71, p. 102035, 2021. 5.2.4

[85] R. Zhao, B. Qian, X. Zhang, Y. Li, R. Wei, Y. Liu, and Y. Pan, "Rethinking dice loss for medical image segmentation," in *2020 IEEE International Conference on Data Mining (ICDM)*, 2020, pp. 851–860. 5.2.4

[86] S. Jadon, "A survey of loss functions for semantic segmentation," in *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, 2020, pp. 1–7. 5.2.4

[87] L. Wright and N. Demeure, "Ranger21: a synergistic deep learning optimizer," *CoRR*, vol. abs/2106.13731, 2021. 5.2.4

[88] R. M. Schmidt, F. Schneider, and P. Hennig, "Descending through a crowded valley - benchmarking deep learning optimizers," *CoRR*, vol. abs/2007.01547, 2020. 5.2.4

[89] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, "Dive into deep learning," *CoRR*, vol. abs/2106.11342, 2021. 5.2.4

[90] F. Isensee, P. F. Jäger, P. M. Full, P. Vollmuth, and K. H. Maier-Hein, "nnu-net for brain tumor segmentation," in *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, A. Crimi and S. Bakas, Eds. Cham: Springer International Publishing, 2021, pp. 118–132. 5.2.5

[91] D. Berrar, "Cross-validation," in *Encyclopedia of Bioinformatics and Computational Biology*, S. Ranganathan, M. Gribskov, K. Nakai, and C. Schönbach, Eds. Oxford: Academic Press, 2019, pp. 542–545. 5.2.5

[92] M. Weeda, I. Brouwer, M. Vos, M. Vries, F. Barkhof, P. Pouwels, and H. Vrenken, "Comparing lesion segmentation methods in multiple sclerosis: Input from one manually delineated subject is sufficient for accurate lesion segmentation," *NeuroImage: Clinical*, vol. 24, p. 102074, 11 2019. 5.4

[93] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, "Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median," *Journal of Experimental Social Psychology*, vol. 49, no. 4, pp. 764–766, 2013. 5.5.2