Deep Learning for Medical Image Segmentation

Dissertation von Anne-Marie Rickmann

2023

Aus der Klinik und Poliklinik für Kinder- und Jugendpsychiatrie
Klinik der Ludwig-Maximilians-Universität München
Vorstand: Prof. Dr. med. Gerd Schulte-Körne

Deep Learning for Medical Image Segmentation

Dissertation
zum Erwerb des Doktorgrades der Naturwissenschaften
an der Medizinischen Fakultät der
Ludwig-Maximilians-Universität zu München

vorgelegt von
Anne-Marie Rickmann

aus
Schleswig

Jahr
2023

Mit Genehmigung der Medizinischen Fakultät
der Ludwig-Maximilians-Universität München

Betreuer:	Prof. Dr. rer. nat. Christian Wachinger
Zweitgutachter:	Prof. Dr. rer. nat. Michael Ingrisch
Dekan:	Prof. Dr. med. Thomas Gudermann
Tag der mündlichen Prüfung:	08. Januar 2024

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Dekanat Medizinische Fakultät
Promotionsbüro

Eidesstattliche Versicherung

Rickmann, Anne-Marie

Name, Vorname

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Dissertation mit dem Thema

Deep Learning for Medical Image Segmentation

selbständig verfasst, mich außer der angegebenen keiner weiteren Hilfsmittel bedient und alle Erkenntnisse, die aus dem Schrifttum ganz oder annähernd übernommen sind, als solche kenntlich gemacht und nach ihrer Herkunft unter Bezeichnung der Fundstelle einzeln nachgewiesen habe.

Ich erkläre des Weiteren, dass die hier vorgelegte Dissertation nicht in gleicher oder in ähnlicher Form bei einer anderen Stelle zur Erlangung eines akademischen Grades eingereicht wurde.

München, 10.01.2024

Ort, Datum

Anne-Marie Rickmann

Unterschrift Doktorandin/Doktorand

Abstract

Medical image segmentation represents a fundamental aspect of medical image computing. It facilitates measurements of anatomical structures, like organ volume and tissue thickness, critical for many classification algorithms which can be instrumental for clinical diagnosis. Consequently, enhancing the efficiency and accuracy of segmentation algorithms could lead to considerable improvements in patient care and diagnostic precision.

In recent years, deep learning has become the state-of-the-art approach in various domains of medical image computing, including medical image segmentation. The key advantages of deep learning methods are their speed and efficiency, which have the potential to transform clinical practice significantly. Traditional algorithms might require hours to perform complex computations, but with deep learning, such computational tasks can be executed much faster, often within seconds.

This thesis focuses on two distinct segmentation strategies: voxel-based and surface-based. Voxel-based segmentation assigns a class label to each individual voxel of an image. On the other hand, surface-based segmentation techniques involve reconstructing a 3D surface from the input images, then segmenting that surface into different regions.

This thesis presents multiple methods for voxel-based image segmentation. Here, the focus is segmenting brain structures, white matter hyperintensities, and abdominal organs. Our approaches confront challenges such as domain adaptation, learning with limited data, and optimizing network architectures to handle 3D images. Additionally, the thesis discusses ways to handle the failure cases of standard deep learning approaches, such as dealing with rare cases like patients who have undergone organ resection surgery.

Finally, the thesis turns its attention to cortical surface reconstruction and parcellation. Here, deep learning is used to extract cortical surfaces from MRI scans as triangular meshes and parcellate these surfaces on a vertex level. The challenges posed by this approach include handling irregular and topologically complex structures.

This thesis presents novel deep learning strategies for voxel-based and surface-based medical image segmentation. By addressing specific challenges in each approach, it aims to contribute to the ongoing advancement of medical image computing.

Zusammenfassung

Die Segmentierung medizinischer Bilder stellt einen fundamentalen Aspekt der medizinischen Bildverarbeitung dar. Sie erleichtert Messungen anatomischer Strukturen, wie Organvolumen und Gewebedicke, die für viele Klassifikationsalgorithmen entscheidend sein können und somit für klinische Diagnosen von Bedeutung sind. Daher könnten Verbesserungen in der Effizienz und Genauigkeit von Segmentierungsalgorithmen zu erheblichen Fortschritten in der Patientenversorgung und diagnostischen Genauigkeit führen.

Deep Learning hat sich in den letzten Jahren als führender Ansatz in verschiedenen Bereichen der medizinischen Bildverarbeitung etabliert. Die Hauptvorteile dieser Methoden sind Geschwindigkeit und Effizienz, die die klinische Praxis erheblich verändern können. Traditionelle Algorithmen benötigen möglicherweise Stunden, um komplexe Berechnungen durchzuführen, mit Deep Learning können solche rechenintensiven Aufgaben wesentlich schneller, oft innerhalb von Sekunden, ausgeführt werden.

Diese Dissertation konzentriert sich auf zwei Segmentierungsstrategien, die voxel- und oberflächenbasierte Segmentierung. Die voxelbasierte Segmentierung weist jedem Voxel eines Bildes ein Klassenlabel zu, während oberflächenbasierte Techniken eine 3D-Oberfläche aus den Eingabebildern rekonstruieren und segmentieren.

In dieser Arbeit werden mehrere Methoden für die voxelbasierte Bildsegmentierung vorgestellt. Der Fokus liegt hier auf der Segmentierung von Gehirnstrukturen, Hyperintensitäten der weißen Substanz und abdominellen Organen. Unsere Ansätze begegnen Herausforderungen wie der Anpassung an verschiedene Domänen, dem Lernen mit begrenzten Daten und der Optimierung von Netzwerkarchitekturen, um 3D-Bilder zu verarbeiten. Darüber hinaus werden in dieser Dissertation Möglichkeiten erörtert, mit den Fehlschlägen standardmäßiger Deep-Learning-Ansätze umzugehen, beispielsweise mit seltenen Fällen nach einer Organresektion.

Schließlich legen wir den Fokus auf die Rekonstruktion und Parzellierung von kortikalen Oberflächen. Hier wird Deep Learning verwendet, um kortikale Oberflächen aus MRT-Scans als Dreiecksnetz zu extrahieren und diese Oberflächen auf Knoten-Ebene zu parzellieren. Zu den Herausforderungen dieses Ansatzes gehört der Umgang mit unregelmäßigen und topologisch komplexen Strukturen.

Diese Arbeit stellt neuartige Deep-Learning-Strategien für die voxel- und oberflächenbasierte medizinische Segmentierung vor. Durch die Bewältigung spezifischer Herausforderungen in jedem Ansatz trägt sie so zur Weiterentwicklung der medizinischen Bildverarbeitung bei.

Acknowledgments

First and foremost I would like to thank my supervisor Prof. Christian Wachinger, for all the discussions, always being there and taking time to talk to his students, for giving me room to explore but also helping me to stay on track and of course for all his help in writing papers.

Next, I want to thank all my colleagues at AI-Med for the invaluable collaboration over the last four years. Our mutual support through the pandemic and during the crunch times of conference deadlines has been greatly appreciated. I've also really enjoyed the times we traveled to conferences together. Thanks for contributing positively to both my professional and personal experiences during this period. I'd like to especially thank Abhijit Guha Roy for motivating me to pursue a PhD, introducing me to the group, and of course for starting team segmentation which I was happy to continue during the last years. Ignacio Sarasua for sharing 3 years of this journey with me, and always being one phase ahead. Sebastian Pölsterl, for enforcing Fancy Fridays with me, sharing a good taste in food, for teaching all of us the importance of pretty tables, and for taking care of everything administrative. Fabian Bongratz for motivating me to continue working on cortical surface reconstruction and of course for joining me in team segmentation, being a great colleague, and being a constant in our often changing offices. Nuno Wolf for pushing me to get out of my post-pandemic comfort zone and introducing me to good coffee. The new members of AI-Med - Mori, Bailiang, and Yitong - who introduced new spirit and topics to the group. The amazing students I got to work with who contributed to several workshop and conference papers.

I want to thank all my friends in Munich, especially Nanda, Paula, Izabela and Agnieszka, for countless girls nights, cooking nights and honest conversations. Chrissi, Katja, and Anton, for our endless game nights and hiking trips. Shabnam and Amin for being some of the best people I know and always giving me food. I'm especially grateful to Agnieszka Tomczak, for sharing this experience with me, going through similar highs and lows, for always listening to my complaining, and laughing about it when I exaggerated too much. For asking the right questions, for inspiring me so much, for endless walks and encouraging my newfound interests in birds and pizza-making.

Finally, I want to thank my parents, who despite the distance have always listened to me, cheered me on and encouraged me to keep going. My sister, who has not only shown keen interest in my work but also ensured I maintained a balance by reminding me to take breaks. My brother, whose sense of humour has been a source of joy and relief. To my entire family, your combined support has helped me to push through challenges. I am deeply thankful for all of you.

Contents

Part I

Introduction and Fundamentals

1 Introduction

1.1 Motivation

Medical image segmentation is an essential process that aids in the diagnosis, prognosis, and treatment of a wide range of medical disorders. It involves dividing medical images into different regions or structures, such as organs, tissues, or pathologies. It is essential for downstream applications like volumetric measurements, tumor growth monitoring, or detecting structural changes due to aging or diseases like Alzheimer's.

However, segmentation of medical images remains a time-consuming and labor-intensive procedure that requires the expertise of trained professionals, such as radiologists. Despite the availability of software tools and automated algorithms, the process can still take many hours, depending on the image resolution and the number of classes.

Deep learning algorithms have demonstrated great promise in medical image processing in recent years. These algorithms are high-speed and sometimes even outperform human experts when trained with large amounts of data [89].

This thesis explores the potential of deep learning in medical image segmentation, specifically focusing on two segmentation types. The first type is voxel-based segmentation, which assigns class labels to individual voxels in the image, making it useful for tasks such as brain tissue or abdominal organ segmentation. The second type is surface-based segmentation, which involves extracting the boundaries of structures as continuous surfaces, providing a geometrically and topologically accurate representation, like in cortical surface reconstruction. Figure 1.1 compares voxel-based and surface-based segmentation, as applied to a brain Magnetic Resonance Imaging (MRI) scan. The right cerebral hemisphere, depicted on the left side of the figure, illustrates voxel-based segmentation, displaying a range of segmented brain structures. The left hemisphere, shown on the right side of the figure, demonstrates surface-based segmentation of the outer boundary of the cortex. A closer inspection of these regions reveals a key distinction between the two methods: surface-based segmentation allows boundary delineation between voxel boundaries, unlike voxel-based segmentation, where the grid-like structure of voxels is clearly visible. This fundamental difference underscores the unique capabilities and characteristics inherent to each segmentation strategy.

Overall, this thesis provides novel deep learning-based solutions for medical image segmentation that can significantly reduce the time and effort required for the segmentation process while maintaining high accuracy and robustness.

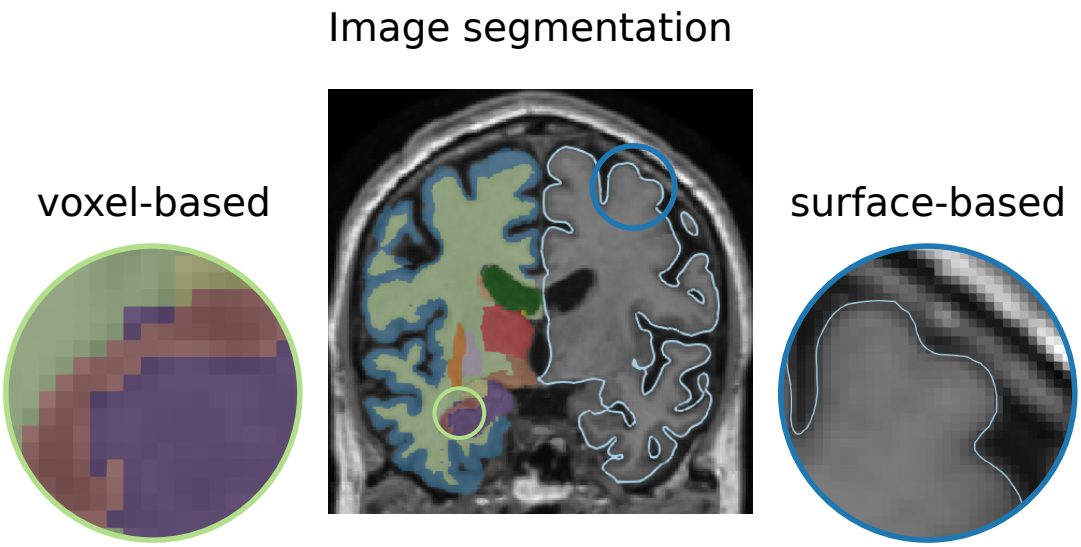

Figure 1.1. MRI scan of the brain with an example of voxel-based segmentation and surface-based segmentation.

1.2 Objectives

The primary goal is to develop and improve deep learning algorithms for medical image segmentation to replace time-consuming traditional methods.

Voxel-based segmentation: The first topic this thesis deals with is voxel-based segmentation. The objective is to develop robust and efficient deep learning algorithms for various medical segmentation tasks, specifically subcortical brain segmentation, abdominal organ segmentation, and white matter hyperintensity segmentation. The focus is on improving the accuracy of 3D segmentation models for subcortical brain segmentation and abdominal organ segmentation, specifically in high class imbalance settings for multi-class segmentation. In the case of white matter hyperintensity segmentation, the challenge lies in effectively utilizing a relatively small labeled dataset to segment a considerably larger, diverse dataset acquired from various scanners. This necessitates a focus on the issue of domain adaptation due to the discrepancies between data sources. Finally, we explore deep learning methods for segmentation after organ resection surgery. These methods are tested and evaluated in large-scale population studies. The intention is to demonstrate the feasibility of deploying deep learning-based solutions in real-world clinical scenarios, often involving patients from diverse demographic backgrounds and with varying anatomies, such as post-surgery changes.

Surface-based segmentation: The second focus of this thesis is surface-based segmentation. The goal is to construct and parcellate cortical surfaces from MRI scans using advanced deep learning techniques. The emphasis here is on improving the accuracy of these techniques and topological correctness, which can facilitate the diagnosis of neurological conditions, as shown on the example of Alzheimer's disease. Another objective is to facilitate group comparisons by providing surfaces with vertex correspondence.

Through these objectives, this thesis aims to advance the field of medical image segmentation, offering efficient, accurate, and robust solutions for a wide range of medical image segmentation tasks.

1.3 Contributions

Deep learning has significantly advanced medical image segmentation in recent years, improving accuracy and segmentation times. However, these methods still face various challenges, such as optimizing network architecture for volumetric input images. In addition, domain adaptation is necessary in medical imaging because it is not always possible to access previously utilized patient data due to privacy concerns. To address these challenges, we propose three different methods.

Recalibrating 3D ConvNets with Project & Excite [135]:
In this work, we focus on voxel-based subcortical and abdominal segmentation. We introduced Project & Excite (PE) blocks, which recalibrate feature maps in both channel-wise and spatial-wise manners, specifically tailored for 3D networks. We assess these modules on two demanding tasks: MRI brain scan segmentation and whole-body Computed Tomography (CT) scan segmentation. Our findings demonstrate that our approach improves Dice scores while only slightly increasing the model's complexity.

STRUDEL: Self-Training with Uncertainty Dependent Label Refinement across Domains [50]:
This work introduces an unsupervised domain adaptation method tailored to segment white matter hyperintensities (WMH) in brain MRI scans. Our method combines self-training with a focus on the uncertainty associated with pseudo-label predictions. By employing an uncertainty-guided loss function, we integrate the uncertainty into the training process, emphasizing labels that are predicted with high certainty. Recognizing that initial pseudo-labels are often noisy, we further enhance our method by integrating the output from third-party software into the generation of pseudo-labels. This combination leads to an improvement in WMH segmentation when compared to traditional self-training methods.

HALOS: Hallucination-free Organ Segmentation after Organ Resection Surgery [137]:
We address the clinically significant issue of dealing with images featuring atypical anatomy, such as post-organ resection surgery scans. State-of-the-art segmentation models frequently result in false-positive organ predictions, referred to as organ hallucinations. In this work, we introduce Hallucination-free Organ Segmentation (HALOS), a method specifically tailored for handling cases after organ resection surgery. Our approach combines two tasks: classifying the presence of organs and multi-organ segmentation. The classification branch assists the segmentation branch, helping it to make more accurate decisions. Feature fusion modules that incorporate the classification output into the segmentation process facilitate the communication between the two tasks. We illustrate the efficacy of our methodology by conducting experiments on a relatively small labeled test set as well as on data from the UK Biobank on a larger scale. Notably, it improves segmentation accuracy, as measured by Dice scores, and reduces the rate of false positive predictions to nearly zero.

Cortical surface reconstruction from MRI scans is an important task in neuroimaging for analyzing brain morphology and detecting diseases such as Alzheimer's. While deep learning-based algorithms provide accurate cortical surface reconstruction, they are often time-consuming and require post-processing. Additionally, it is necessary to establish vertex correspondence between a patient's cortical mesh and a group template for comparing cortical thickness and other measures at the vertex level. Furthermore, parcellation of the cortical surfaces into individual brain regions is necessary for a fine-grained analysis of atrophy patterns. In this context, we present three methods that tackle these challenges.

Vox2Cortex: Fast Explicit Reconstruction of Cortical Surfaces from 3D MRI Scans with Geometric Deep Neural Networks [10]:
In this work, we present Vox2Cortex, a network to generate meshes that accurately depict the boundaries of the cortex. Vox2Cortex combines two types of neural networks, convolutional and graph convolutional networks. Based on an input MRI scan, the method entails deforming a starting template mesh to match the intricately folded shape of the cortex. Through a series of experiments conducted on three separate brain MRI datasets, we validate the efficacy of our method. The meshes created by Vox2Cortex are precise and topologically correct, matching the accuracy of those produced by existing methods. A significant advantage of our method is that it bypasses the laborious and resource-intensive post-processing steps often required in conventional approaches.

Joint Reconstruction and Parcellation of Cortical Surfaces [132]:
We present two innovative options to enhance template-based deformation algorithms for generating surface meshes with atlas-based brain parcellation. One option employs a graph classification branch, while the other uses a new generic 3D reconstruction loss. We obtain accurate parcellations by integrating these approaches with our previously introduced Vox2Cortex model and another state-of-the-art method for cortical surface reconstruction. Additionally, the surfaces we generate maintain a quality comparable to what is achieved by the existing state-of-the-art methods.

Vertex Correspondence in Cortical Surface Reconstruction [133]:
This work proposes a novel approach for learning vertex correspondence by optimizing an L1 loss on registered surfaces, in contrast to the more common Chamfer loss. Our approach yields improved intra- and inter-subject correspondence, making it well-suited for direct group comparisons and atlas-based parcellation. We demonstrate that existing state-of-the-art methods provide inadequate correspondence for accurately mapping parcellations, underscoring the significance of optimizing for precise vertex correspondence.

1.4 Outline

The thesis is structured in four parts:

- **Part I: Introduction and Fundamentals (Chapter 1 and 2):** This part establishes the foundation for the rest of the thesis. It introduces the topic, provides background information, and outlines the main goals and contributions of the work. The fundamental concepts related to medical image segmentation and deep learning are also discussed, along with an introduction to the specific challenges of voxel-based and surface-based segmentation.

- **Part II: Voxel-Based Image Segmentation (Chapters 3 to 8):** This part focuses on voxel-based image segmentation. It begins with introducing the topic and the specific methods under discussion. The following chapters then detail the development and implementation of novel methods designed to address challenges like class imbalance, domain gap, and limited access to annotated data. These proposed methods are summarized and evaluated in the closing chapter of this section.

- **Part III: Surface-Based Image Segmentation (Chapters 9 to 14):** This part explores surface-based image segmentation. It commences with an introduction to the topic and provides the needed background information. Subsequent chapters then present proposed methods for the fast explicit reconstruction of cortical surfaces, joint cortical surface reconstruction and parcellation, and improving vertex correspondence in cortical surface reconstruction. A summary and evaluation of these methods are provided in the final chapter of this part.

- **Part IV: Conclusion (Chapter 15):** The conclusion of the thesis provides a summary of the key findings and contributions of the research.

2 Medical Image Segmentation: History and Challenges

Contents

Medical image analysis encompasses a wide range of techniques aimed at extracting meaningful information from medical images to facilitate diagnosis, treatment planning, and research. It involves the application of computer algorithms and mathematical models to analyze, interpret, and manipulate medical image data obtained from various imaging modalities. Typical tasks in medical image analysis are image segmentation, registration, and classification. Combining these techniques can form computer-aided diagnosis (CAD) systems or facilitate image-guided interventions.

Medical image analysis begins with image acquisition, where medical imaging devices capture data to create digital representations of anatomical structures or physiological functions. Different imaging modalities offer unique imaging capabilities suited for various clinical scenarios.

Accurate and precise segmentation is essential in medical image analysis as it provides quantitative measurements, facilitates visualization, enables feature extraction, and aids in treatment planning and monitoring. It is a prerequisite for many other tasks within the field of medical image analysis, such as registration, quantitative analysis, computer-aided diagnosis, image-guided interventions, and surgical planning.

This chapter provides the necessary background knowledge to understand the methods and approaches presented in this thesis. It covers the history and basics of medical image segmentation, the basics of deep learning, and current challenges in the field.

2.1 Imaging Modalities

Various imaging modalities are available, such as XRay, ultrasound, positron emission tomography (PET), Computed Tomography (CT), and Magnetic Resonance Imaging (MRI), each with specific strengths and applications. In this thesis, I focus on MRI and CT.

2.1.1 Magnetic Resonance Imaging (MRI)

MRI is a non-invasive imaging modality that uses a combination of strong magnetic fields, radio waves, and gradient fields to generate detailed images of the body's internal structures. MRI has excellent soft tissue contrast and does not expose patients to ionizing radiation. Several MRI sequences are commonly used, including:

T1-weighted (T1w)

T1-weighted images emphasize differences in T1 relaxation times, providing good anatomical detail. These images depict fat as bright and water as dark, making them suitable for visualizing soft tissues and brain structures.

T2-weighted (T2w)

T2-weighted images emphasize differences in T2 relaxation times, with fluid appearing bright and fat appearing dark. This contrast is beneficial for identifying edema, inflammation, and other pathological changes in tissues.

Fluid-Attenuated Inversion Recovery (FLAIR)

Fluid-Attenuated Inversion Recovery (FLAIR) is a modified T2-weighted sequence that suppresses the signal from cerebrospinal fluid (CSF), making it easier to identify pathological changes in brain tissues, such as multiple sclerosis lesions or white matter hyperintensities. The top row of Figure 2.1 shows axial, coronal, and sagittal views of a T1 weighted brain MRI scan and the axial slice of the corresponding FLAIR scan of the same subject.

Dixon

The Dixon method is used in MRI to separate water and fat signals in images. This technique is handy for generating fat-suppressed images and can be applied to various MRI sequences, such as T1w or T2w. An example of a Dixon sequence of the abdomen is presented in the bottom row of Figure 2.1.

2.1.2 Computed Tomography (CT)

CT is an imaging modality that generates cross-sectional images of the body using a series of X-ray projections captured from various angles. CT provides excellent spatial resolution and is particularly well-suited for imaging bone and lung structures. CT images can also be enhanced using contrast agents, which are injected into the body to improve visualization of blood vessels and specific organs, such as the liver or kidneys. However, CT exposes

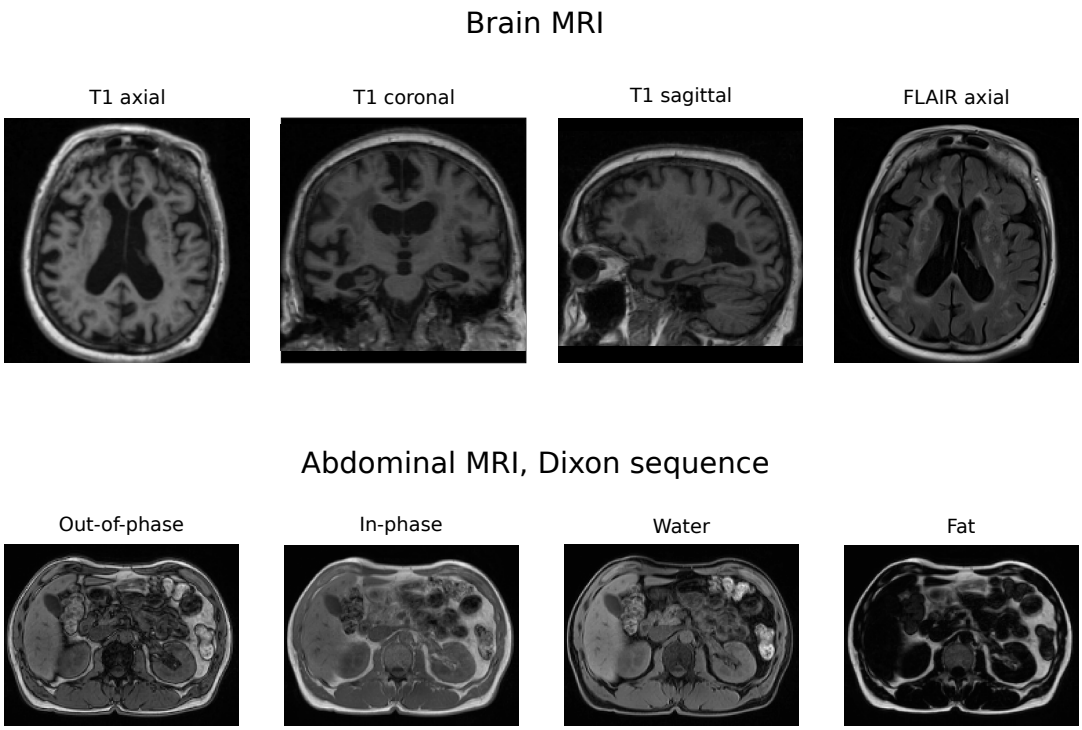

Figure 2.1. Top row: different views of T1 and FLAIR MRI scans of the brain. Bottom row: Axial view of the abdomen's Dixon MRI sequence.

patients to ionizing radiation, which should be considered when selecting an appropriate imaging modality. The Hounsfield Unit (HU) is a unique advantage of CT scans over other imaging modalities. Radiodensity is measured in Hounsfield Units, which are derived from a linear transformation of the original linear attenuation coefficient measurement. This unit is calibrated on a scale where air has a value of -1 000 HU, water has a value of 0 HU, and compact bone has a value of approximately 1 000 HU. Tissues in the human body fall within this scale, each with a specific range of Hounsfield Units. This numeric and uniform scale gives CT a significant advantage in quantitative image analysis. Unlike MRI, where voxel intensities are relative and vary between scans, the Hounsfield Units in CT scans provide a consistent and absolute scale, making the results more reliably comparable between different scans and patients.

2.2 Image Segmentation

Image segmentation is a fundamental task in computer vision that involves dividing an image into meaningful regions or objects. This process is crucial for various applications, including object recognition, tracking, and localization. Segmentation can be broadly categorized into three types: *Semantic segmentation*: Assigns a class label to each pixel in the image, focusing on delineating specific structures or regions of interest. In this thesis, I focus on semantic segmentation for medical image analysis. *Instance segmentation*: Extends semantic segmentation by identifying and distinguishing between multiple instances of the same class within the image. *Panoptic segmentation*: Combines semantic and instance segmentation, simultaneously

assigning class labels to pixels while distinguishing between individual instances of the same class.

2.2.1 Medical Image Segmentation and Applications

Medical image segmentation is the process of partitioning medical images, such as X-rays, CT, and MRI, into different regions or structures. The resulting segmented images can provide valuable information for clinical diagnosis, treatment planning, and disease monitoring. For example, segmenting MRI scans of the brain can help detect and quantify changes in cortical thickness, white matter integrity, and ventricular volume, which are valuable biomarkers for Alzheimer's disease, multiple sclerosis, and other neurodegenerative disorders.

Medical image segmentation is challenging due to various factors, such as image noise, artifacts, variability in anatomical structures, and the presence of pathologies. In addition, manual segmentation by specialists is time-consuming, costly, and subject to intra- and inter-observer variability. Consequently, precise and robust automated segmentation methods are required. In this thesis, I focus on several applications: subcortical brain segmentation, abdominal organ segmentation, and white matter hyperintensity segmentation in Part II, and surface-based segmentation of the cerebral cortex in Part III.

2.2.2 History of Medical Image Segmentation

Medical image segmentation has its roots in the late 1970s [32], and from the beginning, there were two different approaches, intensity-based or voxel-based approaches and boundary-based or surface-based approaches. I will provide a brief overview of the history of medical image segmentation, but refer the reader to the following surveys [32, 85, 89, 126] for detailed information. I divide the history of medical image segmentation into two categories: traditional methods, meaning non-machine-learning methods and machine learning and deep learning methods. One approach that does not fit into this categorization is atlas-based segmentation. Here segmentation is treated as a registration problem, where an atlas image with labels is given, typically a population average, and the atlas labels are transferred to the target image after registration. Registration can also be done by traditional or deep learning methods, but as this thesis does not focus on registration methods, I will omit these here. I will also use atlas-based segmentation techniques in Part III, about surface-based segmentation.

Traditional Methods

Traditional Methods are often complemented by signal processing methods used for pre-processing, like smoothing techniques such as anisotropic diffusion [125], which enhance the images by reducing noise, while preserving edges before applying segmentation algorithms.

Voxel-based techniques constitute some of the earliest methods, leveraging individual voxel characteristics for segmentation. Region-growing and thresholding are classic examples of such methods. Region growing begins from a seed point, expanding the region iteratively based on predefined criteria like pixel intensity or connectivity. On the other hand, thresholding

methods, such as Otsu's method [120], select pixel intensity thresholds to segregate different regions either globally or adaptively.

Surface-based techniques took a slightly different approach, focusing on detecting and utilizing edges and boundaries between regions. Initially, these methods utilized pixel intensity discontinuities or gradients for edge detection[32]. Soon after, more sophisticated surface-based methods, including deformable models such as active contours and level sets, began to take shape [32, 103]. Deformable models exploited shape and appearance priors to segment, match, and track anatomic structures.

Snakes [74], or active contour models, are curves within the domain of an image. These curves adapt and change their shape due to internal and external forces, allowing them to conform to prominent features such as edges and lines within the image. On the other hand, the level set method [119] is another influential surface-based technique, where the object's boundary to be segmented is represented as the zero level set, typically a Signed Distance Function (SDF). In SDFs, the value at any given point signifies the shortest distance to the contour or surface, and its sign indicates if the point is inside or outside the contour. This approach enhances the adaptability of deformable models, as it allows topological changes during the deformation process.

These advancements in conventional segmentation methodologies have been instrumental in paving the way for the subsequent emergence and evolution of machine learning and deep learning techniques in medical image segmentation.

Machine Learning and Deep Learning Methods

The rise of machine learning brought a significant shift in the paradigm for medical image segmentation. Traditional machine learning techniques like clustering [118], decision trees, and support vector machines (SVMs) found their application in this field [32]. They can be applied in voxel-based and surface-based segmentation, although their usage is more prevalent in voxel-based approaches.

For voxel-based segmentation, machine learning techniques often extract handcrafted features from individual voxels, such as texture descriptors or local statistical properties, and then use these features to train classifiers that predict class labels for each voxel. Examples include k-means clustering, which partitions voxels into groups based on their features, and SVMs, which learn hyperplanes in the feature space to separate voxels of different classes.

Despite the power and flexibility of machine learning methods, a significant challenge was the reliance on handcrafted features, which required expert knowledge and often did not generalize well across different tasks or datasets. This limitation has been largely overcome with the advent of deep learning methods, which can learn hierarchical feature representations directly from the input images. In recent years, deep learning, particularly Convolutional Neural Networks (CNNs), has revolutionized medical image segmentation, achieving state-of-the-art performance in various tasks. Deep learning methods learn hierarchical feature representations from the input images, eliminating the need for handcrafted feature extraction. In voxel-based segmentation, architectures such as U-Net [140] have gained popularity. More details on this type of architecture are given in Chapter 4. Besides CNNs, transformer models,

which initially gained prominence in natural language processing applications, have recently emerged as powerful tools for voxel-based segmentation. With their proficiency in grasping long-range dependencies and spatial contexts, transformer models have proven well-suited for medical image analysis [149].

On the surface-based segmentation front, deep learning methods have provided new ways to extract and represent surfaces and shapes within medical images. Examples include approaches for surface-based cortex parcellation [184] or deep learning techniques that operate directly on surfaces by combining CNNs with Graph Neural Networks (GNNs) [174]. Further details about these methods are given in Chapter 10.

2.3 Deep Learning

As this thesis focuses on deep learning methods, it is essential to provide a background on machine learning with a specific emphasis on deep learning. Machine learning focuses on creating algorithms that learn from data and utilize this learning to make informed predictions or decisions. In medical imaging, machine learning algorithms have been widely applied to various tasks such as segmentation, registration, and classification. These algorithms often rely on handcrafted features extracted from the input images and employ a training process to learn optimal parameters specific to the task.

Deep learning is an area within machine learning. It employs artificial neural networks with multiple layers, called deep neural networks. These networks have the capability to learn complex patterns and hierarchical features from large datasets. This enables them to perform excellently in various fields, including medical imaging. In contrast to conventional machine learning techniques, deep learning algorithms autonomously learn the most relevant features from the input data, eliminating the need for handcrafted feature engineering.

2.3.1 Neural Networks

Neural networks modeled after the structure and function of biological neurons in the human brain are the foundation of deep learning. A neural network consists of layers of artificial neurons, also known as nodes or units, that are interconnected. Each neuron receives input signals, performs computations, and produces an output signal, which is then passed on to the next layer of neurons.

An Multi-Layer Perceptron (MLP), a type of neural network, consists of at least three layers of nodes: the input layer, one or more hidden layers, and the output layer. The nodes in each layer have connections, characterized by weights, to all nodes in the subsequent layer, which characterizes the network as fully connected. The hidden layers are responsible for learning complex representations and allow the network to identify and process patterns and correlations within the data. The output layer is tasked with generating the ultimate predictions or decisions, which are derived from the calculations computed in the earlier layers.

The training of a neural network involves a process called backpropagation, which allows the network to adjust its parameters to minimize the difference between its predictions and the desired outputs. During the training phase, a loss function is used to quantify the difference between the predicted and actual outputs. The backpropagation algorithm propagates this error back through the network, adjusting the weights and biases of the neurons based on the calculated gradients.

By iteratively updating the network's parameters through the backpropagation process, the neural network gradually improves its ability to make accurate predictions or decisions. Combined with the vast amount of training data, this iterative learning process enables deep neural networks to learn complex representations and generalize well to unseen data effectively.

Convolutional and Graph Convolutional Neural Networks

In medical image segmentation, CNNs have emerged as a prominent deep learning architecture. CNNs are designed to capture spatial patterns and local dependencies within the input data. They achieve this through convolutional, pooling, and fully connected layers, enabling them to learn hierarchical representations directly from the image data.

Furthermore, GNNs have shown promise for tasks involving graph-structured data, such as surface-based segmentation. GNNs extend the concept of CNNs to graph domains and leverage graph structures to model relationships between data points. They have been successfully applied to tasks that involve surfaces, meshes, or other irregular data representations.

In the subsequent chapters of this thesis, the fundamentals of both CNNs and GNNs will be explored in more detail, providing a deeper understanding of their architectures, training processes, and applications in voxel-based and surface-based medical image segmentation.

2.3.2 Learning Paradigms

Machine learning and deep learning algorithms can be categorized into several learning paradigms. These paradigms differ in how they utilize available data for training, and each has specific applications in medical image segmentation.

Supervised learning is the most common paradigm, where algorithms are trained using labeled data. In medical image segmentation, this involves training a model to map input images to their corresponding segmentation maps. Deep learning methods like CNNs have achieved state-of-the-art performance in this paradigm. However, a disadvantage of supervised learning is its dependence on extensive sets of labeled data, the acquisition of which can be both resource-intensive and time-consuming.

Semi-supervised learning leverages both labeled and unlabeled data. This paradigm is useful when labeled data is limited or expensive to obtain. In medical image segmentation, semi-supervised learning involves, for example, training an initial model on labeled data and refining it using unlabeled data. *Weakly-supervised learning* focuses on training models with limited or noisy annotations instead of precise annotations. This paradigm assumes that obtaining precise

pixel-level annotations for training data is challenging or time-consuming. Instead, weaker forms of supervision, such as image-level labels, bounding boxes, or partial annotations, are used to guide the learning process.

Unsupervised learning involves training algorithms using only unlabeled data. In the context of medical image segmentation, unsupervised learning techniques often involve clustering or generative modeling approaches. *Self-supervised learning* can be used for pre-training segmentation models. It leverages the inherent structure or context within the data itself to define supervisory signals for training. Self-supervised tasks are, for example, inpainting or pixel shuffling.

2.4 Challenges

Although deep learning methods for medical image segmentation have shown great potential, these methods are not without challenges. This section discusses some critical challenges when applying deep learning algorithms to medical image segmentation. These challenges are not unique to image segmentation but are relevant to deep learning approaches in medical image processing as a whole.

2.4.1 Class Imbalance

Class imbalance is a common challenge in medical image segmentation, as some structures or regions of interest may be much smaller or less frequent than others. In medical images, pathological structures, such as tumors or lesions, are often much smaller and rarer than the surrounding healthy tissue. This imbalance can cause the learning process to be dominated by the majority class (healthy tissue), resulting in poor segmentation performance for the minority class (pathological structures).

Further, organs and anatomical structures in medical images can vary significantly in size and shape, both within and across patients. This variation can lead to class imbalance, as smaller structures may be underrepresented in the training data compared to larger ones. As a result, the segmentation algorithm may perform poorly on underrepresented structures. Several techniques can be employed to address the issue of class imbalance:

Choice of loss function: Initial segmentation networks primarily optimized the cross entropy loss function, which may prioritize majority classes, resulting in poor performance for the minority classes. Choosing more balanced loss functions like the Dice loss [109] of focal loss [88] can help alleviate the class imbalance problem.

Weighted Loss Functions: One approach to deal with class imbalance is to assign different weights to different classes in the loss function. This approach is especially useful when segmenting multiple classes in one image. For example, inversely weighting the classes based on their frequency in the training dataset increases the importance of under-represented classes during training.

Data Augmentation: Data augmentation techniques can artificially increase the number of samples from under-represented classes. This can involve applying various transformations, such as rotation, scaling, and flipping, to the original images, creating new, diverse examples for the network to learn from.

Sampling Strategies: Another approach to address class imbalance is to modify the sampling strategy during training. This can involve oversampling from under-represented classes or under-sampling from over-represented classes, ensuring that each class is equally represented in each training batch.

2.4.2 Domain Gap

Domain gap refers to the differences in data distribution between various datasets arising, for example, from heterogeneous imaging protocols, anatomical variations, or pathological manifestations. This gap can significantly impact the performance of segmentation algorithms, as they may not generalize well when applied to previously unseen data. The domain gap poses a major challenge in medical image segmentation, and addressing it is crucial for developing robust and reliable algorithms.

Medical images, such as CT or MRI scans, often come from various sources and can be acquired using different imaging protocols with diverse acquisition parameters and across different manufacturers. These variations can introduce inconsistencies in image appearance and contrast, which might affect the capability of a segmentation algorithm to generalize across different datasets or institutions. The disparity in image pre-processing techniques can further widen the domain gap.

The human body exhibits considerable variability in its anatomy, with individual differences in the size, shape, and position of organs and other structures. These variations can lead to differences in the appearance of anatomical structures in medical images, posing challenges for segmentation algorithms when applied to diverse patient populations. To overcome the domain gap in medical image segmentation, several strategies can be employed:

Domain adaptation: Domain adaptation techniques aim to adapt the model trained on a source domain (e.g., a specific dataset or imaging protocol) to perform well on a target domain (e.g., a different dataset or imaging protocol). This can involve unsupervised or supervised domain adaptation, using adversarial training, self-training, or style transfer techniques.

Domain generalization: Domain generalization methods focus on training a model that can generalize well across multiple unseen domains. This can be achieved through meta-learning, data augmentation, or multi-task learning, which encourage the model to learn domain-invariant features that are robust to variations in data distribution.

2.4.3 Limited Data

In medical image segmentation, the availability of annotated data is crucial for training and evaluating segmentation algorithms. However, obtaining a large amount of high-quality annotated data is a significant challenge due to several factors which can impact the development and performance of segmentation methods.

Manually annotating medical images requires a high level of expertise provided by medical professionals, such as radiologists. These specialists must carefully delineate the contours of anatomical structures or pathological regions in the images, which is labor-intensive and time-consuming. Given the workload, time restrictions medical professionals face, and associated costs, securing sufficient expert-annotated data can be difficult.

Furthermore, even among experts, the annotations may exhibit variability due to differences in interpretation or experience, leading to inconsistencies in the annotated data, commonly referred to as inter-rater variability. Such disparities may subsequently influence the training and evaluation of segmentation algorithms. To mitigate this issue, consensus annotations may be necessary, which could further amplify the time and effort required for data annotation.

Moreover, the privacy concerns of sharing medical images often containing sensitive patient information pose a formidable challenge. Strict regulations and guidelines regulate the sharing and usage of medical data. This can constrict the availability of annotated data necessary for developing and evaluating segmentation algorithms. Several approaches can be employed to address the challenge of limited annotated data.

Data augmentation generates new training samples from existing data. This way, the size of the training data can be effectively enhanced. Therefore it can help improve the algorithm's generalization capabilities without requiring additional expert annotations.

Transfer learning leverages pre-trained models or features learned from large, diverse datasets to enhance the performance of segmentation algorithms on smaller, target datasets. Transfer learning can help mitigate the effects of limited data availability by fine-tuning pre-trained models using the limited available annotated data.

Semi-supervised and self-supervised learning strategies are another effective way to harness labeled and unlabeled data in training segmentation algorithms. By optimizing the usage of the available data, the demand for extensive expert annotations can be lessened.

2.4.4 Ethics and Privacy Concerns

Applying deep learning algorithms in medical imaging comes with ethical and privacy concerns that must be considered when developing and deploying these technologies. This section briefly discusses some of the major concerns associated with using deep learning algorithms in medical imaging.

Data Privacy and Confidentiality: Medical imaging data contains sensitive patient information, necessitating strict measures to ensure data privacy and confidentiality. Safeguarding patient rights and complying with data protection regulations is crucial. Techniques like data anonymization, secure storage, and access control can help protect patient privacy. Throughout the work presented in this thesis, we have used publicly available datasets that follow these guidelines. Although I don't focus on this in this thesis, it is important to mention that there is a research field of privacy-preserving machine learning methods, such as federated learning, which can enable training on sensitive data without compromising individual patient records.

Bias and Fairness: Deep learning algorithms learn from the data they are trained on, making it vital to address potential biases in the training data. Biases can lead to unfair treatment or reduced performance for underrepresented patient groups. Developers must curate diverse and representative training data, considering factors like age, gender, ethnicity, and disease prevalence. Data augmentation, re-sampling, or domain adaptation can help balance the data and improve fairness across patient populations.

Transparency and Explainability: Deep learning models, known as "black boxes," can be challenging to interpret due to their complex structure and decision-making process. This lack of transparency and explainability is a concern in medical imaging, where trust and understanding are crucial. The development of explainable AI techniques is an emerging research field in the medical imaging community that often focuses on interpreting diagnostic decisions made by deep learning methods. However, it can also be relevant in the context of segmentation. Although these techniques are not directly explored in this thesis, they are increasingly important as they provide valuable insights into model decisions. They can enhance trust and support informed decision-making by clinicians and patients.

Part II

Voxel-Based Image Segmentation

3 Introduction

Medical image segmentation is a critical task in medical imaging and computer vision. It contributes to the downstream tasks of diagnosis and treatment planning in various medical disciplines. However, developing effective and reliable segmentation methods remains challenging due to limited available expert annotations and the need for robust models that can handle different imaging modalities and anatomical variations.

In this part, I present three novel deep learning techniques that address these challenges, focusing on Convolutional Neural Networks for volumetric medical images. The covered methods improve state-of-the-art medical image segmentation by addressing the challenges of limited available annotations, model recalibration, and unsupervised domain adaptation.

This part is built upon several publications, all of which are the result of collaborative efforts. For each publication, the specific contributions of the individual authors are detailed:

Rickmann, A., Roy, A. G., Sarasua, I., and Wachinger, C. (2020). Recalibrating 3d convnets with project & excite. IEEE Transactions on Medical Imaging, 39(7), (pp. 2461-2471) This work is covered in Chapter 5. A. Roy, C. Wachinger, and the author conceived the initial idea. The author did the implementation and execution of experiments. A. Roy and I. Sarasua provided feedback throughout the project. C. Wachinger and the author contributed equally to writing the final published article. I. Sarasua helped with proof-reading the final manuscript.

Gröger, F., Rickmann, A., and Wachinger, C. (2021). STRUDEL: Self-training with Uncertainty Dependent Label Refinement Across Domains. In Machine Learning in Medical Imaging: 12th International Workshop, MLMI 2021, Held in Conjunction with MICCAI 2021, Strasbourg, France, September 27, 2021, Proceedings 12 (pp. 306-316). Springer International Publishing. This work is covered in chapter Chapter 6. C. Wachinger and the author conceived the initial idea. F. Gröger implemented the proposed method, F. Gröger and the author ran the experiments. F. Gröger, C. Wachinger, and the author contributed equally to writing the final published article.

Rickmann, A., Xu, M., Wolf, T., Kovalenko, O., and Wachinger, C. (2023). HALOS: Hallucination-free Organ Segmentation after Organ Resection Surgery . In Information Processing for Medical Imaging, IPMI 2023, Springer International Publishing. This work is covered in chapter Chapter 7. C. Wachinger and the author conceived the initial idea. T. Wolf helped with the method development and integration of the DAFT module. M. Xu and the author implemented the proposed method and ran experiments. M. Xu, C. Wachinger, and the author contributed equally to writing the final published article. O. Kovalenko helped with her medical expertise in interpreting some results and provided manual segmentations.

The outline of this part is as follows: First, in Chapter 4, I will explain the fundamental knowledge needed for understanding this part, including current challenges in medical image segmentation, such as limited data and domain differences. In the following chapters, I will present the previously mentioned methods.

One of the primary challenges in medical image segmentation is the need for accurate and robust models that can generalize well across different datasets and imaging protocols, particularly when dealing with limited available annotations. In Chapter 5, we propose a method for 3D model recalibration, one of the first of such techniques to be designed specifically for 3D networks. We show that the approach also leads to better generalization across datasets, including brain and abdominal organ segmentation tasks.

Self-training holds potential as an effective technique for unsupervised domain adaptation by utilizing self-created pseudo-labels. Nonetheless, these pseudo-labels might be of varying quality and may negatively impact the model's performance. In Chapter 6, a new unsupervised domain adaptation method specifically for segmenting white matter hyperintensity is introduced. Self-training with Uncertainty Guided Label refinement (STRUDEL) estimates the uncertainty associated with pseudo-labels and incorporates this information into the training phase via an uncertainty-aware loss function, giving more weight to highly certain labels. Moreover, the methodology is further refined by including the segmentation results from an established method known for its reliability in white matter hyperintensity segmentation.

In Chapter 7, Hallucination-free Organ Segmentation (HALOS) is presented as a technique for segmenting abdominal organs in MR images, with particular efficacy in managing cases post-organ resection surgery. The methodology merges the tasks of identifying missing organs and segmenting multiple organs into a single multi-task model. This integration lessens the instances of organ hallucinations and enhances the overall performance of the segmentation. This work highlights the importance of sturdy deep learning models for efficiently dealing with atypical anatomy in medical image segmentation.

Finally, in Chapter 8, I will discuss the approaches and their limitations and provide an outlook on future developments in the research field of medical image segmentation.

4 Fundamentals for Voxel-Based Image Segmentation

Contents

Voxel-based image segmentation is used in medical imaging and computer vision to partition a 3D volumetric image into multiple regions or segments, each corresponding to a specific anatomical structure or object of interest. In this context, a voxel, short for volume element, represents the smallest unit of the 3D image, analogous to a pixel in 2D images. The goal of voxel-based image segmentation is to assign a label to each voxel in the 3D image, indicating the structure or object to which it belongs. Several methods exist for voxel-based image segmentation, ranging from manual delineation by experts to semi-automatic and fully automatic techniques. In recent years, there has been a surge of interest in deep learning-based methods, particularly F-CNNs, owing to their capacity to yield exact segmentation outcomes. In this chapter, I will explain the fundamentals of voxel-based medical image segmentation, focusing on F-CNNs and current challenges in the field.

4.1 Convolutional Neural Networks for Medical Image Segmentation

Convolutional Neural Networks (CNNs) are a specific category of deep learning models designed for processing grid-like data, such as images. The principle of convolution, which is the key operation in these networks, is not a new concept. Historically, convolution has been a critical operation in signal and image processing, using specially designed convolutional filters or kernels for tasks like blurring, edge detection, and smoothing. In the context of image processing, a convolutional kernel is a small matrix of weights that is passed over the input image. The kernel performs element-wise multiplication with the image sub-region aligned with it, and all these products are then summed to produce a single value in the output feature map. This process is repeated across the entire image, transforming the original image

data in a way that can emphasize certain features or patterns. The innovative idea behind CNNs is that rather than manually designing sequences of convolutional filters for distinct tasks, the filter weights could be made learnable. With the advent of Alexnet, CNNs have gained widespread recognition in computer vision tasks, from image classification to object detection. F-CNNs, first introduced by Long et al. [93], are deep learning models specifically designed for image segmentation tasks. F-CNNs can process input images of arbitrary size and produce segmentation masks of the same size as the input image. F-CNNs typically consist of an encoder, which extracts feature maps from the input image at various resolutions, and a decoder, which upsample these feature maps to produce the final segmentation mask.

In the medical imaging domain, F-CNNs have demonstrated remarkable success in various segmentation tasks, substantially improving the performance and efficiency compared to traditional image processing techniques [17, 85, 89, 109, 139]. A typical F-CNN comprises a series of layers, each serving a specific purpose. The primary building blocks of an F-CNN for image segmentation include the following:

Convolutional layers

These layers apply convolution operations on the input data using a set of learnable filters, which help detect local patterns or features in the input image. Convolutional layers can capture spatial relationships within the input data, making them suitable for image processing tasks.

Activation layers

The introduction of non-linearities into the network is facilitated by activation functions like ReLU (Rectified Linear Units), sigmoid, or tanh to the outputs generated by the convolutional layers. These activation functions empower the network to capture intricate non-linear associations between the inputs and outputs.

Downsampling layers

Downsampling layers are used to reduce the spatial dimensions of the feature maps, which can help control the computational complexity of the network and mitigate overfitting. Standard downsampling methods are pooling operations, including max pooling and average pooling, or strided convolution, which introduces additional network parameters.

Upsampling layers

Upsampling layers play a crucial role in the decoder part of the network, as they help increase the spatial resolution of the feature maps. There are several methods for upsampling in F-CNNs, including transposed convolution and unpooling. *Transposed convolution* (also known as deconvolution or fractionally-strided convolution) is an upsampling method that involves learning a set of filters to perform the upsampling operation. This method consists in convolving the lower-resolution feature map with a learnable set of filters to produce a higher-resolution output. *Unpooling* is an upsampling method that reverses the pooling operation used in the encoder part of the network. For example, during a max-pooling operation, the network retains the location of the maximum value in each pooling region. In the max-unpooling process, these maximum values are placed back into their original positions in the higher-resolution

feature map, while the other positions are filled with zeros. Unpooling can help recover some of the spatial information lost during the pooling operation and comes with the benefit that it adds no additional complexity in terms of learnable parameters to the network.

4.1.1 U-Net and Variants

U-Net is a widely-used F-CNN architecture for image segmentation, particularly in biomedical image analysis. It was first introduced by Ronneberger et al. [139] in 2015 and has since inspired various adaptations and modifications to suit different applications and data types. In this section, I will describe the original U-Net architecture and three of its prominent variants: 3D U-Net [17], V-Net [109], and nnU-Net [68, 69]. An overview of the original U-Net architecture is depicted in Figure 4.1.

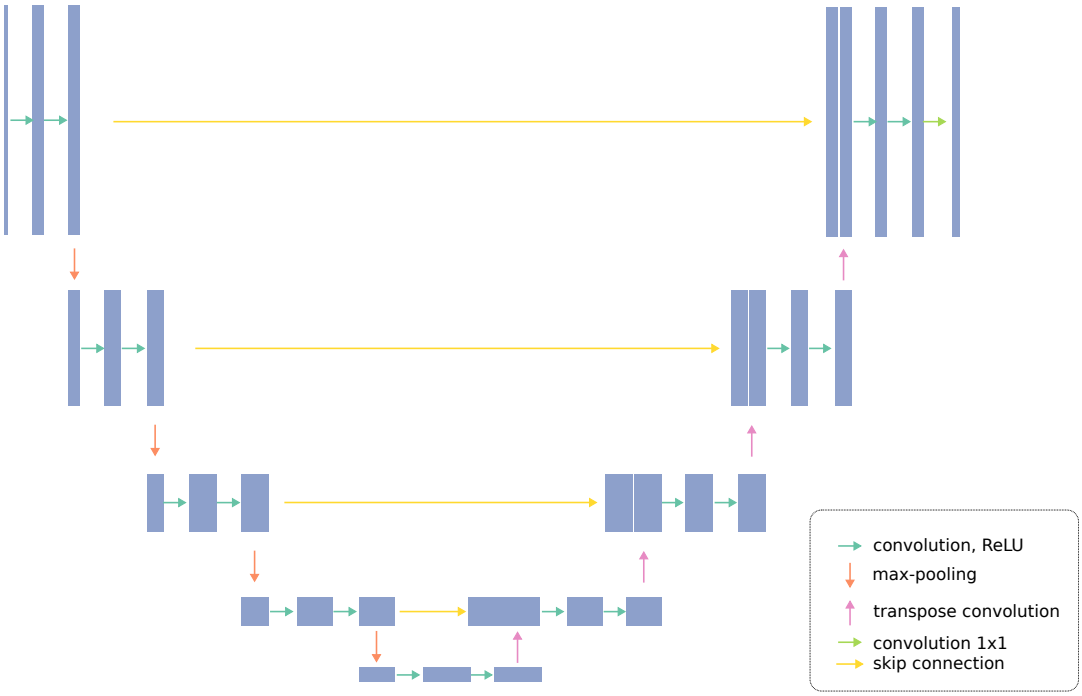

Figure 4.1. Architecture of U-Net, figure adapted from Figure 1 in [139].

U-Net [139] was one of the first F-CNN architectures designed for biomedical image segmentation tasks. It has an encoder-decoder structure with skip connections, forming a U-shaped network. The encoder progressively captures high-level semantic information by downsampling the input image using max-pooling. The decoder recovers spatial information and upsamples the feature maps, using transpose convolution, to generate the final segmentation mask. One of the key components of this architecture is that it connects the encoder and decoder layers by so-called skip connections. These connections pass the feature map from the corresponding encoder layer to the decoder and help retain fine-grained details, leading to precise segmentation results and further providing an additional path for gradients to flow during backpropagation.

3D U-Net [17] extends the original U-Net architecture to handle 3D volumetric data, making it suitable for voxel-based image segmentation tasks. It replaces the 2D convolutional, pooling,

and upsampling layers with their 3D counterparts, enabling the network to capture spatial information across the height, width, and depth dimensions. This modification allows 3D U-Net to effectively segment 3D structures in medical imaging applications, such as brain tumor and organ segmentation. This architecture also adds batch normalization [67] into the architecture.

V-Net [109] is another 3D adaptation of the U-Net architecture proposed concurrently with 3D U-Net. Like 3D U-Net, it proposes using volumetric convolutions, but it also introduces an innovative loss function based on the Dice coefficient. This Dice loss function has since become one of the most frequently employed loss functions in medical image segmentation. The unique characteristic of the Dice loss function is that it promotes a greater overlap between the predicted and the true segmentation masks. This leads to a more robust performance in scenarios where class imbalance exists or the target structures are small. Another noteworthy feature of the V-Net architecture is its incorporation of residual connections at each convolutional stage. In this context, a convolutional stage refers to a sequence of convolutional layers operating at the same spatial resolution. The residual connections take the input feature map at the start of the convolutional stage and add it to the output feature map at the end of the stage. V-Net also differs from 3D U-Net in a few other respects. For instance, it utilizes strided convolution for downsampling instead of the pooling layers used in 3D U-Net. Furthermore, it omits batch normalization layers, and instead of the Rectified Linear Unit (ReLU) activation function, it employs the Parametric ReLU (PReLU) [58].

nnU-Net [68, 69], is a highly adaptable and robust segmentation framework based on the U-Net architecture. It aims to automate the network design and configuration process for various medical image segmentation tasks. nnU-Net includes preprocessing steps, such as intensity normalization and data augmentation, and incorporates 2D and 3D U-Net architectures. The framework dynamically selects the most suitable architecture, loss function, and training strategy for the given task, making it highly versatile and applicable to various segmentation challenges. The nnU-Net framework has become a commonly used baseline for comparing medical image segmentation methods.

U-Net and its variants have proven highly effective for various image segmentation tasks, particularly in medical imaging. These architectures provide a strong foundation for developing advanced segmentation techniques that can address the unique challenges and requirements of voxel-based image segmentation.

4.1.2 2D vs. 3D F-CNNs

In voxel-based image segmentation, F-CNNs can be designed as either 2D or 3D architectures, depending on the nature of the input data and the desired output. The main difference between 2D and 3D F-CNNs lies in the spatial dimensions they process. While 2D F-CNNs handle 2D images and perform convolutions across the height and width dimensions, 3D F-CNNs work with volumetric data and perform convolutions across the height, width, and depth dimensions. This difference results in distinct kernel shapes, with 2D F-CNNs using 2D kernels (e.g., 3x3) and 3D F-CNNs using 3D kernels (e.g., 3x3x3).

Therefore, 2D F-CNNs, such as U-Net [139], have fewer parameters and require fewer computational resources than their 3D counterparts. This makes them faster to train and more suitable for systems with limited hardware capabilities. They are a fitting choice for 2D medical images like dermatology or histology photos, 2D ultrasound, or x-ray scans. Since 2D F-CNNs do not inherently model the depth dimension, they may fail to capture the full spatial context and inter-slice correlations in 3D volumetric data, like Magnetic Resonance Imaging (MRI) or Computed Tomography (CT) scans. This can lead to suboptimal performance in voxel-based image segmentation tasks, especially for structures with complex 3D shapes. A popular strategy to mitigate this problem is to train multiple 2D networks on different views of the 3D data, like slicing the 3D volume in the axial, coronal, or sagittal direction. QuickNAT [143] is a popular architecture aggregating predictions from the three standard views. Another method to overcome this problem is concatenating multiple consecutive slices of a volumetric image in the channel dimension and utilizing a 2D network, as in [59]. This approach is sometimes referred to as 2.5D.

3D F-CNNs can naturally capture the 3D spatial context and inter-slice correlations in volumetric data, making them more suitable for voxel-based image segmentation tasks. This ability to leverage 3D information can lead to better segmentation performance than 2D F-CNNs, particularly for structures with intricate 3D geometries. However, 3D F-CNNs typically have more parameters and require more computational resources than 2D F-CNNs. This can make them more challenging to train and deploy, particularly on systems with limited hardware capabilities. Additionally, the larger memory footprint of 3D F-CNNs can necessitate smaller batch sizes during training, which may affect the convergence properties and stability of the learning process.

4.1.3 Loss Functions

To train a neural network, it is essential to evaluate the congruence between the predicted segmentation output by the network and the actual labels. This necessitates the utilization of a differentiable loss function, which facilitates the computation of gradients needed for the backpropagation process. The optimization process seeks to minimize these loss functions. In image segmentation, several loss functions are commonly adopted, including:

Cross-Entropy Loss

The cross-entropy loss is widely used for segmentation tasks. It measures the pixel-wise discrepancy between the predicted probability distribution and the true distribution of class labels. It encourages the network to produce accurate class probabilities, improving segmentation results.

Dice Loss

The Dice loss, as introduced by Milletari et al. [109], is derived from the Dice coefficient, which measures the overlap between the predicted segmentation and the ground truth. Dice loss is defined as one minus the Dice coefficient, with lower values indicating better overlap. This loss function is beneficial for imbalanced datasets, as it inherently balances the contribution of different classes.

Combined Loss Functions

In practice, combining different loss functions to leverage their respective strengths is common. For example, combining cross-entropy and Dice loss can provide pixel-wise classification accuracy and region overlap consistency, leading to better segmentation results.

4.1.4 Uncertainty

Uncertainty in CNNs is a concept that can provide insights into the confidence level of model predictions. Understanding this uncertainty becomes crucial in medical imaging tasks, where predictions have significant consequences. A common technique approximating this uncertainty is Monte Carlo Dropout (MC Dropout) [43]. Dropout [157] is a regularization technique used during training, where a fraction of neurons in a layer are randomly "dropped out" or temporarily deactivated, helping to prevent overfitting by ensuring the model does not rely too heavily on any single neuron.

In the context of MC Dropout, this idea is extended into the inference phase. Instead of a single forward pass with all neurons active, as typically done in inference, multiple forward passes are conducted with random dropout still in effect. Each forward pass thus results in slightly different output due to the randomness introduced by dropout, forming a distribution of predictions. The variance in this distribution is then used to measure model uncertainty. By providing an estimate of prediction uncertainty, MC Dropout can be a valuable tool in highlighting difficult or outlier cases and could enhance decision-making processes in clinical settings.

4.2 Evaluation of Segmentation Methods

Evaluation metrics are essential for assessing the performance of voxel-based image segmentation methods. They provide quantitative measures of the agreement between the predicted segmentation and the ground truth labels, helping researchers and practitioners compare different approaches and identify areas for improvement. It is important to choose evaluation metrics depending on the task and to evaluate multiple metrics, e.g., boundary-based metrics measure different properties than overlap metrics. A new framework for selecting metrics for medical image analysis suggests choosing an overlap metric, a boundary metric, and other application-specific metrics if needed [99].

4.2.1 Overlap Metrics

Overlap metrics count and compare the overlapping pixels of predicted segmentation and ground truth. They are very commonly used metrics for medical image segmentation but have an essential drawback: they are biased towards single large organs and do not take organ shape into consideration [99]. One of the most commonly used overlap metrics is the *Dice Coefficient*, also known as the Dice Similarity Index. The Dice Coefficient is the ratio of twice the intersection volume to the sum of the volumes of both sets. Dice coefficients range from 0 to 1, with higher values indicating better performance. Another commonly used overlap

metric is the *Intersection over Union (IoU)*, also called the Jaccard Index. The IoU is defined as the ratio of the intersection volume to the union volume of both sets. Like the Dice coefficient, IoU values range from 0 to 1, with higher values indicating better performance.

4.2.2 Boundary Metrics

The *Hausdorff Distance (HD)* measures the maximum distance between the boundaries of the predicted segmentation and the ground truth. It is a worst-case metric that evaluates the largest discrepancy between the two sets, indicating the overall quality of the segmentation.

Contrastingly, the *Average Symmetric Surface Distance* measures the average distance between the boundaries of the predicted segmentation and the ground truth. Unlike the Hausdorff distance, it evaluates the overall agreement between the boundaries, providing a more comprehensive assessment of segmentation performance.

The *Surface Dice* [116], or normalized surface distance, is another useful metric in evaluating image segmentation quality. Rather than just measuring the overlap of pixels, this metric accounts for the physical distance between the surfaces of the predicted and ground truth segmentations. It computes the Dice coefficient on the surface voxels of the predicted and ground truth segmentations, where a surface voxel is defined as a voxel at the boundary of the segmentation region. In this way, the Surface Dice provides an additional measure of spatial accuracy that is more sensitive to the exactness of segmentation boundaries. As with other Dice metrics, a Surface Dice score ranges from 0 to 1, with higher values indicating better performance.

4.2.3 Application-Specific Metrics

In some cases, typically used overlap and boundary metrics are not enough to measure the performance of a proposed segmentation algorithm. For example, depending on the application, looking at the algorithm's runtime or GPU memory demands might be necessary. In some tasks, like lesion segmentation, it might be essential to compute metrics such as lesion Recall or F1 score, as explained in Chapter 6. In Chapter 7, we focus on segmentation after organ resection surgery, where the most crucial application-specific metric is the false positive rate (FPR). In summary, multiple evaluation metrics can and should be used to assess the performance of voxel-based image segmentation methods. It is important to consider various metrics to obtain a comprehensive understanding of the model's performance and identify areas for improvement.

Recalibration of 3D ConvNets

5

Contents

5.1 Introduction

As introduced in Chapter 4, F-CNNs have gained popularity for semantic image segmentation in natural [5, 14, 93] and medical images [140, 143]. However, most tasks in computer vision involve 2D natural images; therefore, architectural advancements are primarily focused on 2D F-CNNs. In contrast, medical imaging primarily deals with 3D scans, such as CT and MRI, which cannot be processed directly by 2D F-CNNs. When our work on recalibration of 3D ConvNets [135] was published, using 2D F-CNN to analyze 3D medical scans slice by slice was common practice. However, this approach ignores contextual information from neighboring slices [108], resulting in suboptimal results. 3D F-CNNs have gained attention in medical image segmentation and have shown promising results [13, 18, 29, 109, 167]. Nevertheless, there are still practical challenges, such as the significantly higher number of

learnable parameters than their 2D counterparts, leading to overfitting when data are scarce, and the considerable GPU memory required for training. These challenges are particularly prevalent in medical image segmentation. Medical datasets are typically much smaller than computer vision datasets, with only 15-20 labeled scans being typical at the time of publication of this article [82, 161]. Although dataset sizes have increased since then, they only consist of hundreds of scans, substantially less than datasets such as ImageNet [26] containing over a million images. Additionally, 3D medical scans demand significantly more memory. Researchers often meticulously engineer 3D F-CNNs for specific tasks by reducing model complexity to address this issue. However, this approach restricts the model's capacity to learn relevant features.

The second challenge may be overcome by dividing the complete volume into smaller subvolumes and training the model. This will reduce the memory demand of the training process. However, this method has limitations, as it reduces the model's contextual awareness, similar to 2D F-CNNs, and requires strategies for assembling the complete volume [64]. Recently, this approach has demonstrated great potential when combined with transformer architectures, as in the case of UNETR [56]. It is worth noting that this chapter solely addresses F-CNNs as the article was published before the transformer architecture era.

Hu et al. [62] introduced the Squeeze and Excite (SE) module, that recalibrates intermediate feature maps, resulting in improved performance in classification tasks within the computer vision domain, with only a marginal increase in model complexity. SE modules enable the modeling of interdependencies between feature map channels and learn to emphasize specific channels based on the task at hand. Incorporating SE modules into 2D Convolutional Neural Network (CNN) architectures has been shown to improve performance with only a small increase in the number of learnable parameters [62, 144], which makes it a promising method for 3D F-CNNs.

This chapter delves into the recalibration of feature maps in 3D F-CNNs by exploring different recalibration blocks originally designed for 2D networks and adapting them to the 3D domain. Additionally, we introduce a novel recalibration module called the Project & Excite (PE) module, specifically tailored for 3D network architectures. We hypothesize that completely discarding spatial information from high-dimensional feature maps, as done in the SE module through global pooling, may result in the loss of crucial information, especially for segmentation tasks requiring precise localization of anatomical structures. Instead, we aim to retain spatial information while simultaneously providing the network with a global receptive field at each stage. To achieve this, we employ traditional tensor slicing techniques [128] to compute average values along the three primary axes of the tensor, generating three projection vectors that indicate the relevance of the slices along each axis. The process is visually illustrated in Figure 5.1. Consequently, as observed in the SE module, the network learns the interdependencies among these projection vectors across the channels for excitation rather than learning dependencies between scalar values across channels.

The research presented in this chapter [135] builds upon our previous work [134] and presents a comprehensive framework for recalibration methods. We thoroughly validated the proposed module across diverse datasets and conducted experiments on two medical image segmentation tasks, leading to the following noteworthy contributions. Firstly, we

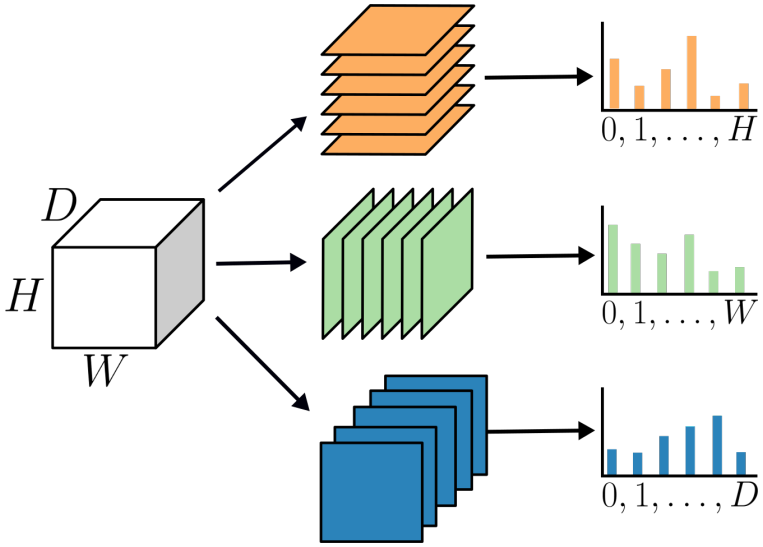

Figure 5.1. Illustration of 3D tensor slicing along the three axes and the subsequent generation of 1D projections through methods such as average pooling, as employed in Project & Excite blocks. © IEEE,2020

introduce a novel computational block, Project & Excite (PE), which enables the recalibration of 3D F-CNNs. Secondly, we demonstrate the seamless integration of PE blocks into 3D F-CNNs by incorporating them into two distinct architectures. Thirdly, we establish that integrating PE blocks enhances segmentation accuracy, particularly for smaller target classes, while adding only a minimal amount of additional model parameters compared to including additional convolutional layers. Fourthly, we introduce the compress-process-recalibrate pipeline, facilitating the direct comparison of different recalibration blocks. Finally, we extend existing recalibration techniques to the 3D setting and conduct comparative evaluations against our proposed PE blocks. These experiments substantiate our hypothesis that preserving spatial information is crucial for achieving optimal performance in 3D settings.

5.1.1 Related Work

In the previous chapter, we discussed the fundamental differences between 2D and 3D networks. In 3D networks, the number of channels is typically lower compared to 2D networks. To address the high memory requirements of 3D networks, a commonly employed approach is training 3D F-CNNs on subvolumes [13, 29, 72, 167]. When using subvolumes for training, the sampling strategy must be tailored to the specific task. As there is no need to store activations for backpropagation during inference time, some networks, usually less complex ones, are able to process the entire volume at once at test time. However, other networks process the volume in segments during inference. In cases where these segments overlap, merging them becomes necessary to generate a complete volume segmentation. This merging process often requires a label fusion strategy to handle overlapping sections effectively. One example of a network using subvolumes is SLANT [64], where brain volumes are divided into overlapping subspaces, and each subspace is registered to a standardized atlas. Subsequently, individual 3D F-CNNs are trained for each subspace. In contrast to the conventional training strategy on subvolumes, our objective is to train 3D F-CNNs on complete volumes without additional preprocessing, postprocessing, or complex stitching methods. This requires designing a low-complexity model

that can still achieve accurate segmentations. In pursuit of this objective, we draw inspiration from the successful use of Squeeze and Excite modules in 2D networks.

The Squeeze and Excite (SE) module [62] comprises two main operations: squeezing and exciting. During the squeezing operation, the spatial dimensions of the feature maps are reduced by global pooling to generate a channel descriptor. This descriptor captures the significance of each channel. In the exciting operation, a set of learnable parameters is used to produce attention weights for each channel. These attention weights are then applied to the original feature maps, enabling the network to selectively enhance informative channels while suppressing less relevant ones. The network can effectively capture long-range dependencies across different image regions, even in deeper layers, by incorporating SE modules. As a result, classification performance is improved without significantly increasing the model's complexity.

Several researchers have expanded the application of Squeeze and Excitation (SE) modules to various computer vision and medical image analysis tasks, adapting them for classification and segmentation purposes. Roy et al. [144] introduced the spatial squeeze and excite (sSE) block as an extension of SE specifically designed for medical image segmentation. Recognizing the importance of preserving fine-grained spatial information, the sSE block compresses channel information and performs recalibration spatially. Their findings indicate that the sSE block surpasses the original channel SE module (cSE) [62] in medical segmentation tasks, and a combination of both modules (scSE) achieves even higher performance. Importantly, their work highlights the potential advantages of lightweight blocks over additional convolutional layers.

Similar to [144], the convolutional block attention module (CBAM) [176] combines channel and spatial attention modules sequentially rather than simultaneously. These modules use max and average pooling to compress channel and spatial information. It has been shown that using max-pooling in addition to average pooling enhances performance over using a single pooling approach. While sSE, scSE, and CBAM have shown promise in 2D segmentation tasks, their 3D extensions have not been thoroughly examined.

Pereira et al. [124] developed the jointly learned channel and spatial recalibration module (SegSE) for medical segmentation problems, which employs dilated convolutions rather than average pooling. While SegSE outperforms pooling-based recalibration approaches, it necessitates more GPU RAM due to the usage of dilated convolutions.

Although originally developed for 2D architectures, cSE blocks have recently been extended to 3D F-CNNs for volumetric segmentation assistance [187]. Zhu et al. directly adapted the cSE module for 3D channel recalibration in medical image segmentation. Their results demonstrate improved performance compared to baseline models without recalibration blocks, although spatial recalibration was not performed. Recently, Zhu et al. [187] extended cSE blocks, first created for 2D structures, to 3D F-CNNs for volumetric segmentation. Despite the absence of spatial recalibration, their results demonstrate improved performance compared to baseline models without recalibration blocks.

Table 5.1. Various squeeze and excite module versions are compared to our proposed Project & Excite (PE) module in terms of the operations of compress $\mathcal{C}(\cdot)$, process $\mathcal{P}(\cdot)$, and recalibrate $\mathcal{R}(\cdot,\cdot)$. The second column lists the CNN type (2D or 3D) for which the module was designed.

		$\mathbf{C}(\cdot)$		$\mathbf{P}(\cdot)$		$\mathbf{R}(\cdot,\cdot)$	
Module	Used in	Linear	Parametric	FC	Conv	Gating function	Recalibration
cSE [62, 187]	2D & 3D CNNs	✓	✗	✓	✗	sigmoid	channel-wise multiplication
sSE [144]	2D CNNs	✓	✓	✗	✓	sigmoid	element-wise multiplication
CBAM channel [176]	2D CNNs	✗	✗	✓	✗	sigmoid	channel-wise multiplication
CBAM spatial [176]	2D CNNs	✗	✗	✗	✓	sigmoid	element-wise multiplication
Project & Excite	3D CNNs	✓	✗	✗	✓	sigmoid	element-wise multiplication

5.2 Methods

We have observed that the previously discussed recalibration modules share a common recalibration process. To comprehensively compare these methods, we propose a universal framework called compress-process-recalibrate (CPR). Within this framework, all recalibration blocks operate on a high-dimensional feature map, typically obtained from a preceding convolutional layer in the network. The first step, performed by the function $\mathcal{C}(\cdot)$, involves compressing the input feature map $\mathbf{U}$ into a lower-dimensional embedding $\mathbf{Z}$. In the case of SE, this compression is achieved through global average pooling, resulting in a vector of scalar values per channel stored in $\mathbf{Z}$.

Subsequently, the processor $\mathcal{P}(\cdot)$ learns a mapping from the low-dimensional embedding $\mathbf{Z}$ to recalibration factors $\hat{\mathbf{Z}}$. For SE, this mapping is accomplished using a fully connected layer.

The final step of recalibration, denoted as $\mathcal{R}(\cdot,\cdot)$, involves rescaling $\hat{\mathbf{Z}}$ using a gating function, followed by element-wise multiplication of the input feature map with $\hat{\mathbf{Z}}$. In the case of SE, the gating function is implemented as a sigmoid layer.

This recalibration process generates a recalibrated output feature map $\hat{\mathbf{U}}$, which emphasizes or suppresses specific channels or spatial locations. A visual representation of the CPR framework can be found in Figure 5.2. Various techniques can be employed for feature map compression, such as linear or non-linear pooling operations, as well as parametric or non-parametric approaches like convolutions. The processor component of CPR is typically parametric and may incorporate fully connected or convolutional subnetworks. For a comprehensive overview of different recalibration blocks within the CPR framework, please refer to Table 5.1. In the subsequent sections, we will delve into the details of existing recalibration blocks, extend them to 3D, and introduce the Project & Excite block within this CPR framework.

5.2.1 3D Channel Squeeze & Excite

The 3D Channel Squeeze & Excite (cSE) module is an adaptation of the original 2D SE block [62] for 3D networks [187].

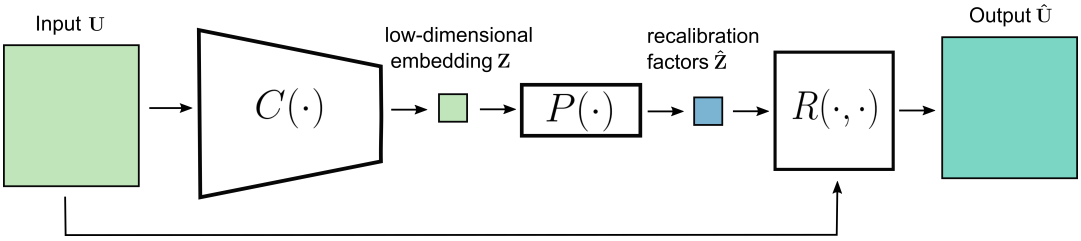

Figure 5.2. Compress-Process-Recalibrate (CPR) framework illustration. Using the Compressor function $\mathcal{C}(\cdot)$, the input feature map $\mathbf{U}$ is compressed, yielding a lower-dimensional embedding $\mathbf{Z}$. The Recalibration function $\mathcal{R}(\cdot,\cdot)$ scales the input feature map using the recalibration factors $\hat{\mathbf{Z}}$ that were learned by the Processor function $\mathcal{P}(\cdot,\cdot)$. The updated feature map is represented by the output $\hat{\mathbf{U}}$. © IEEE,2020

The compression function $\mathcal{C} : \mathbb{R}^{H\times W\times D\times C} \rightarrow \mathbb{R}^{C}$ applies a global average pooling operation to the input feature map $\mathbf{U}$, reducing its spatial dimensions and generating a scalar value per channel $\mathbf{z} \in \mathbb{R}^{C}$, referred to as the "squeeze". For simplicity, let us denote a single channel of the input $\mathbf{U}$ as $\mathbf{U}_c$. The processing function $\mathcal{P}(\cdot)$ takes the squeezed representation $\mathbf{z}$ as input and uses two fully connected layers, represented by $\mathbf{W}_1 \in \mathbb{R}^{\frac{C}{r}\times C}$ and $\mathbf{W}_2 \in \mathbb{R}^{C\times\frac{C}{r}}$, to learn inter-channel dependencies. r is a hyperparameter that controls the channel reduction factor, allowing for trade-offs between computational and memory costs. The ReLU nonlinearity δ is applied to the intermediate result, resulting in the recalibration factors $\hat{\mathbf{z}}$. The recalibration function $\mathcal{R}(\cdot,\cdot)$ applies a sigmoid gating function σ to the recalibration factors $\hat{\mathbf{z}}$, allowing for the emphasis or suppression of multiple channels. Finally, the input feature map is scaled channel-wise with the learned recalibration weights.

The operations of the 3D cSE module can be defined as follows:

$$\mathcal{C}: \quad \mathbf{z} = \text{AvgPool}(\mathbf{U}), \tag{5.1}$$

$$\mathcal{P}: \quad \hat{\mathbf{z}} = \mathbf{W}_2\delta(\mathbf{W}_1\mathbf{z}), \tag{5.2}$$

$$\mathcal{R}: \quad \hat{\mathbf{U}}_c = \sigma(\hat{z}_c)\mathbf{U}_c, \tag{5.3}$$

here, AvgPool denotes the channel-wise average pooling operation.

5.2.2 3D Spatial Squeeze & Excite

The spatial SE block (sSE) [144] was created particularly for segmentation tasks and may be expanded to 3D networks. We replace all functions in this extension with their 3D equivalents. Unlike other recalibration modules, sSE notably integrates the *process* step inside the *compress* transformation. The definitions of the operations $\mathcal{C}, \mathcal{P}$, and $\mathcal{R}$ are as follows:

$$\mathcal{C}, \mathcal{P}: \quad \mathbf{Z} = \mathbf{S} \star \mathbf{U}, \tag{5.4}$$

$$\mathcal{R}: \quad \hat{\mathbf{u}}_c = \sigma(\mathbf{Z}) \cdot \mathbf{U}_c. \tag{5.5}$$

The weights of the convolution kernel are represented as $\mathbf{S} \in \mathbb{R}^{1\times1\times1\times C\times1}$, where C is the number of channels. The compression procedure reduces the channel dimension to 1 by applying a $1 \times 1 \times 1$ kernel on the channel information. In the recalibration procedure, the resultant recalibration map is rescaled using a sigmoid layer and element-wise multiplied with each channel of the input feature map. This preserves geographical information while highlighting or suppressing certain channels.

5.2.3 3D Spatial and Channel Squeeze & Excite

The combination of cSE and sSE blocks has been proposed in [144] as spatial and channel SE (scSE). The input feature map $\mathbf{U}$ is processed separately in the scSE block by a cSE and a sSE block. This produces two output feature maps, $\hat{\mathbf{U}}_{cSE}$ and $\hat{\mathbf{U}}_{sSE}$. These two feature maps are combined using an element-wise max operation to get the final result $\hat{\mathbf{U}}_{scSE}$. To expand scSE to 3D networks, we use the previously reported 3D cSE and 3D sSE blocks, which allow for recalibration in both the channel and spatial dimensions.

5.2.4 3D Convolutional Block Attention Module

The Convolutional Block Attention Module (CBAM) [176] was designed for 2D classification and object detection tasks and has not been widely utilized for 3D segmentation. We study its application to 3D networks and compare our PE blocks to a 3D version of CBAM owing to its similarities to squeeze and excite blocks.

Like the scSE block, CBAM consists of two components: a channel attention block and a spatial attention block. But in contrast to scSE, CBAM applies the channel and spatial blocks sequentially. The channel attention block is related to the 3D cSE block. On the input feature map $\mathbf{U}$, it conducts global max pooling and average pooling and feeds the pooled representations into a shared, fully connected subnetwork. The obtained features are added element by element and sent through a sigmoid gating function. The acquired weights are then used to scale the input feature map by element-wise multiplication. The following are the definitions of the functions $\mathcal{C}_{avg}$, $\mathcal{C}_{max}$, $\mathcal{P}$, and $\mathcal{R}$:

$$\mathcal{C}_{avg}: \quad \mathbf{z}_{avg} = \text{AvgPool}(\mathbf{U}), \tag{5.6}$$

$$\mathcal{C}_{max}: \quad \mathbf{z}_{max} = \text{MaxPool}(\mathbf{U}), \tag{5.7}$$

$$\mathcal{P}: \quad \hat{\mathbf{z}} = \mathbf{W}_2\delta(\mathbf{W}_1\mathbf{z}_{avg}) + \mathbf{W}_2\delta(\mathbf{W}_1\mathbf{z}_{max}), \tag{5.8}$$

$$\mathcal{R}: \quad \hat{\mathbf{U}}_c = \sigma(\hat{z}_c)\mathbf{U}_c, \tag{5.9}$$

where AvgPool and MaxPool represent the channel-wise pooling operations, and $\mathbf{W}_1 \in \mathbb{R}^{\frac{C}{r}\times C}$ and $\mathbf{W}_2 \in \mathbb{R}^{C\times\frac{C}{r}}$ denote the weights of the fully connected layers.

The spatial attention block focuses on condensing the channel information. It conducts average pooling and maximum pooling operations along the channel dimension and concatenates the generated features. The concatenated features are sent through a $1 \times 1 \times 1$ convolutional layer

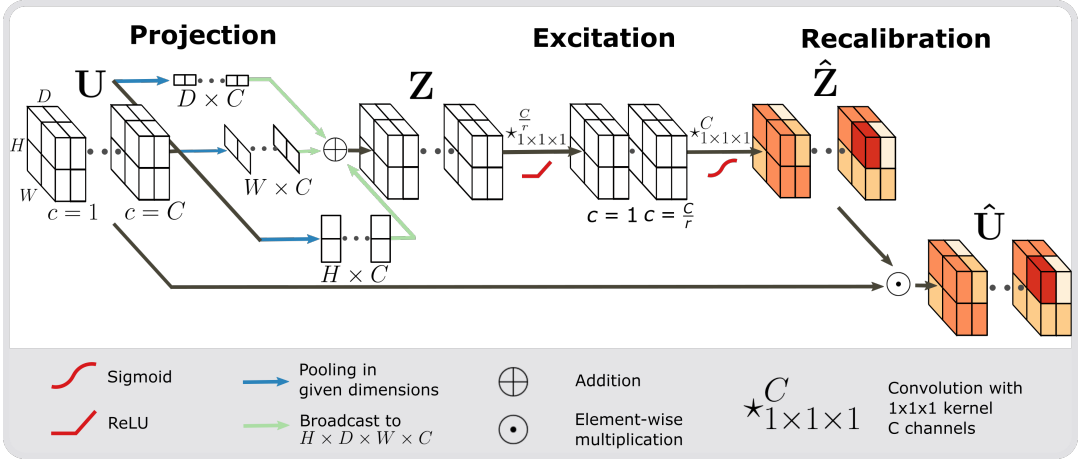

Figure 5.3. Illustration of the proposed Project & Excite block. The 4D input feature map $\mathbf{U}$ is first projected, and three projection vectors per channel are generated using pooling techniques such as average pooling. The three vectors are then expanded to the original input dimension and element by element added to produce the intermediate feature map $\mathbf{Z}$. The feature map is then passed through two convolutional layers in the excitation operation, which first reduce and then increase the channel size. The recalibration factor r is a hyper-parameter that can be used to fine-tune this decrease. Finally, the original input feature map is multiplied by the recalibration map $\hat{\mathbf{Z}}$ to generate the recalibrated feature map $\hat{\mathbf{U}}$ in the recalibration phase. ©IEEE,2020

to construct the spatial attention map, followed by a sigmoid layer. The following are the definitions of the functions $\mathcal{C}_{avg}$, $\mathcal{C}_{max}$, $\mathcal{C}$, $\mathcal{P}$, and $\mathcal{R}$:

$$\mathcal{C}_{avg}: \quad \mathbf{Z}_{avg} = \text{AvgCPool}(\mathbf{U}) \tag{5.10}$$

$$\mathcal{C}_{max}: \quad \mathbf{Z}_{max} = \text{MaxCPool}(\mathbf{U}) \tag{5.11}$$

$$\mathbf{Z} = [\mathbf{Z}_{avg}; \mathbf{Z}_{max}] \tag{5.12}$$

$$\mathcal{P}: \quad \hat{\mathbf{Z}} = \mathbf{V} \star \mathbf{Z} \tag{5.13}$$

$$\mathcal{R}: \quad \hat{\mathbf{U}} = \hat{\mathbf{Z}} \cdot \mathbf{U}_c, \tag{5.14}$$

where AvgCPool$(\cdot)$ and MaxCPool$(\cdot)$ denote the channel-wise average and max pooling operations, $[\cdot;\cdot]$ represents concatenation along the channel dimension, $\star$ indicates the convolution operation, and $\mathbf{V} \in \mathbb{R}^{1\times1\times1\times2\times1}$ denotes the convolutional weights. These two blocks are joined sequentially by sending the input through the channel attention block first and then forwarding the output via the spatial attention block.

5.2.5 Project & Excite Module

The recalibration blocks discussed previously were primarily designed for 2D tasks and may not be optimally suited for 3D segmentation tasks. The cSE, scSE, and CBAM blocks, which compress the spatial information of a volumetric feature map into a single scalar value per channel, may not adequately capture the essential spatial information of large-sized 3D inputs, especially in the first and last layers of an encoder-decoder architecture. To address this limitation, we propose the Project & Excite (PE) module, which aims to preserve valuable

spatial information within the projection operation and learn inter-dependencies between projections across different channels for recalibration.

The compression operation, denoted as $\mathcal{C}(\cdot)$, is divided into three separate projection operations along the spatial dimensions: $\mathcal{C}_H(\cdot)$, $\mathcal{C}_W(\cdot)$, and $\mathcal{C}_D(\cdot)$. These projection operations compress the feature map along each spatial dimension to obtain the projected feature maps $\mathbf{Z}_h \in \mathbb{R}^{C\times H}$, $\mathbf{Z}_w \in \mathbb{R}^{C\times W}$, and $\mathbf{Z}_d \in \mathbb{R}^{C\times D}$. For simplicity, a single channel of the projected feature maps is denoted as $\mathbf{z}_{l,c}$, with l referring to the spatial dimension (e.g., h, w, or d).

The projection operation utilizes pooling operations, such as average or max pooling, to compress the feature map along the spatial dimensions. As an example, we describe the projection operation using average pooling:

$$\mathcal{C}_H: \quad \mathbf{z}_{h,c}(i) = \frac{1}{W}\frac{1}{D}\sum_{j=1}^{W}\sum_{k=1}^{D}\mathbf{U}_c(i,j,k), \tag{5.15}$$

$$\mathcal{C}_W: \quad \mathbf{z}_{w,c}(j) = \frac{1}{H}\frac{1}{D}\sum i = 1^{H}\sum_{k=1}^{D}\mathbf{U}_c(i,j,k), \tag{5.16}$$

$$\mathcal{C}_D: \quad \mathbf{z}_{d,c}(k) = \frac{1}{H}\frac{1}{W}\sum i = 1^{H}\sum_{j=1}^{W}\mathbf{U}_c(i,j,k), \tag{5.17}$$

where $i \in 1,\ldots,H$, $j \in 1,\ldots,W$, and $k \in 1,\ldots,D$.

The projected feature maps $\mathbf{Z}_h$, $\mathbf{Z}_w$, and $\mathbf{Z}_d$ are then broadcasted to the shape $H \times W \times D \times C$ and added together to obtain the combined projected feature map $\mathbf{Z}$. The processor $\mathcal{P}(\cdot)$, which comprises two convolutional layers with a ReLU activation, is then given this merged feature map. While the second layer returns the channel dimension to its original size, the first layer decreases the number of channels by a reduction factor r. The definitions of the process and recalibrate operations are:

$$\mathcal{P}: \quad \hat{\mathbf{Z}} = \mathbf{v}_2 \star \delta(\mathbf{v}_1 \star \mathbf{Z}), \tag{5.18}$$

$$\mathcal{R}: \quad \hat{\mathbf{U}} = \sigma(\hat{\mathbf{Z}}) \odot \mathbf{U}, \tag{5.19}$$

where $\star$ represents the convolution operation, $\odot$ denotes point-wise multiplication, $\mathbf{v}_1 \in \mathbb{R}^{1\times1\times1\times\frac{C}{r}}$ and $\mathbf{v}_2 \in \mathbb{R}^{1\times1\times1\times C}$ are the convolution weights.

The final output of the PE block, $\hat{\mathbf{U}}$, is obtained by performing an element-wise multiplication between the feature map $\mathbf{U}$ and the recalibrated feature map $\hat{\mathbf{Z}}$. This innovative approach preserves more crucial spatial information within the projection operation, leading to improved performance in the recalibration process. The Project & Excite module is a valuable contribution to 3D segmentation tasks as it effectively captures spatial and channel information in the recalibration operations.

5.2.6 Integration into F-CNN Architectures

Existing F-CNNs networks can easily incorporate calibration blocks. According to Hu et al. [62], they are often placed after the nonlinearity after a convolutional layer. We use the same positioning technique in both our PE and 3D extensions. The various placements of PE blocks inside a typical encoder-decoder-based network are shown in Figure 5.4 and are discussed in Section 5.4.1.

Hu et al. also showed how well cSE blocks may be included in residual networks. In this study, as described in Section 5.4.2, we examine the performance of 3D recalibration blocks inside a residual 3D FCNN.

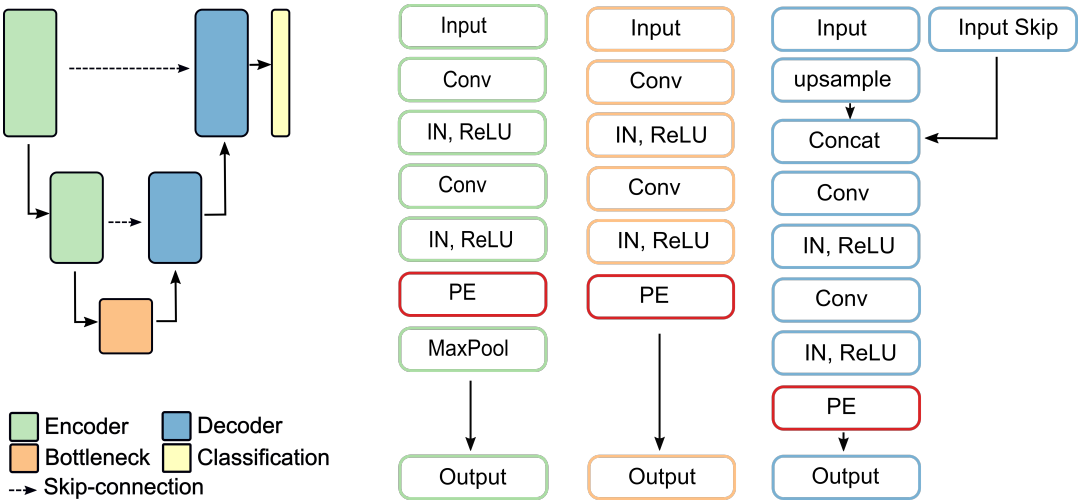

Figure 5.4. Left: Schematic representation of a 3D U-Net, which includes classification, bottleneck, encoder, and decoder layers. Right: Project & Excite blocks, denoted as PE, are positioned in the bottleneck, encoder, and decoder blocks to show how they are integrated into the network. Additionally, we use instance normalization ('IN') in our studies. © IEEE, 2020

5.3 Experimental Setup

5.3.1 Whole-Brain Segmentation of MRI Scans

Throughout the experiments, we utilize three distinct brain MRI datasets. To ensure consistency, we preprocess all MRI scans using FreeSurfer [39] by resampling them to a voxel resolution of $1 \times 1 \times 1mm$. Neuromorphometrics, Inc provided manual annotations for all brain datasets. The objective is to segment 32 cortical and subcortical structures in the 3D MRI brain scans.

The first dataset we use is the Multi-Atlas Labeling Challenge (MALC) dataset [82], which is a subset of the OASIS dataset [100]. It comprises 30 T1 MRI volumes from different subjects. We use this dataset for training our networks. Considering the limited data availability, we employ 5-fold cross-validation. In each fold, we use 24 scans for training and reserve 6 scans for testing. Additionally, two scans from the training set are set aside as a validation set during the model training process.

The second dataset, ADNI-29, consists of 29 scans from the ADNI dataset [70]. It is carefully curated to include a balanced distribution of subjects with Alzheimer's Disease and control subjects. The scans are acquired using both 1.5T and 3T scanners, and the presence of pathology presents a challenging segmentation task. The third dataset, CANDI, is a subset of the CANDI dataset [75], consisting of 13 brain scans of children aged 5-15 with psychiatric disorders. Some scans in this dataset exhibit severe motion artifacts. The ADNI and CANDI datasets are used as unseen datasets for evaluating model generalizability.

5.3.2 Whole-Body Segmentation of CT Scans

In this investigation, we use contrast-enhanced whole-body CT images from the Visceral dataset [161]. This dataset comprises 20 annotated scans with a voxel resolution of 2mm^3. Our objective is to segment organs inside the thorax and abdomen, with a focus on 14 particular organs. We use a 5-fold cross-validation procedure to evaluate our method. Each fold has 16 training scans and 4 testing scans. During the training phase, two scans from the training set are set aside as a validation set inside each fold. In addition, one scan from the test fold is saved as a validation set.

5.3.3 Baseline Architectures

3D U-Net [18], V-net [109], and VoxResNet [13] are three frequently used 3D F-CNN designs. While 3D U-Net and V-Net have similar encoder-decoder architectures, VoxResNet has a unique design with side supervision. Our suggested PE blocks are evaluated using one encoder-decoder architecture (3D U-Net) and one side-supervision architecture (VoxResNet).

3D U-Net

The segmentation network employed in our experiments is a 3D U-Net architecture consisting of an encoding path and a decoding path connected by skip connections. This network architecture is depicted in Figure 5.4. To reduce memory demand during training and ensure the feasibility of training on our available GPUs, we have modified the original design to reduce the number of parameters.

Our modified 3D U-Net architecture includes three encoder blocks and three decoder blocks. Downsampling is performed in the first two encoder blocks, while upsampling is performed in the last two decoder blocks. Each encoder and decoder block consists of two convolutional layers with a kernel size of $3 \times 3 \times 3$. Additionally, we have reduced the number of output channels at each encoder and decoder block to half the size used in the original 3D U-Net architecture. For example, the two convolutions in the first encoder block have channel numbers of 16 and 32 instead of 32 and 64. These modifications help to reduce the computational requirements and memory usage while maintaining the overall architecture and performance of the 3D U-Net.

VoxResNet

VoxResNet, proposed by Chen et al. [13], is a 3D residual network architecture specifically designed for volumetric brain segmentation. The key component of VoxResNet is the VoxRes module, a residual block composed of two 3D convolutional layers. The network architecture includes three downsampling operations performed using strided convolutions with a stride of 2, and upsampling is achieved using transposed convolutions. In addition to the final classifier, the network also includes four auxiliary classifiers. The final classifier is obtained by summing the outputs of all the auxiliary classifiers. The use of auxiliary classifiers with different receptive fields allows for deep supervision, which aids in segmenting structures of varying sizes. To incorporate recalibration blocks into VoxResNet, we have positioned them before each downsampling step. This placement allows for recalibration at different scales within the network. Following the convention established in [62], we have placed the recalibration blocks within the residual blocks of VoxResNet. The architecture of VoxResNet and the placement of recalibration blocks are illustrated in Figure 5.5.

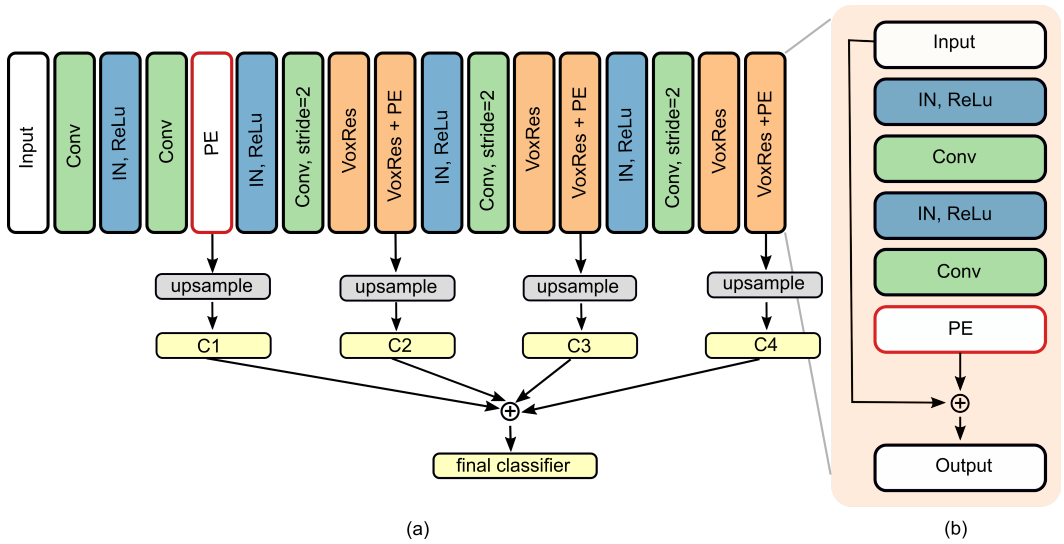

Figure 5.5. Project & Excite blocks, denoted as PE, are placed within the VoxResNet [13] model. (a) Diagram of the VoxResNet architecture with side supervision (C1-C4), adapted from [13]. We choose to integrate PE modules before each downsampling step. In this architecture that means in every second VoxRes module, denoted as VoxRes + PE. The details of the VoxRes module areshown on the side (b). It includes instance normalization (IN), ReLU, and convolutional layers. Before the residual connection, the PE module is integrated. © IEEE, 2020

5.3.4 Training Parameters and Implementation Details

We decided on a batch size of 1 for training to handle the large input size. In our first tests, we found that applying the running mean during testing after batch normalization resulted in noisy validation loss and decreased performance on the test set. In place of it, we used Instance Normalization [164]. In addition, we discovered that for our particular tasks, instance normalization worked better than group normalization. We used Stochastic Gradient Descent (SGD) with a momentum of 0.9 for optimization. We trained each model for 120 epochs. The initial learning rate was set to 0.1, and when the validation loss plateaued for more than 10 epochs, the learning rate was decreased by a factor of 10. Data augmentation techniques, such as elastic deformations and random rotations, were applied to the training set.

To address the class imbalance issue, we combined Cross-Entropy and Dice loss. As demonstrated in [143], the Cross-Entropy loss was weighted by median frequency balancing. Following the approach of Chen et al. [13], we incorporated a weighting factor for the loss of the auxiliary classifiers during the training of VoxResNet. The weighting factor was initially set to 1 and decreased by a factor of 2 every 10 epochs, with a minimum value of 0.001. All models were trained on Nvidia Quadro P6000 GPUs with 24GB of RAM or Nvidia TitanXP GPUs with 12GB of RAM. We utilized the PyTorch checkpoint functionality when training on the TitanXP GPU, which prevents the storage of intermediate activations during the forward pass and recomputes the activations during the backward pass. While this method increases computation time, it is advantageous for training deep neural networks with inputs that require substantial memory resources and volumetric data.

5.3.5 Evaluation Metrics

We have chosen two evaluation metrics to assess the performance of our segmentation method: the volumetric Dice similarity coefficient (DSC) as an overlap metric and the surface Dice coefficient as a boundary metric. The volumetric Dice coefficient measures the overlap between the predicted segmentation mask and the ground truth labels in terms of volume. It is a commonly used metric in medical image segmentation. However, it is worth noting that the volumetric Dice coefficient is insensitive to minor segmentation errors in boundary regions, particularly for large organs. We also employ the surface Dice coefficient [116] to address this limitation. The surface Dice coefficient calculates the overlap of the surfaces of the predicted and ground truth segmentations, considering a specific boundary tolerance. Unlike the volumetric Dice coefficient, the surface Dice coefficient penalizes small segmentation errors in boundary regions. As we lacked multiple manual segmentations to determine the optimal tolerance parameter for each structure, we set the tolerance parameter to the lowest possible value, which is the resolution of the scans, for all structures. Using a combination of evaluation metrics, we can obtain a comprehensive assessment of the segmentation performance and identify areas for improvement.

5.4 Results and Discussion

We use the following criteria to assess the performance of PE blocks: We conduct an ablation study of PE blocks using the MALC dataset, looking at architectural decisions such as pooling and aggregation procedures, the selection of the hyperparameter r, and the placement of PE blocks within the network architecture. The performance of PE blocks is then compared against the 3D extensions of current calibration techniques. We apply all trained models on the ADNI and CANDI datasets to evaluate the performance on untested data. The next step is to see if additional segmentation jobs can benefit from the usage of PE blocks. In order to accomplish the goal of whole-body segmentation on the Visceral dataset, we apply PE blocks without altering any hyperparameters. We employ the 3D U-Net [18] as our baseline architecture in all of these experiments. Finally, we implement PE blocks into VoxResNet [13] and test their effectiveness on the MALC dataset.

Table 5.2. Comparison of various pooling methods and aggregation strategies used within the Project & Excite (PE) block to combine projection vectors. The evaluation metric utilized is the volumetric Dice coefficient, with the mean and standard deviation of the results reported. The PE modules were incorporated into the 3D U-Net architecture, trained, and evaluated on the MALC dataset.

	Pooling		
Aggregation	Avg	Max	Avg&Max
Add	**0.854 ± 0.075**	0.819 ± 0.194	0.848 ± 0.075
Max	0.853 ± 0.075	0.820 ± 0.175	0.817 ± 0.088
Mult	0.844 ± 0.101	0.798 ± 0.176	0.808 ± 0.164

5.4.1 Architecture Ablation Study and Hyperparameters

Pooling and Aggregation Strategy of PE Blocks

We examine various pooling strategies and aggregation techniques for projection vectors within PE blocks. We compare average pooling, max-pooling, and a combination of the two for the projection operation. In the combined pooling method, we use average pooling as described in the Methods section and execute three max-pooling procedures along distinct dimensions. The resultant average and maximum projection vectors are then expanded to the size of the original feature map and independently processed by the shared convolutional layers. Before passing through the sigmoid layer, the element-wise summation combines the recalibration maps generated by the two pooling methods. In addition, we investigate various strategies for combining the three projection vectors. We evaluate addition, element-wise max operation, and element-wise multiplication as aggregation methods.

The results of these experiments are presented in Table 5.2. We observe that average pooling yields the best performance across all three aggregation strategies. When using addition as the aggregation method, we find that the combination of average and max pooling also performs well. However, the performance is lower when using max or multiplication for aggregation. Based on these findings, we select addition as the aggregation strategy for its computational efficiency and ability to be computed in place.

Hyperparameter r

As mentioned in subsection 5.2.5, the hyperparameter r regulates the channel dimension reduction within the Excitation operator. On the MALC dataset, we compare the performance of the 3D U-Net with integrated PE blocks for various r values. We experiment with r values of $2, 4, 8, 16$ and find that $r = 8$ produces the best results. We see a similar pattern for 3D cSE, sSE, and CBAM blocks and, as a result, set $r = 8$ for these blocks. We set $r = 2$ for 3D scSE since it improves performance.

Position of Project & Excite Blocks

In this experiment, we use the 3D U-Net to study the best arrangement of the PE blocks within the F-CNN architecture. We investigate six possible PE block placement configurations: PE blocks are placed after each encoder block (P1), PE blocks after each decoder block (P2), PE blocks occur after the bottleneck block only (P3), PE blocks are placed after all encoder and

decoder blocks (P4), PE blocks after each encoder block and the bottleneck (P5), and finally PE blocks are placed after all encoder/decoder and bottleneck blocks (P6).

All six configurations' findings are reported in Table 5.3 and compared to the baseline 3D U-Net model. To begin, we see that inserting the PE blocks after the encoder (P1) and bottleneck (P3) blocks increases the Dice similarity coefficient (DSC) by 1 percentage point, however placing them after the decoder (P2) blocks has no effect on performance. Second, we discover that adding the PE blocks after each encoder and decoder block (P4) resulted in a DSC improvement of 0.026. This implies that when the encoder blocks also include PE blocks, the PE blocks at the decoder have a favorable effect. Furthermore, we find that inserting the PE blocks after the encoder blocks and the bottleneck (P5) results in a DSC boost of 0.018. This shows that the encoder and bottleneck PE blocks collaborate more effectively. Finally, by arranging the PE blocks after all other blocks (P6), we see a 0.03 improvement in DSC, which is more than the previous setups. As a consequence, this setup is chosen for our investigations. Furthermore, we evaluated the placement of PE blocks after each convolutional layer inside the encoder, decoder, and bottleneck, but we found no gain in performance. Based on these data, we infer that putting the PE blocks after all of the encoder, decoder, and bottleneck blocks (P6) produces the best results and is, thus, the method we will use in our experiments.

Table 5.3. Average Dice score and standard deviations on the MALC dataset resulting from the placement of PE blocks within various layers of the 3D U-Net. A checkmark in a column indicates that the PE block was positioned after each of the corresponding layers.

	Position of PE block			
	Encoders	Bottleneck	Decoders	Mean Dice ± std
3D U-Net	✗	✗	✗	0.823 ± 0.142
P1	✓	✗	✗	0.837 ± 0.127
P2	✗	✗	✓	0.825 ± 0.148
P3	✗	✓	✗	0.835 ± 0.115
P4	✓	✗	✓	0.849 ± 0.088
P5	✓	✓	✗	0.841 ± 0.113
P6	✓	✓	✓	$\mathbf{0.854 \pm 0.075}$

5.4.2 Comparison of 3D Recalibration Blocks

Brain Segmentation

We present the results of whole-brain segmentation in Table 5.4 for two different 3D F-CNN architectures: 3D U-Net [17] and VoxResNet [13]. We integrated PE blocks and other recalibration blocks, namely 3D cSE, 3D sSE, 3D scSE, and 3D CBAM, into both architectures. Due to the significantly increased memory requirement of dilated convolutions, comparing to a 3D version of SegSE [124] was not feasible on our GPUs. Dilated convolutions have a higher memory demand than normal convolutions in 2D, and this effect becomes even more pronounced in 3D due to the cubic scaling of complexity. The computational and memory requirements of 3D dilated convolutions on whole volume inputs would exceed the capabilities of our available GPUs. Therefore, we could not directly evaluate the performance of SegSE in

Table 5.4. Comparison of 3D U-net and VoxResNet segmentation performance on the MALC test set using several 3D recalibration blocks and our suggested Project & Excite block. Volumetric and surface Dice scores for chosen classes are averaged across the hemispheres. Structure names are abbreviated due to space constraints, Gray matter = GM, white matter = WM, inferior lateral ventricle = Inf. LV., amygdala = Amygd., accumbens = Acc..

	3D U-Net [17]											
	Volumetric Dice						Surface Dice					
	Mean ± std	WM	GM	Inf.LV	Amygd.	Acc.	Mean ± std	WM	GM	Inf.LV	Amygd.	Acc.
baseline	0.823 ± 0.142	0.918	0.904	0.382	0.785	0.529	0.928 ± 0.078	0.981	0.975	0.632	0.921	0.877
3D cSE [62, 187]	0.845 ± 0.102	0.920	**0.907**	0.488	0.787	0.754	0.938 ± 0.061	0.981	0.975	0.704	0.920	0.943
3D sSE [144]	0.849 ± 0.077	0.918	0.904	0.618	**0.795**	0.751	0.946 ± 0.022	0.979	0.973	0.890	0.927	0.939
3D scSE [144]	0.835 ± 0.115	0.919	0.905	0.554	0.794	0.527	0.933 ± 0.076	**0.982**	**0.976**	0.805	**0.938**	0.669
3D CBAM [176]	0.831 ± 0.125	0.918	0.903	0.488	0.792	0.525	0.921 ± 0.088	0.978	0.971	0.709	0.925	0.661
Project & Excite	**0.854 ± 0.075**	**0.921**	0.906	**0.627**	0.794	**0.757**	**0.951 ± 0.022**	0.981	0.975	**0.893**	0.929	**0.948**

	VoxResNet [13]											
	Volumetric Dice						Surface Dice					
	Mean	WM	GM	Inf.LV	Amygd.	Acc.	Mean	WM	GM	Inf.LV	Amygd.	Acc.
baseline	0.855 ± 0.076	0.922	0.908	0.621	0.779	0.769	0.938 ± 0.022	0.933	0.939	0.895	0.909	0.951
+ 3D cSE [62, 187]	0.859 ± 0.071	0.926	0.912	0.653	0.779	0.768	0.942 ± 0.021	0.941	0.946	0.903	0.911	**0.952**
+ 3D sSE [144]	0.852 ± 0.072	0.916	0.903	0.644	0.775	0.758	0.934 ± 0.022	0.920	0.930	0.898	0.908	0.944
+ 3D scSE [144]	0.828 ± 0.106	0.911	0.899	0.552	0.763	0.599	0.908 ± 0.054	0.908	0.923	0.799	0.895	0.752
+ 3D CBAM [176]	0.853 ± 0.070	0.919	0.905	0.652	0.781	0.759	0.934 ± 0.022	0.926	0.933	0.901	0.916	0.942
+ Project & Excite	**0.861 ± 0.072**	**0.941**	**0.926**	**0.657**	**0.789**	**0.771**	**0.947 ± 0.023**	**0.984**	**0.977**	**0.904**	**0.922**	**0.952**

the 3D setting. We maintained the same placement of the other blocks in the architecture as ours.

We report the volumetric and surface Dice coefficients, presenting the mean Dice scores across all classes and those for some selected classes. For simplicity, we average the Dice coefficients over both hemispheres.

For the 3D U-Net architecture, we observe that the overall mean Dice score increases by 0.02 when using 3D cSE and 3D sSE, while PE blocks lead to an increase of 0.03, demonstrating their effectiveness. The modules combining channel and spatial recalibration (CBAM and scSE) only result in an improvement of 0.01, suggesting their 3D versions may not be as efficient as the corresponding 2D versions. When examining larger structures, such as white and grey matter, the Dice score improvement is minimal across all blocks. This is likely because the segmentation accuracy for these large structures is already quite high, with Dice scores around 0.92 and 0.90 for white and gray matter, respectively, suggesting that the performance of the 3D U-Net might have reached its limit in these cases. We further investigate the impact of PE blocks on smaller structures, including inferior lateral ventricles, amygdala, and accumbens, which are difficult to segment, as demonstrated by the relatively low Dice scores of the 3D U-Net. We find the best performance for PE and 3D sSE models, where the increase in performance for large structures is modest, but adding these modules can lead to a performance boost for smaller classes.

For the VoxResNet architecture, we find that it surpasses the 3D U-Net by 0.03 in average DSC, primarily due to better performance for smaller structures like the inferior lateral ventricle and accumbens. We believe this is achieved because of the deep supervision, which makes the VoxResNet architecture more suitable for segmenting smaller structures than the 3D U-Net. We observe increased volumetric and surface Dice scores using PE blocks for all structures. While 3D cSE also improves performance, the other blocks, 3D sSE, 3D scSE, and 3D CBAM,

result in an overall decline in performance. PE blocks contribute to increased performance for smaller structures, although it is not as effective as in 3D U-Net. A possible explanation for this is the better performance of the baseline VoxResNet on small structures like inferior lateral ventricles and accumbens, where a further increase in performance is much harder to achieve. Interestingly, although the baseline VoxResNet's performance on the white and grey matter is similar to 3D U-Net, PE blocks boost performance here.

In summary, we demonstrate the effectiveness of PE blocks when integrated into two distinct architectures, 3D U-Net and VoxResNet. We find the best performance for PE and 3D sSE models when combined with 3D U-Net, where the increase in performance for large structures is modest, but adding these modules can lead to a considerable performance boost for smaller classes. The performance of VoxResNet is overall better than 3D U-Net but can be further boosted with PE blocks. Moreover, as in 2D networks, a combination of channel and spatial SE modules works well. However, they must be carefully combined and designed explicitly for 3D architectures, such as our PE blocks.

Table 5.5. Comparative analysis between 3D U-Net models with added 3D recalibration blocks and models with additional convolutional layers. The maximum GPU RAM use during training, the average inference time (including forward pass and evaluation metric computation), and the mean volumetric Dice score are displayed. For this evaluation, we used a single Titan XP GPU to measure the time and memory requirements. The top performing model is highlighted in bold.

	Mean Dice $\pm$ std	# Params	Memory	Time
3D-Unet [18]	0.823 ± 0.142	$5.57 \cdot 10^6$	6.7 GB	$0.56s$
+ 3D cSE [62, 187]	0.845 ± 0.102	$+0.50\%$	7.6 GB	$0.85s$
+ 3D sSE [144]	0.849 ± 0.077	$+0.01\%$	7.7 GB	$0.56s$
+ 3D scSE [144]	0.835 ± 0.115	$+1.98\%$	8.7 GB	$0.87s$
+ 3D CBAM [176]	0.831 ± 0.125	$+0.50\%$	8.2 GB	$1.19s$
+ Project & Excite	$\mathbf{0.854 \pm 0.075}$	$+0.50\%$	8.7 GB	$0.79s$
+ Encoder/Decoder	0.849 ± 0.086	$+39.7\%$	6.8 GB	$0.60s$
+ 2 Conv layers	0.839 ± 0.115	$+3.97\%$	6.7 GB	$0.57s$

Model Complexity

This section investigates the increase in model complexity caused by including PE blocks into the 3D U-Net design. In Table 5.5, we compare PE blocks with 3D cSE [187], 3D sSE, 3D scSE, and 3D CBAM, with findings reported on the MALC dataset. We discover that, whereas PE blocks, CBAM, and 3D cSE blocks all result in the same 0.5% increase in model complexity, PE blocks provide a larger accuracy gain at the same cost. While 3D sSE has the least complexity increase, it does not perform as well as PE blocks. Because of the smaller reduction factor of 2, 3D scSE blocks result in a greater parameter increase, as mentioned in subsection 5.4.1.

It might be claimed that the improved performance is due to the increased complexity, which could also be accomplished by including additional convolutional layers. We investigated this further by doing two more tests. First, we added an additional encoder and decoder block to the design, which raised model complexity by almost 40% and resulted in performance comparable to adding 3D sSE blocks. Second, we merely added two more convolutional layers at the second encoder and second decoder, resulting in a minor increase in model

complexity ($\sim 4\%$). In this situation, we see a performance boost equivalent to scSE, with double the parameter increase, but it still falls short of the performance of PE blocks. As a result, recalibration blocks outperform just adding convolutional layers.

We also compare the models regarding maximum GPU RAM utilization during training and the time required to segment a single scan. The PE and scSE blocks use more GPU RAM than other modules. On a Titan XP GPU, the inference time for all modules except CBAM is less than one second. Integrating PE modules results in quicker model convergence, which may be important when GPU time is restricted. We found a mean overall DSC of 0.845 for PE models after training was stopped at 80 epochs, compared to 0.796 for the baseline 3D U-Net.

Table 5.6. Models that were trained on the MALC dataset are evaluated on new datasets, namely CANDI and ADNI, to evaluate their generalizability. We assess the segmentation performance in terms of volumetric and surface Dice scores for chosen structures. Structure names are abbreviated due to space constraints, Gray matter = GM, white matter = WM, inferior lateral ventricle = Inf. LV., amygdala = Amygd., accumbens = Acc.. All scores are averaged over the two hemispheres.

	ADNI											
	Volumetric Dice						Surface Dice					
	Mean ± std	WM	GM	Inf.LV	Amygd.	Acc.	Mean ± std	WM	GM	Inf.LV	Amygd.	Acc.
3D U-net	0.743 ± 0.146	0.832	0.780	0.351	0.687	0.414	0.820 ± 0.112	0.907	0.889	0.498	0.823	0.590
+ 3D cSE	0.769 ± 0.101	**0.837**	0.787	0.471	0.694	**0.648**	0.855 ± 0.065	**0.914**	**0.896**	0.673	0.833	**0.886**
+ 3D sSE	0.764 ± 0.087	0.831	0.780	0.572	0.705	0.606	0.851 ± 0.045	0.907	0.886	0.823	**0.840**	0.846
+ 3D scSE	0.750 ± 0.131	0.835	0.783	0.484	0.692	0.383	0.831 ± 0.094	0.911	0.888	0.703	0.824	0.534
+ 3D CBAM	0.744 ± 0.125	0.831	**0.795**	0.460	0.650	0.434	0.837 ± 0.084	0.910	**0.896**	0.663	0.829	0.618
+ Project & Excite	**0.776 ± 0.080**	0.835	0.780	**0.618**	**0.698**	0.639	**0.867 ± 0.046**	0.913	0.892	**0.859**	0.839	0.883
	CANDI											
	Volumetric Dice						Surface Dice					
	Mean ± std	WM	GM	Inf.LV	Amygd.	Acc.	Mean ± std	WM	GM	Inf.LV	Amygd.	Acc.
3D U-net	0.675 ± 0.169	0.848	0.859	0.228	0.506	0.377	0.723 ± 0.128	0.884	0.876	0.414	0.526	0.512
+ 3D cSE	0.703 ± 0.139	0.848	0.856	0.324	0.535	**0.580**	0.755 ± 0.090	0.886	0.875	0.580	0.559	**0.762**
+ 3D sSE	0.690 ± 0.123	0.837	0.848	0.414	0.527	0.547	0.739 ± 0.085	0.866	0.864	0.736	0.538	0.721
+ 3D scSE	0.676 ± 0.167	0.849	0.861	0.347	0.483	0.350	0.726 ± 0.118	0.889	0.882	0.617	0.517	0.468
+ 3D CBAM	0.699 ± 0.151	0.842	0.850	0.332	0.528	0.391	0.747 ± 0.103	0.880	0.866	0.594	0.550	0.526
+ Project & Excite	**0.719 ± 0.126**	**0.858**	**0.865**	**0.427**	**0.555**	0.552	**0.780 ± 0.078**	**0.895**	**0.890**	**0.759**	**0.576**	0.730

5.4.3 Deployment on Unseen Datasets

In prior trials, we trained and evaluated models only on data from the MALC dataset. This experiment investigates a more realistic scenario in which the model is trained on MALC and applied to previously unseen datasets (ADNI and CANDI). We study the role of several recalibration blocks in getting dependable performance on these unseen datasets. The total mean volumetric and surface Dice scores for chosen structures on both unseen datasets are displayed by Table 5.6.

We observe a decline in performance for all models compared to those tested on MALC (presented in Table 5.4). This can be attributed to the limited training set and the domain shift between MALC, ADNI, and CANDI. Factors such as scanner types, imaging parameters, and patient demographics might differ among these datasets. We also note that the decrease in performance is more noteworthy for the CANDI dataset. Since CANDI contains scans of children only, while MALC consists of adult scans, this is expected. Overall, MALC and ADNI are more similar, except that ADNI includes scans of patients with Alzheimer's disease.

For the models tested on ADNI, we find that PE blocks lead to the most considerable improvement in Dice and surface Dice scores. As with the experiments on MALC, we observe the largest increase for smaller structures like inferior lateral ventricles. Interestingly, there is a substantial drop in performance on larger structures, such as white matter and grey matter, compared to the models tested on MALC. This drop is even more pronounced than on the CANDI dataset. We believe this could result from some scans of patients with Alzheimer's disease, which can cause cerebral cortex atrophy and affect the shape and volume of white and grey matter segmentations.

On the CANDI dataset, we observe that PE blocks outperform all other blocks on average and for all selected structures except the accumbens. Interestingly, while sSE blocks were the second-best module on the MALC dataset, they are outperformed by cSE on both unseen datasets. This suggests the effectiveness of PE blocks over sSE blocks and demonstrates their more reliable performance, even on unseen data from different distributions. However, the overall performance on these datasets has dropped considerably compared to MALC, indicating that the training data may not have been diverse enough and possibly too small. While this was not the focus of our work, it could be addressed in future research. In Figure 5.6, we present visualizations of the segmentation performance of PE models compared to the baseline 3D U-Net and other recalibration blocks for the CANDI dataset.

5.4.4 Experiments on Whole-Body Segmentation

We test the efficacy of PE blocks in the segmentation of thoracic and abdominal organs using contrast-enhanced whole-body CT scans to assess their applicability to a new task and modality. Table 5.7 displays the volumetric and surface Dice scores for the Visceral dataset for all models. Unlike the results found in brain datasets, we find that the 3D sSE model performs the worst on the Visceral dataset, with a mean Dice score roughly 0.06 lower than the baseline model. Similarly, neither the scSE nor the CBAM models outperform the baseline model.

When investigating larger structures such as the liver and right lung, we observe a pattern similar to brain segmentation, where all models' performance is on par with the baseline 3D U-Net. The segmentation accuracy for these organs is already quite high, and we believe that further improvement is challenging. This is consistent with the fact that lung segmentation in CT scans is relatively easier due to the clear contrast between air and surrounding tissue, while liver segmentation benefits from the contrast enhancement.

Following that, we concentrate on medium-sized organs, such as the right kidney, and smaller structures, such as the trachea and sternum, which are more difficult to segment precisely. Both the cSE and PE models demonstrate relatively moderate gains in Dice score improvements for the kidneys and trachea. Although the CBAM model performs best for trachea segmentation, its overall performance is mediocre. Both the cSE and PE models show a significant increase of roughly 0.3 in DSC for the sternum. However, the sSE model fails to segment the sternum fully. The 3D scSE and CBAM models' performance is similarly inadequate for this class. Because all of these modules entail compressing the channel dimension, these observations highlight the relevance of the information conveyed in the channel dimension. We conclude that PE models

give the greatest overall results and the most consistent performance across all structures, implying that PE blocks are more resilient.

We show visualizations of all segmentations in Figure 5.7. In particular, we show a thoracic slice with the lungs, aorta, trachea, and sternum segmented. The baseline model, sSE, scSE, and CBAM models all struggle to segment the sternum properly, while both the sSE and scSE models fail to segment the trachea.

Table 5.7. Comparison of 3D U-Net segmentation performance on the Visceral dataset with various 3D recalibration blocks. Volumetric and surface Dice scores are supplied for chosen classes, with the right side reported for the lungs and kidneys.

	Volumetric Dice						Surface Dice					
	Mean	Liver	Lung	Kidney	Trachea	Sternum	Mean	Liver	Lung	Kidney	Trachea	Sternum
3D U-Net [18]	0.810 ± 0.137	0.922	0.965	0.907	0.815	0.438	0.771 ± 0.121	0.755	0.924	0.857	0.895	0.481
+ 3D cSE [62, 187]	0.837 ± 0.091	0.929	**0.967**	0.916	0.817	**0.737**	0.805 ± 0.092	0.776	**0.936**	0.880	0.900	**0.799**
+ 3D sSE [144]	0.751 ± 0.268	0.925	0.964	0.919	0.347	0	0.711 ± 0.248	0.768	0.915	**0.896**	0.381	0
+ 3D scSE [144]	0.802 ± 0.147	0.927	0.966	0.914	0.659	0.419	0.763 ± 0.125	0.766	0.933	0.882	0.729	0.454
+ 3D CBAM [176]	0.797 ± 0.174	0.924	0.954	0.913	**0.831**	0.291	0.752 ± 0.156	0.750	0.901	0.875	0.902	0.316
+ Project & Excite	$\mathbf{0.844 \pm 0.088}$	**0.934**	**0.967**	**0.920**	0.822	0.733	$\mathbf{0.814 \pm 0.086}$	**0.779**	0.934	0.895	**0.905**	0.796

5.5 Conclusion

In this chapter, our main focus was on addressing the challenges associated with whole-volume medical image segmentation using 3D F-CNNs. These networks typically have limited depth and reduced feature capacity in order to manage the increased dimensionality. This is often achieved by reducing the number of convolutional layers and channels per layer. To enhance the performance of 3D F-CNNs, we explored the use of feature recalibration techniques. These techniques have shown promising results in 2D networks with minimal increase in model complexity. We initially extended several existing 2D recalibration blocks to the 3D domain. Additionally, we introduced the 'compress, process, recalibrate' framework, which allowed for easy comparison of different recalibration blocks. Finally, we proposed the Project & Excite (PE) module, specifically designed for 3D F-CNN architectures.

The PE module demonstrated improved segmentation performance while minimizing the increase in model complexity. Through extensive experiments conducted on multiple datasets and applications, we provided evidence that PE blocks outperform other recalibration blocks and are more effective than simply adding more convolutional layers in 3D F-CNNs. Interestingly, we observed that PE modules show a greater improvement in segmenting smaller structures than other recalibration blocks, which we attribute to the preserved spatial information within the projection operation. Consistent with our findings, PE blocks consistently performed well across various datasets and base architectures, while other recalibration blocks sometimes led to a decrease in performance. This confirms that PE blocks are an effective and robust design choice for 3D segmentation tasks, particularly when targeting small structures. We also acknowledge the limited availability of training data with expert annotations for medical image segmentation tasks, which poses a significant challenge. In the following chapters, we will address this problem by exploring self-training with limited annotations in Chapter 6 and learning with weakly supervised data in Chapter 7.

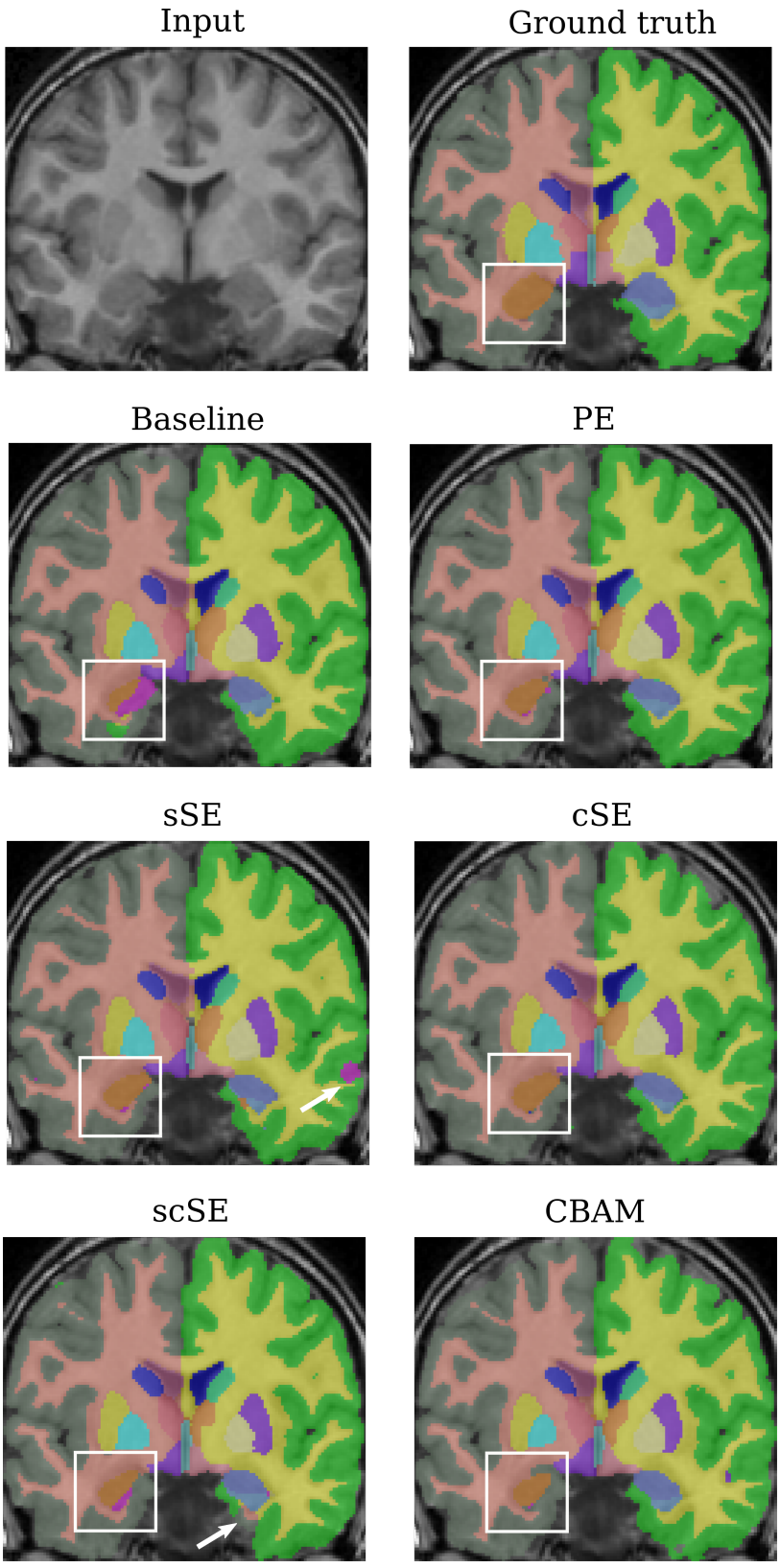

Figure 5.6. Visualisation of segmentation results on an example scan of the CANDI dataset. We compare the performance of the backbone segmentation model, a 3D U-Net, and various 3D recalibration blocks. The white boxes indicate an improvement due to using recalibration blocks over the baseline. White arrows point to areas where recalibration blocks lead to incorrect segmentations. © IEEE, 2020

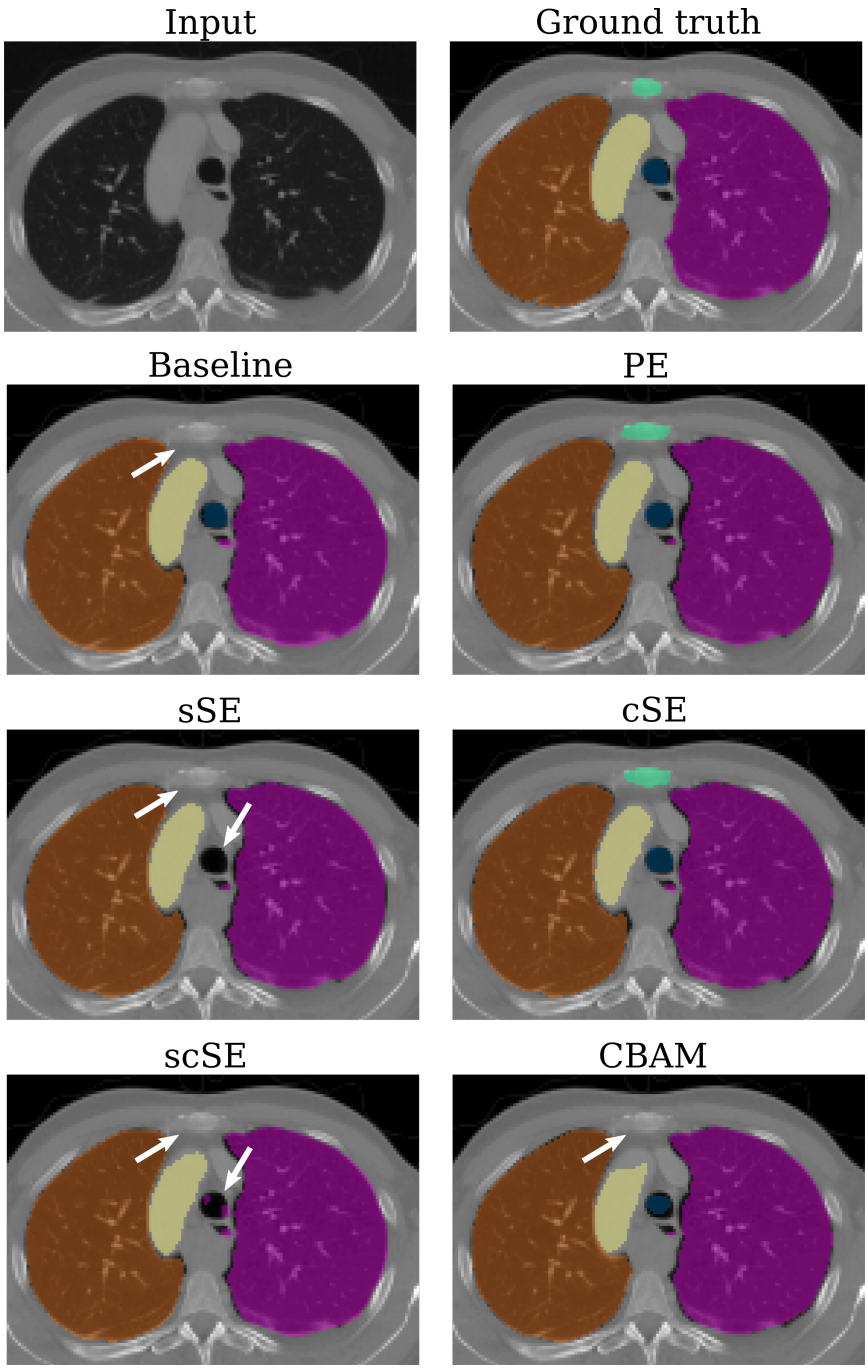

Figure 5.7. Visualization of segmentation performance on the Visceral dataset. We show an input CT scan of the thorax with gorund truth and baseline segmentations in comparison to several recalibration blocks. The white arrows indicate structures, notably the sternum, and trachea, that some models failed to segment correctly. Our PE model segments the sternum and trachea correctly. © IEEE, 2020

6 Self-Training with Uncertainty Dependent Label Refinement

Contents

6.1 Introduction

As the global population ages, the societal implications of dementia are becoming increasingly significant. Accumulating evidence suggests that changes related to aging, both functional and structural, can manifest as cerebral Small Vessel Disease (SVD). This SVD is increasingly recognized as playing a central role in escalating the risk of developing dementia [131]. White matter hyperintensities (WMHs) of presumed vascular origin have captured attention as a promising neuroimaging biomarker for SVD. WMH can be visualized using Fluid-Attenuated Inversion Recovery (FLAIR) MRI. Within these images, WMHs are characterized by regions that exhibit a diffuse pattern and are hyperintense compared to the adjacent white matter [127]. WMHs are further distinguished into periventricular and deep white matter hyperintensities based on location. Periventricular WMHs are found close to the brain's ventricles, while deep WMHs are located more internally within the white matter. The distinction between periventricular and deep WMHs is important, as their distribution patterns and severity may reflect different underlying pathological processes [49]. Additionally, it's worth mentioning that WMHs are not exclusive to dementia and SVD but are also significant in other neurological disorders such as multiple sclerosis. The extent and distribution of WMHs are crucial in gauging the severity of underlying conditions. In this context, the Fazekas scale [37] is often employed to assess the extent of WMHs. This scale rates WMHs on a scale from 0 to 3, where 0 indicates no WMHs, 1 represents "punctate" WMHs, 2 denotes "early confluent" WMHs, and 3 signifies "confluent" WMHs. A higher score on the Fazekas scale indicates more extensive white

matter damage, which could suggest a more advanced stage of SVD or other neurological disorders. Accurately identifying and assessing WMHs through neuroimaging techniques like FLAIR MRI is paramount in diagnosing and monitoring dementia and other neurological conditions. This also highlights the necessity for efficient image segmentation methods to assist in accurately identifying WMHs.

With the introduction of Convolutional Neural Networks (CNNs), there have been tremendous breakthroughs in segmenting WMHs [81]. However, it's important to recognize that the success of CNNs is linked to the richness and diversity of the training data. In situations with significant divergence between source and target domains, a phenomenon known as domain shift, CNNs might see a steep decline in their effectiveness. A recent study that compared various techniques revealed that conventional segmentation tools, which do not leverage deep learning, actually outperformed CNNs when it came to the accuracy of WMH labels in the context of domain shift [165]. These traditional approaches proved more robust in handling inconsistencies due to scanner variations and image artifacts. This highlights the necessity for deep learning-based models like CNNs to be powerful but also adaptable and robust when facing diverse data landscapes.

Unsupervised Domain Adaptation (UDA) is an approach in machine learning that tackles the challenge of adapting a model trained on one domain (source) to perform well on a different but related domain (target) without using any labeled data from the target domain. This is particularly valuable when obtaining labeled data for the target domain is costly or impractical, like in medical image analysis.

Self-training is a UDA methodology that has recently gained traction. The process starts with training a segmentation model on annotated source data. Then the trained model is applied to the target data, and the predictions are saved as so-called pseudo-labels. These self-generated pseudo-labels are then used for fine-tuning the model to the target data and iteratively updated. However, pseudo-labels may contain noise, as illustrated in Figure 6.1, making it critical to estimate the reliability of pseudo-labels and prevent error propagation.

This chapter explains our unique approach to self-training, which we call Self-training with Uncertainty Guided Label refinement (STRUDEL). The idea for this approach comes from research on brain lesion segmentation, which showed that uncertainty measurements might identify incorrect pixel-wise predictions [113]. To include the uncertainty in the pseudo-labels into the ongoing model revisions, we employ a Bayesian segmentation approach to measure the uncertainty (as demonstrated in Figure 6.1) and then incorporate it using an uncertainty-guided loss function. We recommend adding the output from the Lesion Prediction Algorithm (LPA) [148] to enhance the initial construction of pseudo-labels. This method has been shown to give reliable results in a number of different domains [165]. Our empirical evaluations employ STRUDEL with a U-Net as the underlying structure and a modified network with an amplified receptive field.

The results of our experiments in WMH segmentation across multiple datasets underscore the vital role of domain adaptation and highlight the notable improvement when uncertainty and LPA are integrated into the training procedure.

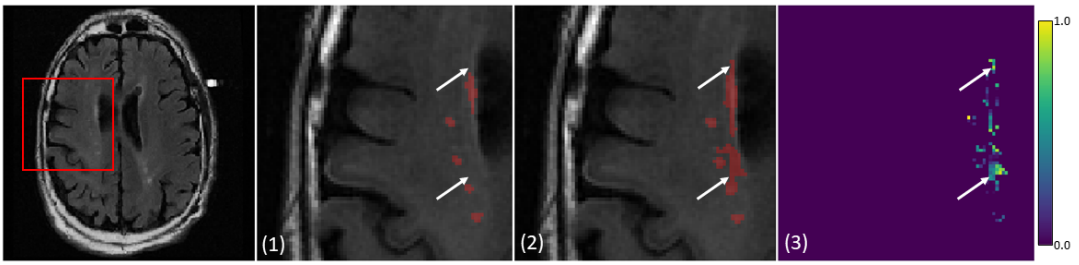

Figure 6.1. This graphic showcases a FLAIR scan featuring three components: (1) white matter hyperintensity segmentation ground truth, (2) pseudo-labels that over-segment regions not indicated in the ground truth, and (3) uncertainty map of the pseudo-label prediction. White arrows direct attention to instances of false positive predictions, which are associated with elevated levels of uncertainty. © Springer, 2021

6.1.1 Related Work

White Matter Hyperintensity Segmentation techniques have recently been evaluated in the WMH segmentation challenge [81]. The challenge had 20 submissions, and the 11 top-performing methods were deep-learning-based. The challenge also showed that 3D convolutional networks performed worse than 2D networks, and the top-performing methods used dropout. The higher performance of 2D models can be explained by the 2D nature of the given data. Specific inter-scanner robustness experiments indicated that enhancing the robustness of these methods is still necessary, which aligns with the findings in [165].

Unsupervised Domain Adaptation (UDA) methodologies enable the adaptation of models from a source domain without the necessity of direct supervision in the target domain, often employing adversarial learning techniques. The primary objective of these methods is to learn features that are invariant across domains by diminishing the differences between the source and target domains or transforming images from one domain to another. Several strategies have demonstrated their effectiveness in medical contexts [65, 71]. However, the complexity and diversity in training adversarial networks can make the process rather challenging and intricate [2, 129].

Self-training, on the other hand, presents an alternate approach to UDA, and its high efficiency has been highlighted in recent studies [188]. The fundamental idea behind self-training is to utilize the model's predictions from earlier stages as pseudo-labels that inform and guide the subsequent stages of network training. This technique has been investigated for semantic segmentation in both medical and non-medical fields, with cutting-edge results reported on standard datasets [115, 151, 178, 180, 189, 190]. For an in-depth categorization of methodologies that tackle the challenges of limited dataset annotations, one can refer to the comprehensive review by Tajbakhsh et al. [159]. The viability of integrating uncertainty guidance in the self-training paradigm was recently illustrated in a study on segmenting micro-CT scans with sparse annotations [185].

6.2 Methods

6.2.1 Problem Definition

Consider a dataset from the source domain, denoted as $\mathcal{S}$, comprising samples $\mathcal{X}^{\mathcal{S}} = \{\mathbf{X}_i^{\mathcal{S}}\}_{i=1}^{N}$ along with their respective labels $\mathcal{Y}^{\mathcal{S}} = \{\mathbf{Y}_i^{\mathcal{S}}\}_{i=1}^{N}$. Additionally, let there be an unlabeled dataset from the target domain, denoted as $\mathcal{T}$, with samples $\mathcal{X}^{\mathcal{T}} = \{\mathbf{X}_i^{\mathcal{T}}\}_{i=1}^{M}$. The goal of domain adaptation is the generation of high-quality label predictions within the target domain. In the context of unsupervised domain adaptation, the aim is to effectively weave the unlabeled samples from the target domain into the training of the network - it is common for M to be greater than N. Self-training achieves this by estimating pseudo-labels $\tilde{\mathbf{Y}}^{\mathcal{T}}$ for the target domain and iteratively refining these labels, which in turn enhances the overall learning procedure.

6.2.2 STRUDEL: Self-Training with Uncertainty

A graphical representation of the STRUDEL (Self-TRaining with Uncertainty DEpendent Label refinement) process is depicted in Figure 6.2, and the accompanying pseudo-code is provided in Algorithm 1. The first step involves training the backbone segmentation model on the source dataset $\left(X^{\mathcal{S}}, Y^{\mathcal{S}}\right)$ through conventional supervised learning. Following this, a random subset of the target sample, denoted as $r(X^{\mathcal{T}})$, with a size of P, is chosen without replacement and is subjected to the base model to generate initial pseudo-labels. Given the disparity between the domains, the initially produced pseudo-labels are likely to be of low quality, as domain shift leads to a drop in performance, as it was described in previous studies [81, 165]. As the study by Vanderbecq et al. [165] found that traditional WMH segmentation algorithms are more robust to domain changes, we advocate for the additional utilization of established software for pseudo-label generation, with LPA being a specific example. The refinement of the pseudo-labels is accomplished by employing a pixel-wise OR operation between the predictions of the base model and those of the LPA, resulting in the pseudo target labels, denoted as $\tilde{Y}^{\mathcal{T}} = \left\{\tilde{\mathbf{Y}}_i^{\mathcal{T}}\right\}_{i=1}^{P}$.

In step three, the base model is fine-tuned utilizing $\tilde{Y}^{\mathcal{T}}$. With this optimized model, we perform segmentation on the same randomly chosen sample as before, $r(X^{\mathcal{T}})$, producing enhanced-quality pseudo-labels. The underlying assumption is that the model is capable of generating predictions that surpass the quality of the initially noisy training labels, as outlined in [53]. Concurrently, we compute the segmentation uncertainty for every sample, denoted as $\mathbf{U}$. Subsequently, in step four, a new model is trained from scratch. During this phase, we enrich the training set by integrating the updated pseudo-labels $\tilde{Y}^{\mathcal{T}}$ along with the associated uncertainties $\mathbf{U}$. As highlighted in [188], training a new model at this stage offers benefits compared to just fine-tuning the existing one. The target scans and labels extracted from this new model are then added to the fixed training set, which initially contained only the annotated data from the source domain. Following this, the process moves to the next iteration, starting from step 2. In this iteration, the newly-trained model assumes the role of the base model. Another random subset is selected, and the model is used to derive pseudo-labels.

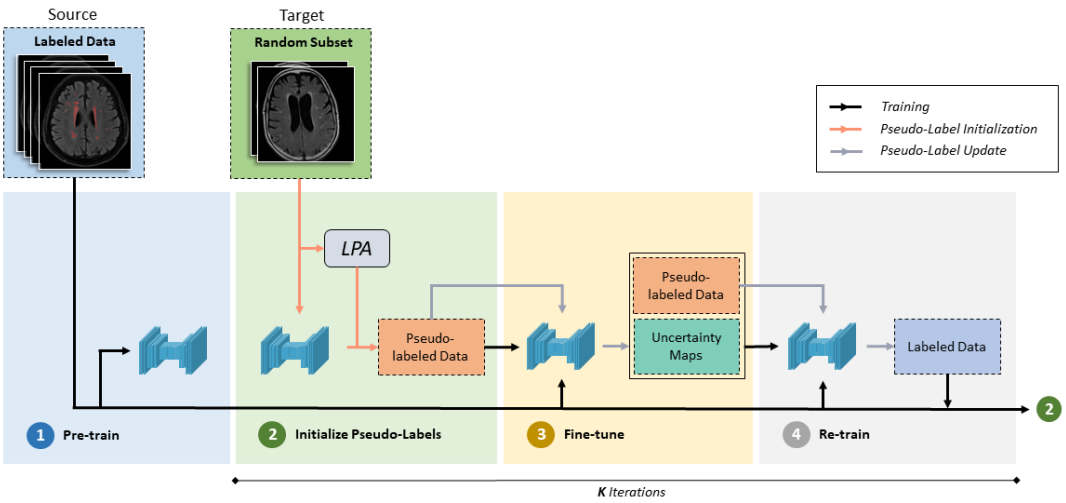

Figure 6.2. Illustration of our STRUDEL Self-Training pipeline for unsupervised domain adaptation. First, a labeled source dataset is used to pre-train the backbone segmentation network (Step 1). Then in step 2, the pre-trained network is fed with a random subset of the target data to produce initial pseudo-labels. The lesion prediction algorithm (LPA) output is also incorporated into pseudo-label initialization. Then in step 3, the network is fine-tuned using the pseudo target labels, generating source data and uncertainty maps. In step four, the network is re-trained from scratch using the source data and target data with pseudo-labels and uncertainty maps. After step 4, the random subset of target data with pseudo labels is added to the labeled dataset, and steps 2-4 are repeated k times with new random subsets drawn from the target data. © Springer, 2021

6.2.3 Uncertainty-Guided Pseudo-Labels

Label noise is a common disadvantage of inferring pseudo-labels. To make our approach more resistant to label noise, we recommend using uncertainty guidance, which rewards regions with low uncertainty and penalizes those with high uncertainty. Bayesian machine learning principles inspire our approach to estimating uncertainty. We utilize dropout [43] to estimate Monte Carlo (MC) samples. As the WMH challenge also recommends using models with dropout layers [81], we believe this will also improve segmentation precision. As a result, we use dropout layers to train the backbone segmentation network. During the testing phase, C stochastic forward runs are executed, facilitating the collection of Monte Carlo samples. The expectation over the MC samples, $\mathbb{E}(\hat{\mathbf{Y}})$, provides a more accurate label prediction, which is utilized to update the pseudo-label. This is also motivated by the results of the WMH challenge, where it was discovered that ensemble methods produce superior segmentation outcomes. In addition, calculating the variance across C MC samples gives us a pixel-by-pixel measure of the segmentation's uncertainty:

$$\mathbf{U}(\hat{\mathbf{Y}}) = \{\sigma_1, ..., \sigma_{H\times W}\} = \frac{1}{C}\sum_{i=1}^{C}\left(\hat{\mathbf{Y}}_i - \mathbb{E}(\hat{\mathbf{Y}})\right)^2, \tag{6.1}$$

where $\mathbf{U}(\hat{\mathbf{Y}})$ represents the uncertainty map, σ_i denotes the pixel-wise variance. The image's dimensions in terms of height and width are represented by H and W, respectively. The model prediction extracted from the ith Monte Carlo sample is denoted by $\hat{\mathbf{Y}}_i$. Anticipating that the uncertainty values will be small, we undertake a rescaling procedure to confine these values

within the [0, 1] range. To incorporate uncertainty during the training phase of the network, we devise an uncertainty-aware binary cross-entropy loss, termed UBCE loss:

$$\mathcal{L}_{\text{UBCE}} = -\frac{1}{H \times W} \sum_{n=1}^{H \times W} (1 - \sigma_n) \left[\tilde{y}_n \cdot log\left(\hat{y}_n\right) + (1 - \tilde{y}_n) \cdot log\left(1 - \hat{y}_n\right) \right], \tag{6.2}$$

where $\tilde{y}_n$ represents the n_{th} pixel of the pseudo-label $\tilde{\mathbf{Y}}$ and $\hat{y}_n$ represents the n_{th} pixel of the prediction $\hat{\mathbf{Y}}$. Note that the uncertainty-aware cross entropy $\mathcal{L}_{\text{UBCE}}$ is applied only to pseudo-labeled data during the re-training step (see Algorithm 1 line 11), whereas the standard cross entropy $\mathcal{L}_{\text{BCE}}$ is applied to fixed data samples. The definition of the combined loss function is:

$$\mathcal{L} = \mathcal{L}_{\text{Dice}} + \mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{UBCE}}. \tag{6.3}$$

Algorithm 1: Self-Training with Uncertainty on Noisy Labels

input : Source data $\mathcal{X}^{\mathcal{S}}$, Source labels $\mathcal{Y}^{\mathcal{S}}$, Target data $\mathcal{X}^{\mathcal{T}}$
output: Output model $\mathcal{M}_K$

1 $r() \leftarrow$ random sampler;
2 $\mathcal{M}_0 \leftarrow$ train base model with $(\mathcal{X}^{\mathcal{S}}, \mathcal{Y}^{\mathcal{S}})$;
3 $\mathcal{D}_{\text{fix}} \leftarrow (\mathcal{X}^{\mathcal{S}}, \mathcal{Y}^{\mathcal{S}})$; // initialize fixed training set
4 **for** $k \leftarrow 1$ **to** K **do**
5 $\quad \mathcal{X}_k^{\mathcal{T}} \leftarrow r(\mathcal{X}^{\mathcal{T}})$; // sample random subset
6 $\quad \tilde{\mathcal{Y}}_k^{\mathcal{T}} \leftarrow pixel - wise\_or(\mathcal{M}_{k-1}(\mathcal{X}_k^{\mathcal{T}}), LPA(\mathcal{X}_k^{\mathcal{T}}))$; // init. pseudo-labels
7 $\quad \mathcal{D}_k \leftarrow \mathcal{D}_{\text{fix}} \cup (\mathcal{X}_k^{\mathcal{T}}, \tilde{\mathcal{Y}}_k^{\mathcal{T}})$; // merge training data
8 $\quad \mathcal{M}_{k-1} \leftarrow$ fine-tune $\mathcal{M}_{k-1}$ with D_k;
9 $\quad \tilde{\mathcal{Y}}_k^{\mathcal{T}}, U_k \leftarrow \mathcal{M}_{k-1}(\mathcal{X}_k^{\mathcal{T}})$; // update labels and get uncertainty
10 $\quad \mathcal{D}'_k \leftarrow \mathcal{D}_{\text{fix}} \cup (\mathcal{X}_k^{\mathcal{T}}, \tilde{\mathcal{Y}}_k^{\mathcal{T}})$; // merge training data
11 $\quad \mathcal{M}_k \leftarrow$ re-train model with D'_k and uncertainty U_k;
12 $\quad \mathcal{D}_{\text{fix}} \leftarrow \mathcal{D}_{\text{fix}} \cup (\mathcal{X}_k^{\mathcal{T}}, \mathcal{M}_k(\mathcal{X}_k^{\mathcal{T}}))$; // update fixed training set
13 **end**
14 **return** $\mathcal{M}_K$;

6.2.4 Segmentation Backbone Architectures

To evaluate whether our self-training approach applies to different network architectures, we choose to assess STRUDEL on two distinct neural network structures for the segmentation model represented by M. Our first choice is the popular U-Net [140], which has demonstrated its performance in challenging segmentation tasks. Also, the deep learning models evaluated in the WMH challenge were mostly U-Net based [81].

To make the U-Net better suited for segmenting small datasets and small target structures like WMH, we developed a new version of the network. We refer to this improved version of the U-Net as the OctSE-Net. First, octave convolutions [16] are used in place of traditional convolution layers to frequency-wise separate feature maps. By providing more context and

reducing memory utilization, Octave convolutions can enhance segmentation performance. The second adjustment is the addition of recalibration blocks, such as the channel and spatial SE block [144], that can enhance accuracy by recalibrating feature maps and, as also presented in Chapter 5, have been shown to aid in segmenting small structures. Without substantially altering the model parameters, these changes improve the receptive field. This can enhance segmentation accuracy without encouraging overfitting, allowing for easier cross-domain generalization.

We add dropout layers after each convolutional block in both architectures for uncertainty estimation. Note that we chose 2D network architectures because the Magnetic Resonance Imaging (MRI) FLAIR scans used for training are 2D scans.

6.3 Experiments and Results

6.3.1 Datasets

For the source dataset, we make use of the data available through the WMH challenge (available at `https://wmh.isi.uu.nl`). As for the target dataset, we turn to the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (accessible at `http://adni.loni.usc.edu`). The WMH challenge [81] facilitates a uniform assessment of automated segmentation techniques, employing a standard dataset for testing. This dataset offers manually annotated data for 60 participants from three locations. Each participant's data comprises 3D T1-weighted images and 2D FLAIR images with multiple slices, both co-registered. As advised by [60, 162], our work involves bias-field corrected T1 scans and the original FLAIR scans. This dataset functions as the labeled source domain for our study.

The second dataset is procured from the Alzheimer's Disease Neuroimaging Initiative (ADNI). The ADNI-2 dataset [8] contains over 3,000 scans collected from 58 different sites, and it serves the role of multi-domain target data in our study. This dataset encompasses T1-weighted and 2D FLAIR images for every subject. We have ensured the linear alignment of these scans within each session using ANTs [4]. Additionally, T1 scans have undergone bias field correction via N4 normalization [163], leveraging the ANTs software. This particular dataset acts as our unlabeled target domain.

To evaluate our techniques quantitatively on the ADNI dataset, we selected a subset consisting of 30 subjects, chosen based on the scanner type and WMH lesion load, and subjected this subset to manual annotation. Of these annotated scans, 21 are reserved strictly for testing, while the remaining nine are employed to train alternative supervised segmentation methods, which we delve into in the subsequent section.

6.3.2 Implementation Details

Our framework and all the baseline experiments were developed utilizing PyTorch version 1.6.0. The models were trained with the Adam optimizer, maintaining default parameters

(betas=(0.9, 0.999), eps=1e-08), a learning rate set at 1e-4, and a batch size of 4. All image intensities were normalized to have a mean of zero and a variance of one, and the center cropping of axial segments produced a consistent pixel size of 192×192 across all datasets. During training, we applied common spatial augmentation techniques for regularization, including flipping, rotation, scaling, and elastic deformation.

For the self-training component, we fixed the size of the random subset in each iteration at $P = 35$. We established thresholds at 0.5 for network prediction and 0.75 for LPA when generating binary segmentation maps to form pseudo-labels. Employing a higher threshold for LPA assists in mitigating overly sensitive predictions. The training process from scratch spanned 80 epochs, while the fine-tuning phase was allocated 20 epochs. We designated the number of stochastic forward passes as $C = 10$ and observed that an increment in C did not improve segmentation performance. The dropout rate was adjusted to 0.2. We refrained from employing any explicit post-processing in the experiments. Training and testing were primarily performed on a Geforce Titan RTX GPU.

6.3.3 Experiments

To assess our method's capacity for robust domain transfer, we compare it to other approaches. First, we compare it to a **Base Model**, the backbone segmentation network trained on the source data and then applied to the target domain. Next, as we have access to a limited amount of labeled target data, we compare two training strategies that utilize this labeled target data. The **Joint Model** is trained on a dataset consisting of source and target labeled samples; therefore, it is trained on a larger dataset than other models, and we anticipate that it will outperform the base model, although domain adaptation models are anticipated to perform better. The **Fine-Tuning** model is trained on the source data and then fine-tuned using a limited quantity of labeled target data. We believe this model will outperform the joint model because it suffers from less data imbalance.

In addition, we evaluate two unsupervised domain adaptation (UDA) methods that employ pseudo-labels: **Self-Training** without uncertainty guidance and the proposed **STRUDEL** with uncertainty guidance, both of which employ LPA labels. In addition, we present results for STRUDEL without LPA and for LPA alone, with the threshold parameter set to 0.45, as recommended in [165] for ADNI. We evaluate the accuracy of segmentation for all experiments using evaluation metrics suggested by the WMH segmentation challenge [81]: (1) Dice Similarity Coefficient (DSC), (2) modified Hausdorff distance (95th percentile; H95), (3) absolute log-transformed volume difference (lAVD), (4) sensitivity for detecting individual lesions (Recall), and (5) individual lesion F1-score (F1).

6.3.4 Results & Discussion

The quantitative segmentation outcomes on the target domain of ADNI are detailed in Table 6.1, while a deeper examination of the DSC, represented as boxplots, is provided in Figure 6.3. To establish a reference, the DSC of the Base Models on the source dataset stands at 0.73 for U-Net and 0.76 for OctSE-Net. The direct application of these Base Models to the ADNI

Table 6.1. Comparison of different segmentation methods on the target data. The networks in the top part use the U-Net architecture as the segmentation backbone, and the models in the bottom part are based on OctSE-Net. The next three columns denote the type of data used for supervision during training. Where a tick indicates that the corresponding data has been used for training and an X means it has not been used. S stands for ground truth source labels, T stands for ground truth target labels and P stands for pseudo labels. The performance is assessed by Dice Coefficient (DSC), 95th Percentile Hausdorff Distance (H95), log transformed absolute volume difference (lAVD), lesion Recall and F1. We report mean ± standard deviation.

Methods	S	T	P	DSC ↑	H95 [mm] ↓	lAVD ↓	Recall ↑	F1 ↑
LPA	✗	✗	✗	0.57±0.16	23.1±23.4	0.71±0.49	0.81±0.16	0.39±0.18
U-Net								
Base Model	✓	✗	✗	0.45±0.28	27.1±37.5	1.09±1.70	0.67±0.32	0.48±0.21
Joint Model	✓	✓	✗	0.64±0.19	17.2±25.0	0.60±0.52	0.74±0.29	0.52±0.15
Fine-Tuning	✓	✓	✗	**0.73±0.16**	**11.2±23.0**	0.36±0.41	**0.75±0.22**	**0.65±0.14**
Self-Training	✓	✗	✓	0.64±0.20	17.8±28.8	0.51±0.68	0.51±0.27	0.50±0.23
STRUDEL	✓	✗	✓	0.69±0.18	**11.2±14.5**	**0.30±0.32**	0.58±0.27	0.64±0.22
OctSE-Net								
Base Model	✓	✗	✗	0.60±0.23	19.7±29.5	0.77±1.12	0.80±0.26	0.61±0.19
Joint Model	✓	✓	✗	0.73±0.15	11.8±24.7	0.34±0.37	**0.89±0.10**	0.59±0.14
Fine-Tuning	✓	✓	✗	0.73±0.15	11.4±23.4	0.41±0.38	0.77±0.18	0.64±0.17
Self-Training	✓	✗	✓	0.73±0.13	14.7±18.2	**0.25±0.27**	0.56±0.21	0.63±0.17
STRUDEL	✓	✗	✓	**0.78±0.10**	**7.79±8.52**	0.27±0.23	0.77±0.16	**0.70±0.15**
↪ w/o LPA	✓	✗	✓	0.67±0.20	12.9±13.4	0.63±0.58	0.58±0.23	0.66±0.18

dataset delivers underwhelming results, regardless of the backbone architecture employed. This highlights the extent of the domain shift between the source and target domain and the struggle both network architectures face in dealing with this shift.

In agreement with the findings documented in [165], LPA significantly outperforms the baseline U-Net. Furthermore, OctSE-Net surpasses LPA, lending credence to our hypothesis that OctSE-Net constitutes a more robust architecture.

Nonetheless, a closer look at the results in Figure 6.3 reveals that both base models generate some predictions with zero DSC, a situation not encountered with LPA. These outliers could result in inadequately initialized pseudo-labels, underscoring the rationale for including LPA in pseudo-label initialization, as substantiated by the subpar outcomes for STRUDEL without LPA. Furthermore, both baseline methods demonstrate high variance, whereas LPA appears more robust.

In our experiment, we also trained a joint model on source and target data, using the target dataset's ground truth labels instead of pseudo-labels. This model serves as a benchmark, justifying the fine-tuning and self-training approaches. As observable, fine-tuning outperforms the joint training approach for the U-Net architecture. All self-training-based experiments ceased to show further improvements after five iterations, at which point results were reported. Figure 6.4 displays the performance over various iterations for both base architectures. Moreover, we also experimented with a simple fine-tuning model that uses target ground truth labels. This approach led to substantial improvements across all metrics. Standard self-training results are also presented, excluding the integration of LPA or our proposed uncertainty guidance. For U-Net, this self-training matches the performance of the joint model but falls short when

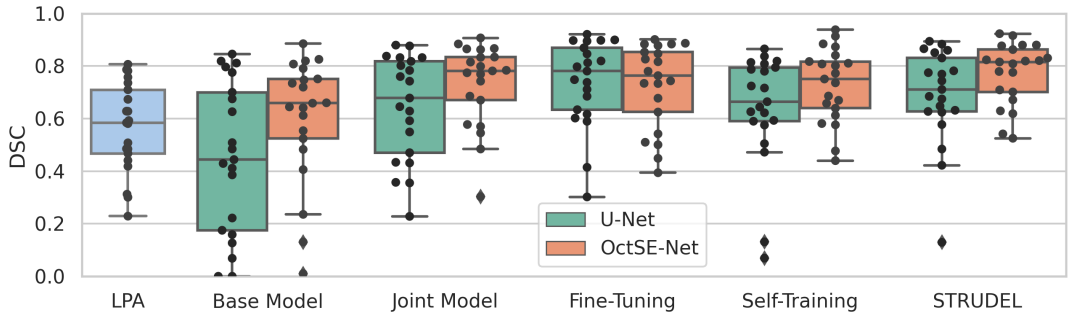

Figure 6.3. Boxplot illustrating the Dice Similarity Coefficient for various methods outlined in Table **??**. Points that fall beyond the whiskers, represented by a diamond shape, are classified as outliers based on the interquartile range. © Springer, 2021

compared to fine-tuning. However, for OctSE-Net, the self-training performance is competitive with the joint model and fine-tuning.

When considering DSC and H95 metrics, STRUDEL outperforms all other methods, holding either the first or second position for lAVD and lesion F1. While LPA and the joint model perform impressively in lesion recall, their performance in lesion F1 suffers due to a high frequency of false positive predictions. STRUDEL, on the other hand, displays robust performance across these metrics, a factor we attribute to uncertainty's capability in effectively capturing false positives. The Wilcoxon signed-rank test results on DSC demonstrate that the improvements made by STRUDEL compared to Self-Training for OctSE-Net ($p < 0.005$), and the superiority of OctSE-Net compared to U-Net for STRUDEL ($p < 0.001$), are statistically significant.

An example segmentation is demonstrated in Figure 6.5, with the corresponding Dice Similarity Coefficients relative to the ground truth being 0.45 for the initial pseudo-label, 0.83 for straightforward self-training, and 0.85 for STRUDEL.

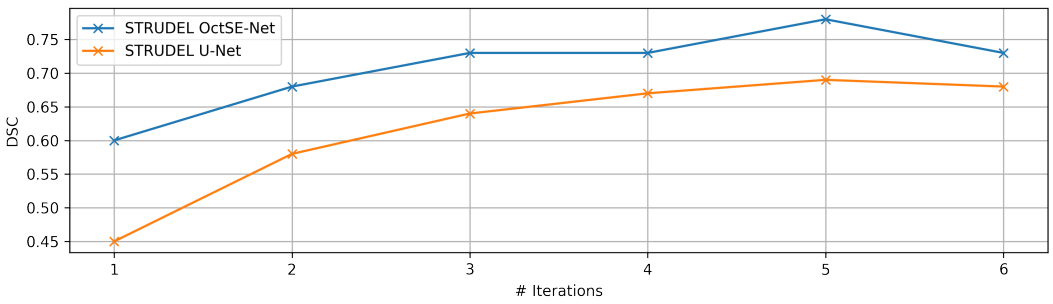

Figure 6.4. Demonstration of the influence of architecture on the behavior of our proposed STRUDEL method across iterations, in terms of the Dice Similarity Coefficient (DSC). Both networks reach a peak after five iterations. © Springer, 2021

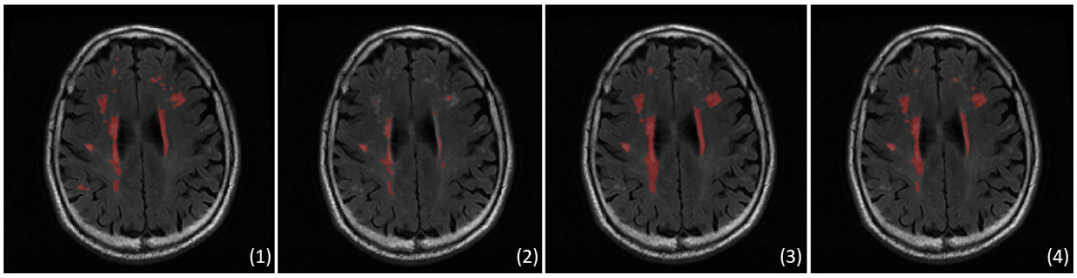

Figure 6.5. A FLAIR scan from the ADNI2 dataset with overlays of segmentation maps. (1) ground truth segmentation and predictions from our networks: (2) initial pseudo-label, (3) standard Self-Training without uncertainty guidance, and (4) our proposed Self-training with Uncertainty Guided Label refinement (STRUDEL). © Springer, 2021

6.4 Conclusion

Self-Training presents a straightforward and resource-efficient approach for UDA; nevertheless, noisy pseudo-labels can hinder its efficacy. In this chapter, we introduced STRUDEL, a novel methodology leveraging uncertainty guidance within self-training to tackle UDA challenges.

STRUDEL incorporates uncertainty into the loss function to direct the learning process amidst noisy labels. Furthermore, integrating an established algorithm, namely the LPA, in the pseudo-label initialization stage added another layer of refinement to the process. Incorporating these elements within STRUDEL minimizes the impact of noisy pseudo-labels. Our empirical evaluations showcased the potential of self-training with uncertainty guidance as a potent mechanism for UDA. Notably, pairing this approach with a robust and high-performing network architecture surpassed even supervised methods, underscoring the promise that STRUDEL holds for future applications. This chapter also mainly concentrated on domain adaptation, where the domain shift lies in the image acquisition by using different scanner manufacturers or imaging protocols. In the next chapter, we will focus on anatomical domain shift, where the domain shift lies in the anatomy of the patients, which we will demonstrate on the example of missing organs after organ resection surgery.

7 Hallucination-Free Organ Segmentation After Organ Resection Surgery

Contents

7.1 Introduction

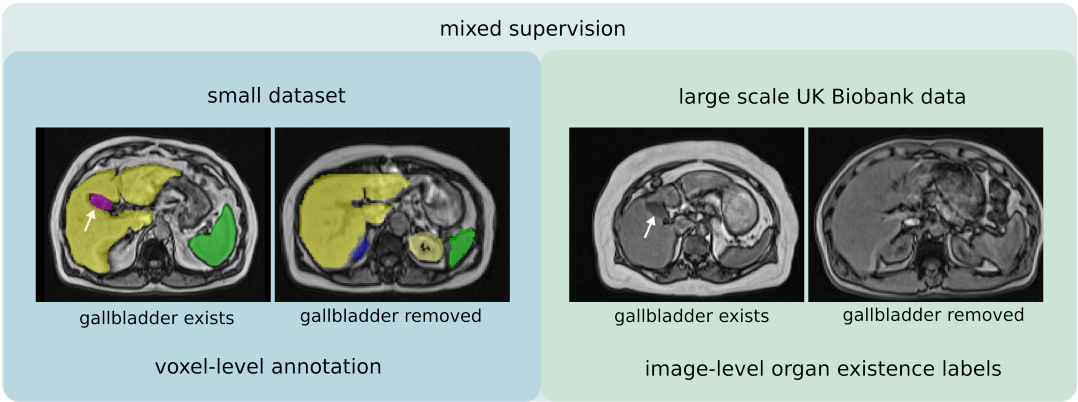

Figure 7.1. In HALOS, we use mixed supervision by combining two datasets: one with image-level binary labels indicating the existence of organs, and another with voxel-level annotations (segmentation maps) for multiple organs. The gallbladder is shown by the white arrow. © Springer, 2023

Deep learning techniques have emerged as the leading approach in medical image segmentation tasks, including the segmentation of brain structures [142], tumors [92], and organs within the abdomen [15, 45, 69, 141]. One of the challenges faced in medical image segmentation is the generalization of models to unfamiliar datasets, which is often hindered by domain discrepancies between the training and testing sets, thereby impacting performance adversely. There has been a wealth of research in the quest for robustness and adaptability in models [52], with many solutions addressing domain shifts that arise from disparities in image intensity distribution due to variations in scanning procedures or modalities. This was examined on the white matter hyperintensity segmentation example in the preceding chapter Chapter 6.

Anatomical domain shifts, particularly those related to the absence of organs due to surgical procedures, have not been as extensively studied. Medical imagery of the human abdomen is generally consistent, containing a set ensemble of organs, unlike natural images that may present arbitrary compositions. This inherent consistency in human anatomy has been advantageous for model training, with the introduction of shape priors [91, 117, 186] to leverage this feature. However, as the field evolves toward clinical applications and extensive population studies, encountering deviations from typical anatomical configurations becomes more probable, affecting segmentation accuracy.

This chapter focuses primarily on gallbladder resection (cholecystectomy), one of the most common abdominal operations. Gallstones are the most common reason for gallbladder removal, as they rarely affect other organs or the body as a whole. Nephrectomy, the surgical removal of the kidney, may be required for more severe conditions such as kidney malignancies, so we expand our examination to include this procedure. In addition, kidneys have a significantly larger volume than gallbladders, and their removal can result in organ shifts after surgery [158].

As will be evidenced by our experimental results, a common pitfall of contemporary segmentation networks is their tendency to falsely detect organs in images where they have been surgically removed - a phenomenon we term *organ hallucination*. Organ hallucinations have likely been overlooked due to the scarcity of organ resection cases in publicly available segmentation datasets. The reason behind the limited representation of organ resection cases in most segmentation datasets is likely the combination of the time-consuming and costly process of manual segmentation, which results in smaller datasets.

Fortunately, the horizon is changing with the introduction of grand-scale population imaging studies, such as the UK Biobank (UKB) Imaging study [90], which sets out to include 100,000 individuals. Such large-scale studies present more representative data of the population. Notably, within our subset of the UK Biobank data, cholecystectomies (gallbladder resections) have a prevalence rate of 3.7%, supplying enough data to thoroughly explore this area of research.

We introduce a novel model named HALOS, which stands for Hallucination-free Organ Segmentation, tailored explicitly for organ segmentation in post-surgical scenarios where organs may have been resected. At the core of HALOS is the simultaneous learning of two tasks – classifying the presence of organs and segmenting six abdominal organs: the liver, spleen, left and right kidneys, pancreas, and gallbladder. Our training approach employs a mixed

supervision strategy to address the constraints of the available data; we utilize voxel-level annotations for a limited dataset alongside image-level labels indicating organ removal in a larger dataset, as illustrated in Figure 7.1. What sets HALOS apart is a flexible feature fusion module [175], which integrates either the classification output or, when accessible, ground truth labels denoting organ existence into the segmentation stream.

This work delivers three critical contributions: Firstly, it introduces a versatile and robust multi-task model that combines the tasks of multi-organ segmentation and organ existence classification. On the large UKB dataset, our approach generates almost no false positives. Second, the classification results are fed into the segmentation pipeline through a unique feature fusion technique called dynamic affine feature map transform, that we for the first time employ in a multi-scale fashion. The severity of the organ hallucination problem is emphasized in this work by comparing ours results to those of state-of-the-art segmentation models.

7.1.1 Related Work

7.1.2 Abdominal Multi-Organ Segmentation

Currently, convolutional neural networks are the leading approach for abdominal organ segmentation in Computed Tomography (CT) and Magnetic Resonance Imaging (MRI) scans [9, 15, 45, 141, 171]. One noteworthy method is nnU-Net [69]. nnU-Net automates the majority of the tasks involved in developing segmentation models. It achieves this by employing flexible 2D, 3D, and ensemble U-Net architectures and automated pre-processing and training processes. The network configuration is dynamically adapted based on the data, making it capable of handling different medical imaging modalities and tasks. nnU-Net has won multiple medical image segmentation challenges and has proven to be a reliable and versatile method. As a result, we use nnU-Net as a baseline in our experiments.

7.1.3 Missing Organ Segmentation

To our knowledge, the missing organ problem has only been investigated for CT scans in [158]. The authors introduce an automatic missing organ detection method and an atlas-guided multi-organ segmentation approach that accounts for missing organs. The missing organ detection method uses features based on post-surgical organ motion and intensity homogeneity to detect the absence of organs. The method was evaluated on cases after kidney and spleen removal on a CT dataset of 44 scans, where 9 scans had missing organs. This approach relies heavily on simulation for parameter tuning and is thus susceptible to distribution shifts. A more recent method [160] investigated the Dice loss, commonly used for training deep learning segmentation models. The authors argue that setting the reduction dimension across the entire batch would help predict images with missing organs. Nonetheless, this method was not tested on cases after organ resection. We compare our approach to this method in our experiments.

7.1.4 Classification-Assisted Segmentation

Since image-level labels are more accessible to acquire than voxel-wise annotations, previous work has explored incorporating these additional labels. A common approach is extending the segmentation network with a classification branch [104, 110, 177]. In [110], both branches are trained jointly with fully-annotated and partially-annotated data, initially sharing layers for the segmentation of 2D brain tumors and classification of tumor existence. They demonstrated that the additional classification improved segmentation performance significantly compared to conventional fully supervised learning. In our experiments, we compare our approach to this methodology.

7.1.5 Feature Fusion

Feature fusion is a concept that certain classification-assisted segmentation approaches adopt for intertwining the segmentation and classification paths. For instance, in [177], dedicated segmentation and classification models are developed to diagnose Covid-19. The feature maps originating from the classification and segmentation models are merged utilizing Squeeze-and-Excitation (SE) blocks [63]. Once the feature maps are fused, they are passed to the decoder for segmentation. Another approach, apart from feature fusion, is incorporating metadata like age, gender, or specific biomarker readings. The Dynamic Affine Feature Map Transform (DAFT) [175] computes the scaling factors and shifts employed to excite or suppress the feature maps at the channel level based on the metadata mentioned earlier. This process is depicted inFigure 7.2.

7.2 Methods

Figure 7.2 visually represents HALOS' dual-branch architecture, which combines Multi-task Learning with Feature Fusion for effectively managing cases with missing organs. We will now delve into the specifics of each component within this pipeline.

7.2.1 Segmentation Branch

We use a U-Net architecture derived from nnU-Net [68] in the segmentation branch. NnU-Net is widely regarded as one of the most adaptable and powerful segmentation models for medical images. The nnU-Net pipeline calculates the best U-Net architecture and data augmentation method for a given dataset automatically. This led us to feed our segmentation dataset into the nnU-Net pipeline, where we could pick the architecture and data augmentation technique of the nnU-Net model that performed the best. The final model is a 3D U-Net with 32 channels and five downsampling levels. This structure includes an encoder $\mathcal{E}$, a bottleneck $\mathcal{B}$, and a decoder $\mathcal{D}$.

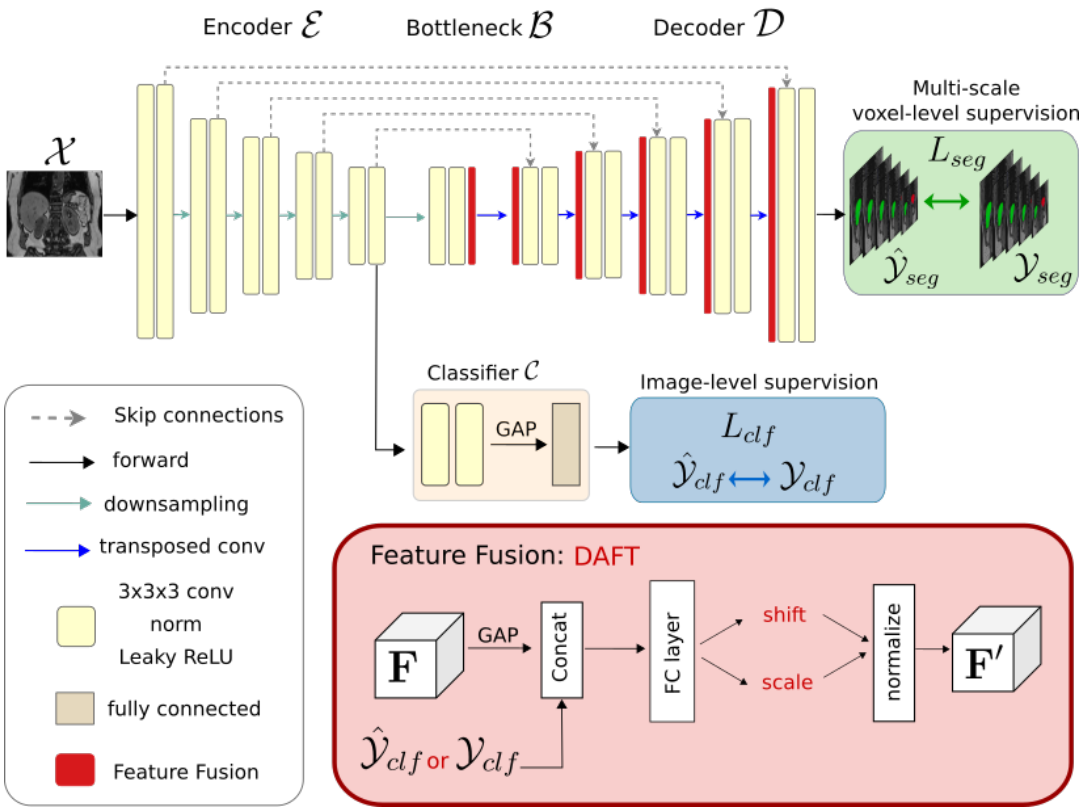

Figure 7.2. Overview of the HALOS pipeline. The U-Net network is extended with a classification branch that classifies whether the image contains the organ of interest. Further, our network is trained in multi-scale voxel-level supervision, also called deep supervision. The DAFT feature fusion modules are added to the bottleneck and each decoder block. © Springer, 2023

Encoder-decoder structured networks offer the ability to derive intermediate representations at different resolutions due to their consecutive downsampling steps. The U-Net is trained on $\mathcal{X} = \{\mathbf{X}_i\}_{i=1}^{N}$ input MR images and ground truth segmentation maps $\mathcal{Y}_{seg} = \{\mathbf{Y}_{seg_i}\}_{i=1}^{N}$ to generate segmentation predictions. The predictions $\hat{\mathcal{Y}}_{seg} = \{\hat{\mathbf{Y}}_{seg_i}\}_{i=1}^{N}$ are learned under voxel-level supervision, optimizing the segmentation loss, $\mathcal{L}_{seg}$. The loss is defined as the average of Dice and Cross-Entropy loss. As suggested by the nnU-Net pipeline, we additionally compute the loss on segmentation predictions from intermediate feature maps from the decoder at lower resolution, which is commonly known as deep supervision. Formally, the segmentation loss is defined as:

$$\begin{aligned} \mathcal{L}_{seg} &= \mathcal{L}_{CE} + \mathcal{L}_{Dice}, \qquad \mathcal{L}_{CE} = -\frac{1}{N}\sum_{n}^{N}\sum_{c}^{C}\mathbf{Y}_{seg_c}\log(\hat{\mathbf{Y}}_{seg_c}), \\ \mathcal{L}_{Dice} &= \frac{1}{N}\cdot\left(1 - \frac{2\cdot\sum_{n}^{N}|\hat{\mathbf{Y}}_{seg_n}\cap\mathbf{Y}_{seg_n}| + \epsilon}{\sum_{n}^{N}|\hat{\mathbf{Y}}_{seg_n}| + |\mathbf{Y}_{seg_n}| + \epsilon}\right). \end{aligned} \tag{7.1}$$

In this equation, we denote the class-wise ground truth $\mathbf{Y}_{seg_c}$, class-wise predictions $\hat{\mathbf{Y}}_{seg_c}$, the number of classes C, the number of segmentation samples N, and a smoothing term ϵ. It is crucial to note that certain implementations of the Dice loss only add ϵ to the denominator to avoid division by zero. In our situation, adding ϵ to the numerator and denominator is essential since we want to ensure a Dice loss of 0, rather than 1, for accurate negative predictions of gallbladders. Further, this definition of the Dice loss already includes batch reduction, meaning the nominator and denominator are summed over the batch dimension before the division.

7.2.2 Classification Branch

Gathering global labels at the image level is remarkably more economical than collecting manual annotations at the voxel level, albeit the trade-off is in the level of detail captured. Consequently, we introduce a classification module into our framework to investigate the impact of incorporating prior knowledge of organ existence on the final segmented outputs. The classifier (designated as $\mathcal{C}$) is integrated into the U-Net model above the encoder ($\mathcal{E}$), and it processes a feature map that is sourced from a particular encoder block. Although the placement of the classifier can be adjusted through hyperparameters, our experiments indicated that incorporating it into the fourth or fifth block of the encoder yielded the most effective results.

$\mathcal{C}$ is structured with a convolutional block, the design of which mirrors that of an encoder block. This is succeeded by a 3D global average pooling phase and, ultimately, a densely connected layer that yields the classification results. The classifier is fine-tuned using MR scans labeled with image-level surgery data, symbolized as $\mathcal{Y}_{clf}$. We compute the classification loss, referred to as $\mathcal{L}_{clf}$, using the mean cross-entropy, and it is balanced by considering the true distribution of classes within the dataset used for training.

7.2.3 Feature Fusion

A core component of HALOS is the feature fusion module. We posit that merging the classifier's information back into the segmentation branch, instead of solely having a multi-task model, arms the model with the necessary information about organ presence during segmentation. The module fuses prior knowledge of gallbladder removal with feature maps at distinct points along the segmentation branch, including the bottleneck and all decoder stages, as illustrated in Figure 7.2. It's worth mentioning that either ground truth labels ($\mathcal{Y}_{clf}$) or the classifier's predictions ($\hat{\mathcal{Y}}_{clf}$) could be fed into the feature fusion module, depending on whether data on prior surgeries is accessible during testing. While training, we predominantly rely on ground truth labels for feature fusion. Although employing the classifier's predictions during training is intriguing, it warrants additional exploration.

We adopt DAFT [175] for the fusion process. Originally designed to integrate 3D images with sparse tabular data, DAFT can be effortlessly incorporated into any Convolutional Neural Network (CNN). In preliminary experiments, we experimented with Squeeze-and-Excitation (SE) blocks [62] as an alternative, but they did not work as well. In our setup, the fused tabular data comprises either the binary classification output or the ground truth label of gallbladder removal. As far as we know, this is the first time DAFT has been used in segmentation models or across multiple scales. We anticipate that sharing information at multiple scales of the decoder will emphasize the prior knowledge about the organ's existence and lead the decoder to generate fewer false positive predictions for non-existing classes.

Figure 7.2 delineates the precise locations of the feature fusion modules within the U-Net structure. We merge the classification labels with the bottleneck feature map with the most abstract information. After each transpose convolution, this merged version is fed into the decoder, where we iterate the feature fusion blocks. We avert interference with additional normalization layers by placing the DAFT-based feature fusion ahead of each decoder block.

To put it in mathematical terms, for every element in a batch, let's denote the classifier's predicted output as $\hat{y} \in \mathbb{R}$, and $\mathbf{F}_{d,c} \in \mathbb{R}^{D \times H \times W}$ as the input feature map's c-th channel for block $d \in 0, \ldots, 5$ in the decoder, where D, H, W represent the depth, height, and width of the feature map respectively, as depicted in Figure 7.2.

DAFT [175] is trained to ascertain scale $\alpha_{d,c}$ and offset $\beta_{d,c}$, and can be formally expressed as:

$$\mathbf{F}'_{d,c} = \alpha_{d,c}\mathbf{F}_{d,c} + \beta_{d,c}, \tag{7.2}$$

$$\alpha_{d,c} = f_c(\mathbf{F}_{d,c}, \hat{y}_d), \qquad \beta_{d,c} = g_c(\mathbf{F}_{d,c}, \hat{y}_d), \tag{7.3}$$

Here, f_c and g_c denote arbitrary functions that map from the image and tabular space to a scalar. In line with [175], a singular fully-connected neural network, h_c, embodies f_c and g_c and yields one pair of α and β values.

MR images with voxel-level and image-level labels are randomly selected throughout the training phase to compose batches. These batches are then utilized to update the segmentation model and the classifier. Utilizing the previously outlined $\mathcal{L}_{seg}$ and $\mathcal{L}_{clf}$, HALOS's final loss is computed as:

$$\mathcal{L} = \alpha \cdot \mathcal{L}_{seg} + (1 - \alpha) \cdot \mathcal{L}_{clf}, \tag{7.4}$$

In this equation, α denotes the weight attributed to the segmentation loss.

7.3 Results and Discussion

7.3.1 Experiment Setup

Segmentation Data

In this study, we make use of whole-body MRI scans that have been annotated at the voxel level, and these scans are sourced from three different studies: the German National Cohort (NAKO) [7], the Cooperative Health Research in the Region of Augsburg (KORA) [6], and UKB [90]. These datasets represent a diverse population drawn from Germany and the United Kingdom. All three studies employed a two-point Dixon sequence to capture abdominal images, and for our research, we specifically use the oppose-phase scans. Our pre-processing approach aligns with the recommendations made in earlier studies [73, 136]. It is important to note that the whole-body MRI scans are an assembly of multiple scans, each focusing on different body parts, ranging from the neck down to the toes. These scans have fields of view that overlap. The separate scans are stitched together during pre-processing to create a singular volume. Additionally, we extract a specific region of interest that encompasses the abdominal organs under investigation. The scans are then resampled to achieve a uniform resolution of $2 \times 2 \times 3$ mm^3. We also employ the N4 bias field correction technique [163] as part of the pre-processing. A medical expert conducted manual segmentation to identify the six organs of interest to annotate the scans. The dataset consists of a total of 63 scans, which include 16 from NAKO, 15 from KORA, and 32 from UKB. Within this dataset, 18 patients had undergone gallbladder removal. We partitioned this dataset into three subsets: training (42 scans, 9 of which are post-gallbladder resection), validation (7 scans, 3 of which are post-gallbladder resection), and testing (11 scans, 6 of which are post-gallbladder resection).

UKB Data

The dataset from the UK Biobank is substantially more extensive than the segmentation data, but it solely comprises image-level annotations, which indicate the presence of organs. We harness this dataset for training the organ existence classifier, an integral component of our multi-task pipeline. Moreover, it proves useful for assessing the robustness of the model, as

we can compute the instances of false-positive segmentation for gallbladders that are not present. We sourced information regarding past surgeries from the UKB database. As the information about previous surgeries is self-reported by the study participants, our medical expert ensured the accuracy of the labels. Out of the 19,000 images that we obtained from the UK Biobank, 701 were identified as post-gallbladder removal cases. We also made a random selection of subjects with no specific conditions. We divided this data into two groups. The first group, meant for training and validation of the models, contained 899 scans of individuals with gallbladders and 349 of those without. The second group, an independent test set, included 952 scans with gallbladders and 352 without. Notably, the ratio of cases without a gallbladder in each group is approximately 0.4. The UKB data underwent the same pre-processing procedures as described earlier.

Implementation Details and Hyperparameter Tuning

Our research uses DGX A100 GPUs as the computational backbone for experiments. Our code is based on Python, and we use PyTorch and MONAI as the key libraries. The tuning of hyperparameters such as the loss weight α, weight decay, learning rates for both the segmentation model and classifier, the type of normalization (be it instance or batch normalization), batch size, and the location of the classifier is carried out with the aid of Ray Tune. We leverage PyTorch's automated mixed precision feature for training our models. The source code is openly accessible and can be found at the following link: `https://github.com/ai-med/HALOS`.

Metrics

Our goal in this work is to provide accurate and robust segmentations for patients with and without removed gallbladders. Therefore, we evaluate the performance with a standard segmentation metric, the Dice Score, to evaluate the overall performance. As our models are based on nnU-Net, which already provides state-of-the-art segmentation, we are mainly interested in performance decrease due to organ resection. In cases where an organ was removed, the Dice score is not defined for this organ. Therefore we set it to 1 for true negative cases and 0 for false positive cases by setting ϵ to 1. As a result, we can observe significant changes in the Dice score when reducing the false positive rate. The main metric we are interested in regarding organ hallucinations is the false positive rate (FPR) of resected organs. We count an image as a false positive if one or more voxels were segmented as a non-existing organ. Further, on the large-scale UKB data, we provide other metrics derived from the confusion matrix, such as false negatives and F1 scores. To evaluate the performance of the organ existence classifier, we provide balanced accuracy (BAcc).

Baselines

In addition to the nnU-Net baseline outlined in subsection 7.2.1, we also evaluate two simple baseline methods to tackle the problem of organ hallucinations, which we term oversampling and post-processing. Given the data imbalance concerning the number of samples with intact versus resected organs, we believe this is a potential reason for organ hallucinations. Therefore for the oversampling baseline, we oversample cases without a gallbladder during training to achieve a balance in class frequency. It is imperative to acknowledge that class frequencies

Table 7.1. This table showcases a comparison between HALOS and the baseline nnU-Net, as well as some straightforward baselines involving oversampling and post-processing, in addition to techniques from related studies such as Dice loss with batch reduction[160] (labeled as + batch red.) and the multi-task model [110].FF indicates feature fusion, 'pred' indicates the use of predicted organ existence labels during testing, and 'gt' indicates the use of ground truth labels during testing. The table exhibits Dice scores for all organs in addition to the false positive rate (FPR) for cases following cholecystectomy. For both metrics, the mean and standard deviation derived from a 5-fold cross-validation are presented. The asterisk (*) signifies that the optimal 3D U-Net architecture for our dataset, as recommended by the nnU-Net pipeline, was reimplemented.

	Dice Scores ↑							FPR ↓
Method	Mean	liver	spleen	r kidney	l kidney	pancreas	gallbl.	
nnU-Net*[68]	0.823±0.014	0.938±0.004	0.891±0.006	0.898±0.003	0.894±0.002	0.643±0.016	0.674±0.076	0.267±0.149
+ oversampling	0.832±0.008	0.940±0.006	0.894±0.005	0.901±0.005	0.891±0.005	0.655±0.011	0.712±0.052	0.233±0.091
+ post-proc. (gt)	0.847±0.005	0.938±0.004	0.891±0.006	0.898±0.003	0.894±0.002	0.643±0.016	0.819±0.009	0±0
+ batch red. [160]	0.818±0.010	0.945±0.002	0.895±0.002	0.901±0.005	0.894±0.006	0.663±0.014	0.610±0.045	0.400±0.091
multi-task [110]	0.822±0.010	0.930±0.006	0.879±0.004	0.895±0.003	0.885±0.002	0.625±0.016	0.716±0.054	0.233±0.091
HALOS w/o FF	0.825±0.010	0.941±0.002	0.892±0.009	0.898±0.004	0.892±0.005	0.657±0.013	0.668±0.073	0.3±0.139
HALOS (pred, gt)	0.853±0.002	0.939±0.003	0.899±0.005	0.899±0.003	0.893±0.004	0.649±0.021	0.840±0.015	0±0

are already factored into the loss function computation. Conversely, in scenarios where it is definitively known at the testing phase whether an organ has been excised, an intuitive solution is the elimination of any false positive detections through post-processing. This is accomplished by reassigning such pixels to the background class. The post-processing approach capitalizes on pre-existing knowledge of gallbladder resections to remove false positives. However, this strategy is contingent upon the availability of this information and is rendered infeasible when such data is absent.

7.3.2 Experiments on Cholecystectomy Cases

We train all models using 5-fold cross-validation and report the average results over all folds on the segmentation test set in Table 7.1 and the UKB test set in Table 7.2. First, we want to understand the severity of the organ hallucination problem and evaluate the baseline model and several simple approaches for reducing false positives. The average FPR is relatively high for the baseline nnU-Net on both datasets, around 0.26. This also shows that our small segmentation dataset seems to be representative of the population in UKB. The high FPR results in a relatively low gallbladder Dice score of 0.674. Remember that cases with false positives are counted as a Dice score of zero, therefore having a high impact on the Dice score. The segmentation performance on the pancreas is also low at 0.643, but this organ is difficult to segment due to its shape variability. We can see that a well-performing segmentation model like nnU-Net leads to false positives in around a quarter of cases. Next, we want to evaluate whether simple approaches can alleviate the problem of organ hallucinations. As the amount of gallbladder resection cases in our training segmentation data is around 24% and in the UKB data 40%, a first observation is the class imbalance and a simple oversampling of gallbladder resection cases during the training process might help to reduce false positives. We observe that the oversampling only marginally improves the false positive rate on both datasets, indicating that the high FPR is not solely caused by class imbalance. Further, on the

Table 7.2. Comparison of HALOS and the baseline nnU-Net, as well as basic baseline approaches such as oversampling and post-processing, and techniques from related research, such as Dice loss with batch reduction [160] (abbreviated as + batch red.) and the multi-task model [110], on the UKB dataset. FF here stands for feature fusion; gt for feature fusion using ground truth labels at test time; and pred for feature fusion using classification predictions at test time. The table provides the balanced accuracy (BAcc) of all classifiers, as well as the false positive (FP), false negative (FN), true positive (TP), true negative (TN), false positive rate (FPR), and F1 score for instances with excised gallbladders. The displayed values are the mean and standard deviation derived from 5-fold cross-validation.
The replication of the nnU-Net pipeline's suggested 3D U-Net architecture for our dataset is denoted by an asterisk (*).

Method	FP ↓	TN ↑	TP ↑	FN ↓	FPR ↓	F1 ↑	BAcc ↑
nnU-Net*[68]	91.2 ±30.62	260.8 ±30.62	537.2 ±16.62	62.8 ±16.62	0.259 ±0.087	0.875 ±0.009	
+ oversampling	66.6 ±6.633	285.4 ±6.633	522.6 ±13.18	77.4 ±13.18	0.189 ±0.028	0.879 ±0.011	
+ post-proc. (gt)	0 ±0	352 ±0	537.2 ±16.62	62.8 ±16.62	0 ±0	0.945 ±0.015	
+ batch red. [160]	135.2 ±57.15	216.8 ±57.15	530.2 ±24.39	69.8 ±24.39	0.384 ±0.162	0.838 ±0.017	
multi-task [110]	100.2 ±16.48	251.8 ±16.48	578.2 ±3.701	21.8 ±3.701	0.285 ±0.047	0.905 ±0.054	0.874 ±0.045
HALOS w/o FF	52.6 ±17.67	299.4 ±17.67	547.4 ±22.39	52.6 ±22.39	0.149 ±0.050	0.869 ±0.056	0.896 ±0.047
HALOS (gt)	2 ±2.550	350 ±2.550	564.8 ±14.74	35.2 ±14.74	0.006 ±0.007	0.968 ±0.010	0.933 ±0.005
HALOS (pred)	11 ±5.339	341 ±5.339	541.8 ±14.20	58.2 ±14.20	0.031 ±0.015	0.940 ±0.010	0.933 ±0.005

UKB data, we observe that oversampling reduces false positives and increases false negatives, which is unwanted behavior.

Using ground-truth information about cholecystectomy, post-processing predictably results in higher gallbladder Dice scores and zero FPR. However, the post-processing approach has a drawback: the model's false positive predictions may arise in neighboring organs, creating a segmentation hole. Gallbladders are typically located in the fossa vesicae biliaris, a depression on the visceral surface of the liver between the quadrate and right lobes. Due to this area's proximity to the liver, we observed numerous errors in our baseline that are either localized within the liver or partially within the liver and other tissues, such as visceral fat. In Figure 7.3, the gallbladder is hallucinated in the fossa vesicae biliaris (C), the liver (D), and the intestine (E) as examples of common organ hallucinations. We want to point out that there is no change to be observed in Dice scores of other organs like the liver, as the liver is a large organ, and small holes from removing false positive gallbladder segmentations did not have an effect on the Dice score on our small segmentation dataset.

The recent work [160] suggests lowering ϵ in the Dice loss to a low number, e.g., 10^{-7}, increasing the batch size, and reducing the Dice loss across the batch dimension. The authors argue that this will help with missing organ cases but have not evaluated their method on a dataset with resected organs. To test their method, we set the batch size to 8, which was the maximum amount of GPU RAM we had. Our baseline model has a batch size of 2, an epsilon of 1, and also reduces across the batch dimension. Surprisingly, when using the method by [160], denoted as '+ batch red' in Table 7.1 and Table 7.2, we observe an increase in FPR for both datasets. We removed the batch reduction in the Dice loss in preliminary experiments but found no noticeable difference in performance.

Next, we look at the effect of multi-task learning, where the idea is that learning shared features between segmentation and classification branches will lead to better performance of

the segmentation network. The multi-task model proposed in [110] incorporates a classifier just before the decoder's segmentation output. To test this approach, we employ our nnU-Net model and enhance it with a classifier, as recommended in [110] at decoder block 5. This model results in a slightly lower FPR on segmentation data but a higher FPR on UKB data. Furthermore, we train Hallucination-free Organ Segmentation (HALOS) without feature fusion, which makes it a simple multi-task model and differs from the approach by [110] only in the location of the classifier. This leads to unexpected gain in FPR, a decrease in gallbladder Dice score, and a minor loss in FPR on UKB compared to nnU-Net, despite the classifier's balanced accuracy being 0.896. As a result, we contend that multi-tasking training alone cannot diminish organ hallucinations.

We trained HALOS using the ground truth labels $\mathcal{Y}_{clf}$ as input for the DAFT modules. In preliminary experiments, we have also tried using the classifier's predictions $\hat{\mathcal{Y}}_{clf}$ as input but did not achieve comparable results. In future work, this could be interesting to explore further. HALOS achieves a reduction of FPR to 0 on the segmentation dataset. On the UKB data, it is slightly increased but still near zero at 0.006. The classifier attains a balanced accuracy of 0.93. When we use the classifier's prediction for feature fusion at test time, we observe a slight increase in FPR to 0.03 over using the ground truth labels on the UKB data. As the FPR on the segmentation data is zero, there is no difference between the classification output and the ground truth labels. This demonstrates the adaptability of our strategy; depending on whether prior information regarding gallbladder resection is available at the time of testing, the ground truth labels or classifier predictions can be used for feature fusion.

7.3.3 Experiments on Nephrectomy Cases

Next, we evaluate the capability of HALOS to tackle cases involving organ removal beyond gallbladder resections by examining its performance in post-nephrectomy instances. Acknowledging that no extra fine-tuning of hyperparameters was executed for this assessment is crucial. We compiled a dataset with 46 scans (6 missing left kidneys and 2 missing right kidneys) for training purposes and 10 scans (2 missing left kidneys and 1 missing right kidney) for testing. In the case of the UKB dataset, we partitioned the data into a training and validation set comprising 200 scans (17 missing left kidneys and 5 missing right kidneys), along with a separate test set of 55 scans with 4 and 2 instances of missing left and right kidneys, respectively.

Like the gallbladder analysis, we trained the baseline nnU-Net and HALOS models using 5-fold cross-validation. In contrast to the binary classification used in the gallbladder experiments, the classifier in this example was tasked with classifying whether all kidneys are intact or whether the left or right kidneys are missing. Below are the results of the post-nephrectomy analysis.

In the UKB dataset, the baseline nnU-Net registered a False Positive Rate of 0.2 for the left kidney and a perfect 1 for the right kidney. Comparatively, HALOS demonstrated remarkable improvement with an FPR of 0 for the left kidney, and though it had a high FPR of 0.7 for the right kidney, the count of false positive voxels was considerably diminished to 16.5 from nnU-Net's 129. Additionally, HALOS exhibited a superior Dice score for the left kidney (0.9024)

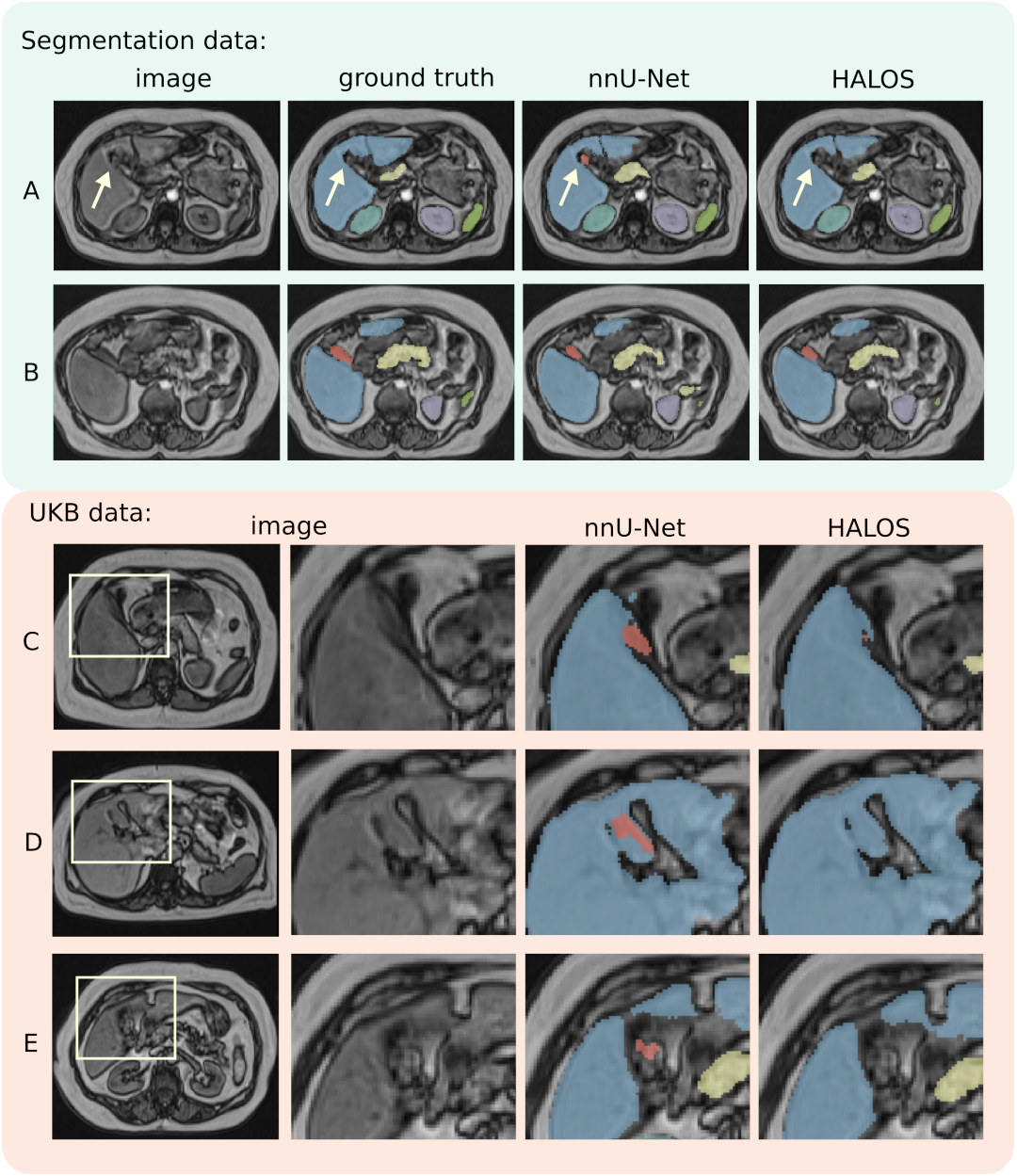

Figure 7.3. Visual representation of segmentation outcomes on both the segmentation dataset (upper row) and UKB dataset (lower row), featuring a side-by-side comparison between nnU-Net and HALOS. A: Illustrates a scan where the gallbladder has been removed; nnU-Net generates a false positive, while HALOS successfully avoids false positives. Scan B demonstrates an intact gallbladder. The gallbladder is well defined by both nnU-Net and HALOS. C: In this case, the gallbladder's former position is wrongly predicted as a false positive by both models. However, HALOS has a significantly reduced number of falsely predicted voxels. D: In this case, nnU-Net incorrectly identifies a region within the liver as the gallbladder (false positive). HALOS accurately segments the liver without any false positives in the corresponding area. E: Displays a scenario where nnU-Net wrongly identifies a section of the intestine as the gallbladder (false positive), whereas HALOS does not make this error and produces no false positives.

compared to nnU-Net (0.8413). However, the right kidney's Dice scores were nearly identical for both models (0.864 for HALOS versus 0.867 for nnU-Net).

One plausible explanation for these results is the constraints imposed by the limited size of the dataset coupled with a notable class imbalance, particularly given that the training set contained only two instances of right kidney removal. The balanced accuracy for the HALOS classifier was 0.93 for the left kidney but substantially lower at 0.58 for the right kidney, reinforcing the supposition that the class imbalance plays a more prominent role in this scenario.

7.4 Conclusion

In this chapter, we introduced HALOS, a versatile multi-task model that combines classification and segmentation for more accurate and hallucination-free organ segmentation. We incorporated multi-scale feature fusion through a dynamic affine feature-map transform to enrich the segmentation branch's feature maps with prior knowledge regarding the presence of organs. Our experiments demonstrate that HALOS substantially reduces the number of false positives on a comprehensive UK Biobank testing set and enhances the Dice scores for the gallbladder and left kidney on a more compact segmentation testing set when compared to nnU-Net and an assortment of baselines and multi-task approaches, particularly for scenarios following cholecystectomy and nephrectomy.

A central advantage of HALOS is its adaptability. Depending on the information available during testing, HALOS can effortlessly switch between employing organ existence ground truth labels and the predictions made by its classifier. This adaptability ensures that HALOS can effectively perform under different data scenarios without compromising its efficacy.

As we move forward, our ambition is to broaden the scope of HALOS to encompass more types of organ resections. Potential future application areas include cases of hysterectomy, where the uterus is removed surgically, and splenectomy, which entails the spleen's removal. Since datasets that include post-removal cases of organs like the kidney or spleen are typically small, we believe that creating synthetic data could be a worthwhile approach to tackle this challenge. Additionally, employing self-supervised pre-training on extensive datasets might also prove to be advantageous.

8 Discussion of Voxel-Based Segmentation Methods

8.1 Summary

In this part, I have discussed three voxel-based medical image segmentation methods. Firstly, we presented Project & Excite (PE) modules, a novel approach to recalibrating 3D Fully-Convolutional Neural Networks (F-CNNs). Inspired by the Squeeze and Excite modules used for 2D channel recalibration, we designed our PE modules specifically to address the challenges faced with 3D networks. Our research revealed that our recalibration method outperformed other 3D extensions and proved particularly beneficial for segmenting smaller structures, a common challenge in the field.

Our second work introduced Self-training with Uncertainty Guided Label refinement (STRUDEL), a method developed for self-training with uncertainty-dependent label refinement. Our focus here was on the task of white matter hyperintensity segmentation. The methodology of STRUDEL was particularly geared toward cases where the available data set consisted of a small amount of labeled data and a large unlabelled data set. Using Bayesian uncertainty measures to guide the self-training process proved innovative and efficient, leading to notable improvements in the segmentation model's performance. STRUDEL demonstrated significant potential for the task of domain adaptation and managed to even surpass supervised methods in some cases.

Lastly, we addressed the often-overlooked challenge of anatomical domain shifts due to organ resection. Our solution, Hallucination-free Organ Segmentation (HALOS), is a flexible multi-task model that concurrently learns organ existence classification and segmentation. By incorporating the dynamic affine feature-map transform (DAFT), our model enhanced the feature maps of the segmentation branch with prior information about organ presence. This significantly reduced false positive predictions on a large-scale UK Biobank test set compared to several baselines and multi-task strategies.

8.2 Discussion

While our PE module has proven effective in medical image segmentation, especially for small structures, it is only one possible approach to developing efficient and accurate 3D segmentation techniques. The recent successes of transformer architectures, which are fundamentally based on attention mechanisms, signal another exciting avenue of research. These architectures have gained popularity in computer vision and have made strides in medical

image segmentation [149]. This is because transformers can capture long-range dependencies without being constrained by the fixed geometric structures of convolutions, potentially offering an advantage in complex segmentation tasks. However, the application of transformer architectures in medical imaging is still relatively new, and there are challenges to address. For example, the high computational cost of transformers can be prohibitive for 3D imaging data. Additionally, while the sequence-based nature of transformers is well-suited for NLP tasks, adapting them for image data, which is fundamentally 2D or 3D, poses challenges.

Currently, there is a trend towards developing architectures that combine the strengths of traditional U-Nets, which have a proven track record in medical image segmentation, with the capabilities of transformer networks [149]. These hybrid architectures could leverage the spatial awareness of U-Nets with the global receptive field of transformers, opening up new possibilities for segmentation tasks.

The idea of refining pseudo-labels using uncertainty has been proposed in several works concurrent to our work, e.g., in [172], which shows the relevance of our idea. Another aspect worthy of attention in future research is the choice of the initial segmentation model used for pseudo-label generation. In our work, we utilized the lesion prediction algorithm (LPA), but there could be potential benefits to exploring other robust algorithms or even combining the outputs from multiple algorithms to generate initial pseudo-labels. This ensemble approach could further reduce label noise and increase the reliability of pseudo-labels, leading to improved performance of the STRUDEL method.

The impact of HALOS is yet to be fully realized at the time of this thesis submission, but the potential for expansion and adaptation of the model is considerable. Future directions include the evaluation of HALOS across a range of organ resection cases, broadening its applicability. Furthermore, the model's dynamic affine feature map transform (DAFT) module offers opportunities to incorporate a broader range of meta-data, such as demographic information (age, sex, BMI) or comorbidity details, which could enrich its prediction capabilities.

Additionally, HALOS isn't confined to organ resection cases. Its flexible design allows it to be applied to other instances of anatomical domain shifts, such as cases featuring severe scoliosis, deformities like horseshoe kidneys, or even patients with extra organs, such as polysplenia. As the field moves toward large-scale epidemiological studies, like the UK Biobank, the role of incorporating meta-data will grow in importance. Recognizing and addressing novel challenges, like the missing organ problem, will be an essential part of future work, ensuring our machine learning methodologies remain effective and relevant in the evolving landscape of medical image segmentation.

The focus of this part of the thesis has been on voxel-based medical image segmentation. This approach is well-suited for a broad range of organs, including subcortical brain structures and abdominal organs. Despite the effectiveness of this methodology, when it comes to the segmentation of the cerebral cortex, surface-based segmentation methods often prove to be more advantageous. The reasons behind this and an in-depth exploration of these surface-based techniques will be the focus of the next part of this thesis, offering a comprehensive perspective on the diverse strategies utilized in medical image segmentation.

Part III

Surface-Based Segmentation

9 Introduction

Apart from subcortical segmentation and lesion segmentation, as discussed in Chapter 5 and Chapter 6, cortical surface reconstruction is another essential field in neuroimage computing. The representation of the cortical surface as a triangular mesh is particularly valuable for analyzing cortical thickness and sulcal morphology. Traditional methods like FreeSurfer [39] include a surface reconstruction pipeline that computes all cortical measurements on a surface representation instead of a voxel representation. Additionally, the cortex can be further subdivided, or segmented into smaller areas, often called parcels, hence the name cortex parcellation.

I will focus on cortical surface reconstruction from brain Magnetic Resonance Imaging (MRI) scans and surface-based parcellation in this part. This part is based on the following publications, all of which are collaborative works. For each publication, the contributions of each author are listed.:

Bongratz, F., Rickmann, A.*, Pölsterl S., and Wachinger, C. Vox2Cortex: fast explicit reconstruction of cortical surfaces from 3D MRI scans with geometric deep neural networks, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, ©IEEE, 2022* [10] *: the authors contributed equally

This work is covered in chapter 11. C. Wachinger and the author conceived the initial idea. F. Bongratz implemented the proposed architecture, and F. Bongratz and the author ran the experiments. F. Bongratz, C. Wachinger, and the author contributed equally to writing the final published article. S. Pölsterl helped with group study experiments and proof-reading the final manuscript.

Rickmann, A., Bongratz, F.*, Pölsterl S., Sarasua, Ignacio, and Wachinger, C. Joint Reconstruction and Parcellation of Cortical Surfaces, Machine Learning in Clinical Neuroimaging: 5th International Workshop, MLCN 2022, Held in Conjunction with MICCAI 2022, Singapore, September 18, 2022, Proceedings, ©Springer Nature, 2022* [132] *: the authors contributed equally

This work is covered in chapter 12. F. Bongratz, C. Wachinger, and the author conceived the initial idea. F. Bongratz did the implementation and execution of experiments on the class-based reconstruction loss, and the author did the implementation and running of experiments on classification networks. F. Bongratz, C. Wachinger, and the author contributed equally to writing the final published article. S. Pölsterl and I. Sarasua helped with group study experiments and proof-reading the final manuscript.

Rickmann, A., Bongratz, F., and Wachinger, C.: Vertex Correspondence in Cortical Surface Reconstruction, accepted at International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI) 2023 This work is covered in chapter 13. C. Wachinger, F. Bongratz, and the author conceived the initial idea. The author did the implementation and execution of experiments. The author wrote a significant portion of the manuscript that all authors took part in writing.

I will discuss the principles behind these methods, their advantages and limitations, and how they compare to existing state-of-the-art techniques in the field. The outline of this part is as follows, first, I will introduce the cortical surface reconstruction problem statement and explain the fundamentals of the field. Then I will explain the concept of Vox2Cortex. Next, I will discuss the extensions of Vox2Cortex, including joint parcellation approaches and the improvement of vertex correspondence. Finally, I will discuss the three presented methods and shed light on some promising avenues for future exploration. By the end of this part, readers will clearly understand the challenges involved in cortical surface reconstruction and surface-based parcellation and the latest approaches for addressing them.

10 Fundamentals for Cortical Surface Reconstruction

Contents

In the development of medical image segmentation, two distinct approaches have emerged – the intensity-based or voxel-based methodologies and the boundary-based or surface-based methodologies. While both approaches have merits and applicabilities, they differ fundamentally in methodology. Voxel-based techniques focus on the voxel grid, independently classifying each voxel based on its features and the surrounding context. In contrast, surface-based methods explicitly model the boundaries of the segmented objects, allowing them to capture geometric and topological properties more naturally.

The specific advantages of surface-based segmentation become evident in neuroimaging, where surface representations are particularly valuable for analyzing cortical thickness and sulcal morphology. Precise surface reconstructions can give us more accurate measurements and allow us to perform analyses that are otherwise challenging with voxel-based representations, such as cortical parcellation. Therefore, while both voxel and surface-based techniques are integral to the broader field of medical image segmentation, the work in this part of the thesis is devoted to surface-based segmentation, focusing specifically on cortical surface reconstruction and parcellation.

This part of the thesis differs from the previous part, not only by its emphasis on surface-based segmentation but also by the consistent theme of the presented works, which all aim to solve

the same task and are inherently built upon each other. In this chapter, I will explain the tasks of cortical surface reconstruction and parcellation, providing a detailed overview of relevant concepts such as mesh representations and topology. I will then reflect upon the related works, focusing on the third-party software FreeSurfer, followed by an introduction to the deep learning concept of Graph Convolutional Neural Networks. Finally, I will provide an overview of the data used in the subsequent chapters.

10.1 Cerebral Cortex

The brain's outermost layer is the cerebral cortex, responsible for higher cognitive functions, sensory processing, and motor control. The cortex is divided into distinct functional areas, which have been mapped and defined using various parcellation atlases. Magnetic Resonance Imaging (MRI) plays a crucial role in studying the anatomy and function of the cerebral cortex in vivo.

The surface of the cortex is highly folded, forming characteristic gyri (ridges) and sulci (valleys). This folding increases the surface area of the cortex, allowing for a higher density of neurons and more complex processing capabilities. The cortex is divided into two hemispheres, each of which is divided into four main lobes: the frontal, parietal, temporal, and occipital lobes. Each lobe is associated with specific functions, such as motor control in the frontal lobe, somatosensory processing in the parietal lobe, auditory processing in the temporal lobe, and visual processing in the occipital lobe.

The cortex's folding increases its surface area, allowing for more neurons and complex connectivity. The folding patterns of the human cerebral cortex develop through a complex process that begins in the embryonic stage and continues into early adulthood. During fetal development, the cortex undergoes rapid expansion, and the initially smooth surface begins to fold, creating the characteristic gyri and sulci [30]. As the brain ages, it experiences atrophy, characterized by a decrease in cortical thickness and a reduction in the number of neurons and synapses [25, 112]. This process can be exacerbated by neurodegenerative diseases, such as Alzheimer's disease, Parkinson's disease, and multiple sclerosis, which lead to more pronounced brain atrophy and cognitive decline [31]. MRI plays a crucial role in studying the development, aging, and pathology of the cerebral cortex by providing non-invasive, high-resolution images of the brain's structure and function, enabling researchers to investigate cortical folding patterns and track changes over time in both healthy and diseased brains.

10.2 Cortical Surface Reconstruction

Cortical surface reconstruction is a process employed in neuroimaging to transform 3D volumetric brain data, commonly derived from T1w MRI scans, into a set of 2D cortical surface representations. This task involves several steps, including extracting the cortical boundary from the MRI scans and removing topological defects. Notably, the output of this procedure typically includes four distinct surfaces - an inner (white matter) surface, representing the gray-white matter boundary, and an outer (pial) surface, representing the gray matter-pial

boundary, for each cerebral hemisphere. Figure 10.1 shows MRI scans of three different subjects of different ages, white matter, and pial surfaces. Cortical surface reconstruction primarily aims to formulate accurate geometric models of the cortical surfaces, effectively delineating the intricate gyri and sulci of the brain's cortex. These reconstructed surfaces are invaluable for subsequent tasks such as brain mapping, cortical thickness estimation, and cortical parcellation. Moreover, biomarkers such as cortical thickness play a crucial role in diagnosing neurodegenerative diseases like Alzheimer's.

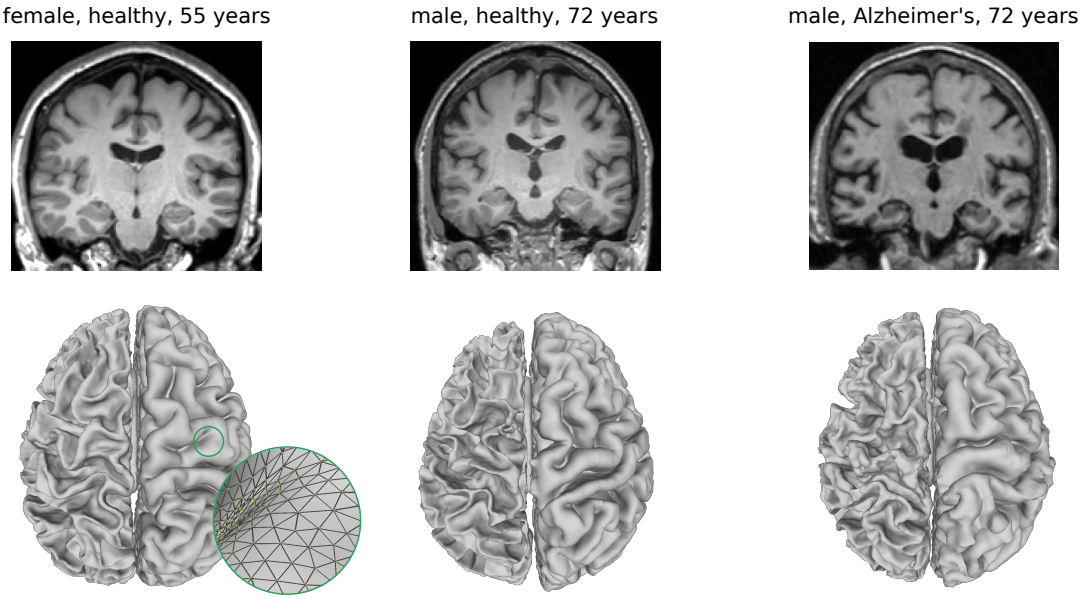

Figure 10.1. Top: MRI T1w scans from the ADNI dataset of subjects 55 and 72 years old with and without Alzheimer's disease. Bottom: Corresponding left white matter and right pial surfaces, extracted with FreeSurfer. Zoom in on the triangular mesh structure for the left-most pial mesh.

10.2.1 Triangular Meshes

Triangular meshes have been widely adopted as a standard representation of the cerebral cortex in medical imaging. Composed of interconnected vertices and edges that form triangular faces, these structures allow a comprehensive and precise geometric portrayal of the intricate cortical surface. Figure 10.1 shows the triangular structure on an example of a pial surface. One primary advantage of triangular meshes is their inherent flexibility. They can capture complex shapes and topologies with varying levels of detail, depending on the density of the triangulation.

10.2.2 Topology

Topology is a branch of mathematics that deals with the properties of geometric shapes that remain invariant under continuous deformations, such as stretching, twisting, or bending. In the context of surfaces and triangular meshes, topology is crucial for understanding the structure of the mesh and its underlying shape [33]. Preserving the topology in cortical surface reconstruction is important as it directly influences the accuracy and consistency of subsequent

analysis. In this subsection, I will explain relevant concepts of topology that will be often referred to in the later parts of this thesis.

Spherical Topology

A surface with genus-zero topology is characterized by having no holes or handles, making it topologically equivalent to a sphere. In the case of triangular meshes, the mesh can be deformed into a sphere without tearing or gluing its faces. A genus-zero surface has the property of being simply connected, meaning that any closed loop on the surface can be continuously deformed to a single point without leaving the surface. The surfaces of the human cerebral cortex have a genus-zero or spherical topology, and it is essential for cortical surface reconstruction approaches to guarantee this topology.

Self-Intersecting Faces

Self-intersecting faces occur when two or more faces of a mesh intersect or overlap, resulting in an inconsistent or non-manifold surface representation. These intersections can lead to ambiguities and issues when processing or analyzing the mesh, as they violate the basic assumption that each point on the surface is uniquely defined. Regarding topology, self-intersecting faces do not directly affect the genus of a mesh, as the underlying structure remains unchanged. However, they can cause difficulties in accurately computing surface properties that rely on a consistent surface representation, like surface area. The percentage of self-intersecting faces can be used as an evaluation metric to assess the surface quality.

Spheres and Icosahedrons

As a triangular mesh is a discrete representation, approximating a sphere using icosahedron expansion is a common technique in computer graphics and computational geometry for generating a mesh with approximately uniform vertex distribution on a spherical surface. The process starts with an icosahedron, a polyhedron with 20 equilateral triangular faces, 12 vertices, and 30 edges. The icosahedron undergoes a recursive subdivision to approximate the spherical surface more closely. FreeSurfer uses this approximation of a sphere, e.g., the spherical representation of the FsAverage template is an order seven icosahedron with 163,842 vertices, and the lower resolution fsaverage6 template is an order six icosahedron with 40,962 vertices. When we speak of a sphere in the subsequent chapters, we refer to an icosahedron approximation of a sphere, also called an icosphere.

10.2.3 FreeSurfer

Throughout this part, I frequently reference FreeSurfer [39] and often employ it as a benchmark or even a silver standard when training models. Consequently, I will provide an overview of FreeSurfer's capabilities here and delve deeper into the surface reconstruction process, as it is most relevant to the work showcased in this thesis. More recent related work, especially deep learning based techniques, will be explained in the following chapters.

FreeSurfer is an open-source software package to process and analyze human brain structural MRI data. The software offers a variety of tools for tasks such as reconstructing the brain's cortical surface, measuring cortical thickness, segmenting subcortical structures, and examining

longitudinal data. FreeSurfer is applicable in diverse research areas, including investigating neurodegenerative diseases, brain development, and the aging process. Researchers in the neuroimaging community have widely adopted FreeSurfer, often using it as a reference standard in their work.

The FreeSurfer pipeline involves the segmentation of subcortical structures of T1w MRI scans [40], the reconstruction of cortical surfaces [24, 41], inter-subject alignment of cortical surfaces based on cortical folding patterns [38], the parcellation of cortical surfaces [27, 28, 42] and measurement of cortical thickness [39].

The command for the FreeSurfer cortical surface reconstruction pipeline is `recon-all`. It involves an initial pre-processing step involving intensity normalization and skull stripping. Throughout the experiments presented in this thesis, we have always used the `recon-all` output `orig.mgz` as the input MRI scan, which has not yet undergone any intensity normalization or skull stripping unless mentioned otherwise.

The next step in the pipeline is segmenting the brain's white matter (WM) and gray matter (GM) tissues using a probabilistic atlas. This step also identifies the boundary between the WM and GM, which is the initialization for constructing the white matter cortical surface.

The pipeline generates an initial triangular mesh representation of the WM-GM boundary by tessellating the segmented brain volume [24]. It then applies an iterative refinement process to minimize the discrepancies between the mesh and the MRI data. This results in a smooth, accurate surface representation of the WM-GM interface, the `recon-all` output files are called `lh.white` and `rh.white`, and I refer to them as white matter surfaces. The pial surface, which represents the interface of the gray matter and cerebrospinal fluid (CSF), is reconstructed by a deformation of the white matter surface. The vertices of the WM surface are moved outward to find the GM-CSF boundary. This deformation is guided by an intensity gradient computed from the MRI data. The intensity gradient captures the differences in intensity between the gray matter and the surrounding cerebrospinal fluid [24]. The pipeline refines the initial estimate of the pial surface by minimizing an energy function that considers the MRI intensity values, surface smoothness, and prior anatomical knowledge. The goal is to obtain a pial surface closely following the GM-CSF boundary while maintaining smoothness and anatomical plausibility. Once the pial surface has been generated, it is checked for topological defects and self-intersections. If any issues are detected, the pipeline attempts to correct them using local smoothing or other heuristics. The output files are called `lh.pial` and `rh.pial`. The top row of Figure 10.2 shows the input MRI scan, overlayed voxel-based white matter segmentations, and the extracted white matter and pial surfaces for the left (yellow) and right (green) hemisphere. Since a deformation of the white surface generates the pial surface, the surfaces come with vertex correspondence between the two. FreeSurfer further computes morphological measurements like cortical thickness. For this, they use the correspondence between white and pial surfaces. The thickness is computed as the average distance measured from each surface to the other [39]. The reconstructed surface must be registered to a standard anatomical space to compare surfaces to a reference group. FreeSurfer provides average surfaces called FsAverage. It is an average representation of the cortical surface derived from many individual brains processed using the FreeSurfer cortical surface reconstruction pipeline [38]. The FsAverage template is a standard reference space for group-level analyses and inter-subject comparisons

in neuroimaging studies. Once the cortical surfaces are reconstructed for individual subjects, they are registered to the FsAverage template. First, the surface is inflated to create a spherical representation of the cortical surface. This inflated surface is then registered to a spherical atlas, the spherical representation of the FsAverage template. The spherical registration is based on the curvature computed on the original surface. The bottom row of Figure 10.2 shows how the white matter surfaces are inflated. The curvature is overlayed on the sphere, and the subject's sphere is registered to the FsAverage sphere. Based on the registration, FreeSurfer further maps an atlas parcellation from the FsAverage template, based on the Desikan Kiliany Tourville atlas [77] or Destrieux atlas [28], onto the surfaces. Registering individual subjects to the FsAverage template brings their cortical surfaces into a standard anatomical space. This allows for comparing and integrating data across subjects, even when their brain morphologies differ significantly. Once individual data have been normalized to the FsAverage space, researchers can perform group-level analyses, such as studying differences in cortical thickness or functional activation patterns between different populations (e.g., healthy controls vs. patients with a specific neurological disorder). The FsAverage template also provides a convenient reference for visualizing neuroimaging results on a standardized brain surface, such as statistical maps or cortical parcellations.

10.2.4 Marching Cubes Algorithm

The Marching Cubes algorithm bridges the world of voxels and the domain of surfaces, enabling the transition from volumetric data to a polygonal mesh representation. It is a widely-used computer graphics technique for creating a polygonal, usually triangular, mesh representation of an isosurface from a 3D scalar field, such as volumetric data from medical imaging. The Marching cubes Algorithm was first introduced by Lorensen et al. [94] and has since been advanced to be more computationally efficient by Lewiner et al. [87]. Throughout the methods presented in this thesis, we have always used the implementation by Lewiner et al. [87]. The primary goal of the Marching Cubes algorithm is to extract an isosurface, which is a surface that represents points of equal value (e.g., intensity) within the 3D scalar field. This isosurface can then be visualized, manipulated, or analyzed. The 3D scalar field is divided into a grid of cubes. Each cube has eight vertices, each with an associated scalar value.

For each cube in the grid, determine whether the isosurface intersects it by comparing the scalar values at the cube's vertices with the isosurface's threshold value (also called the isovalue). If the value at a vertex is greater than the threshold, it is considered inside the surface; otherwise, it is outside the surface. This step results in 256 possible cube configurations, which can be reduced to 15 unique cases due to symmetry. For each cube configuration, a predefined set of triangles is generated to approximate the isosurface within the cube. These triangles are created by interpolating the positions of the intersecting points along the cube's edges based on the scalar values at the vertices and the isovalue.

The triangles generated in the previous step are connected to form a continuous triangular mesh representing the isosurface. The Marching Cubes algorithm has been widely used in various fields, including medical imaging, computer graphics, and computational fluid dynamics. In the case of cortical surface reconstruction, it is commonly used to generate a

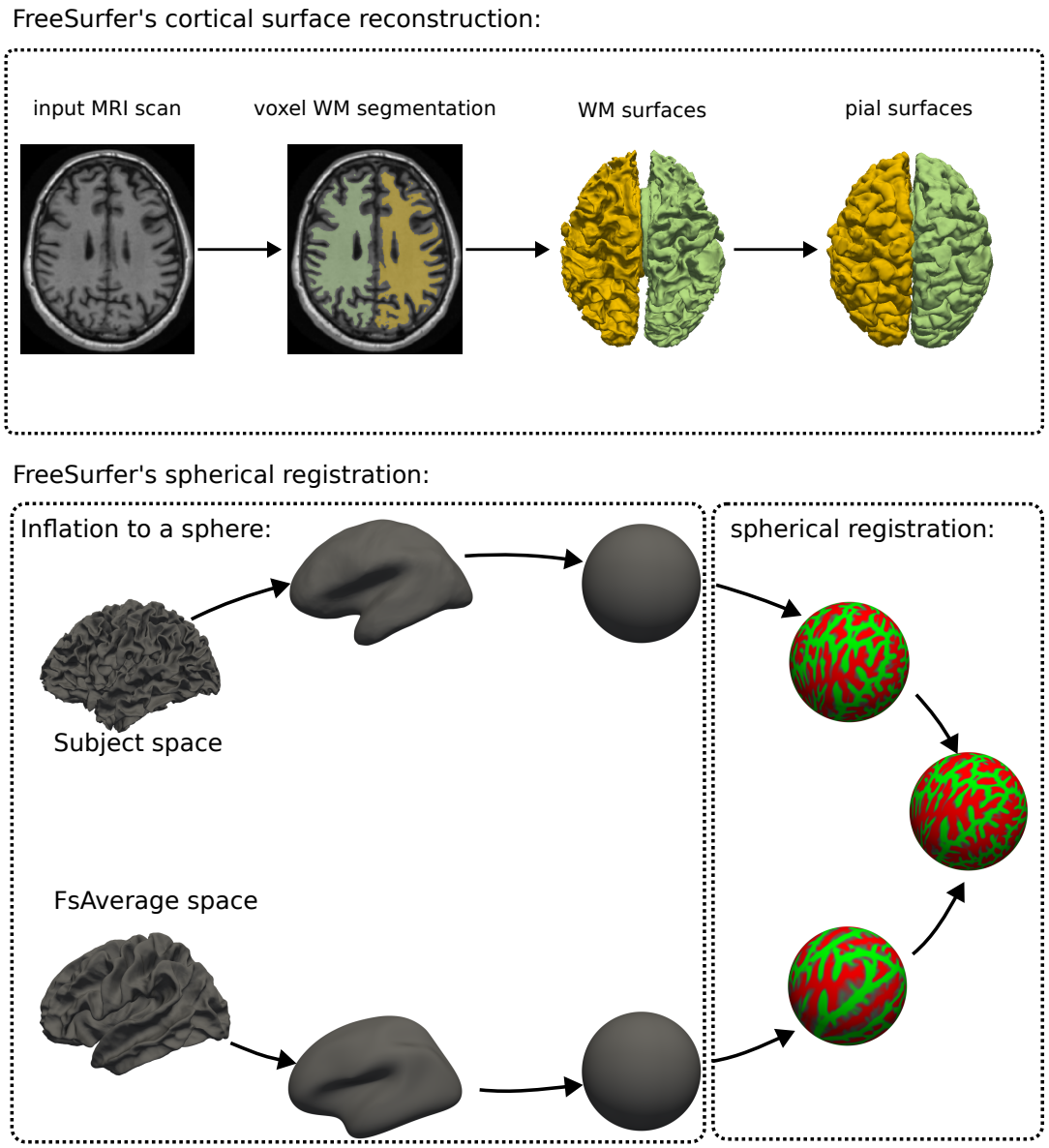

Figure 10.2. Top: Overview of FreeSurfer's cortical surface reconstruction. First, an input MRI scan is segmented (only white matter segmentation is shown for simplicity). Then an initial white matter surface is extracted (following refinement and topology correction omitted for simplicity), deforming to a pial surface. Bottom: FreeSurfer's spherical registration from subject space to FsAverage space. White matter surfaces of the subject and FsAverage template are iteratively inflated to a sphere. Curvature information is mapped to the sphere, and the subject's sphere is registered to the FsAverage sphere based on the curvature information.

surface representation from an initial voxel-based segmentation or Signed Distance Function (SDF).

10.3 Cortex Parcellation

Cortex parcellation is dividing the cerebral cortex into distinct regions or parcels based on various anatomical or functional attributes. These regions often correspond to different

brain areas associated with specific functions, such as motor control, vision, or language processing. Cortex parcellation is essential in neuroimaging analysis, as it facilitates studying structural and functional brain organization. It allows for quantitative measurements of specific brain regions, such as volume, thickness, or surface area, and enables region-based analysis of functional brain activities. Additionally, having a consistent parcellation scheme across individuals is crucial for group studies, enabling meaningful comparisons and statistical analysis across subjects. The process of parcellation, however, poses significant challenges due to the complex and highly variable structure of the human brain. Hence, various methods have been developed, with a trend toward machine learning and deep learning approaches in recent years.

Parcellation can be done using either voxel-based or surface-based techniques. Voxel-based parcellation is a variant of voxel-based brain segmentation, as, e.g., presented in Chapter 5. However, the number of classes significantly surpasses the roughly 30 class distinctions found in brain segmentation as in Chapter 5, often exceeding 100 classes, including subcortical brain structures and cortical parcels. Such an increase intensifies issues such as class imbalance and demands for computational resources and memory.

Given that the primary purpose of cortical parcellation is to enable per-parcel comparisons of cortical metrics, such as gyrification or cortical thickness, the parcellation is usually mapped onto cortical surfaces. As such, surface-based parcellation, which conducts the segmentation directly on the surface, is a logical choice. This method streamlines the process and enables a more integrated and intuitive approach to exploring the cerebral cortex.

Brain parcellation atlases are references or guides for dividing the cerebral cortex into distinct regions. These atlases differ based on their underlying principles, the granularity of divisions, and the specific brain features they focus on. I will not detail functional atlases as we deal with structural MRI scans throughout this thesis. Commonly used structural parcellation atlases include the Desikan-Killiany (DK) [27], Desikan-Killiany-Tourville (DKT) [78] and Destrieux [28, 42] atlases. An example of the Desikan-Killiany and Destrieux parcellations is visualized in Figure 10.3 These atlases are also used within the FreeSurfer software. They differ in the number of regions, e.g., the DK atlas divides the cortex into 34 areas per hemisphere, whereas the Destrieux atlas divides it into 74 areas per hemisphere.

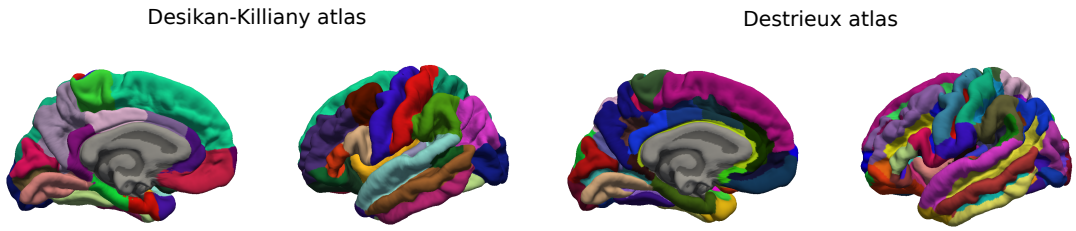

Figure 10.3. FreeSurfer's FsAverage template with Desikan-Killiany atlas and Destrieux.

10.4 Graph Convolutions

The methods presented throughout the chapters in this part will be based on combinations of convolutional neural networks and graph convolutional networks. For this, I will introduce the concept of graph convolutions here.

Graph Convolutions and Graph Neural Networks (GNNs) are tools in the machine learning landscape that operate on data structured as graphs. In essence, graphs are a more general structure than regular grid data (like images), representing complex relationships and interactions between entities. This is particularly relevant in the context of cortical surfaces, which can be naturally represented as triangular meshes, which are graphs with vertices corresponding to positions on the surface and edges representing adjacency relationships.

The standard convolutional operations in Convolutional Neural Networks (CNNs) are tailored for image processing and inherently assume a regular, grid-like data structure. On the other hand, graph convolutions adapt the concept of convolutions for irregular structures like graphs, allowing neural networks to assimilate information from the immediate neighbors of each graph node.

A GNN is a type of neural network that employs graph convolutions as its basic operation, akin to regular convolutions in CNNs. In the context of cortical surface parcellation, a GNN would be a natural choice to classify each vertex in the cortical surface graph into a specific parcel, effectively performing surface-based parcellation. Further, they can also be used to learn a deformation field on the vertices by learning to deform a mesh template to a patient's cortex, conditioned on the MRI scan.

Throughout the following chapters, our focus will be on employing spectrum-free graph convolutions, particularly in the style of message-passing operations, as referenced in [12]. We will adopt the implementation delineated in [130]. In the context of GNNs, the term 'message passing' describes the process by which a node updates its features based on the features of its neighboring nodes. Specifically, at each network layer, every node sends a 'message' containing its current features to its neighbors. Each receiving node then aggregates these messages to compute its new features.

In formal terms, a graph convolutional layer updates the features of a previous layer $\mathbf{f}_i \in \mathbb{R}^{d_{\text{in}}}$ for a vertex $\mathbf{v}_i \in \mathbb{R}^3$ by performing aggregation as shown:

$$\mathbf{f}_i' = \frac{1}{1+|N(i)|}\left[\mathbf{W}_0\mathbf{f}_i + \mathbf{b}_0 + \sum_{j\in N(i)} (\mathbf{W}_1\mathbf{f}_j + \mathbf{b}_1)\right], \tag{10.1}$$

Here, $\mathbf{W}_0, \mathbf{W}_1 \in \mathbb{R}^{d_{\text{out}} \times d_{\text{in}}}$ along with $\mathbf{b}_0, \mathbf{b}_1 \in \mathbb{R}^{d_{\text{out}}}$ signify linear transformations and $N(i)$ denotes the set of neighbors of $\mathbf{v}_i$. It is worth noting that, akin to CNNs, graph convolutional layers are typically succeeded by batch normalization and Rectified Linear Unit (ReLU) layers within our network models.

10.5 Challenges

At the time of publishing our initial work on this topic [10], there were scarce deep learning methodologies for cortical surface reconstruction. Among the few was DeepCSR [21], which employed an implicit surface reconstruction approach, learning a SDF used in marching cubes to extract the mesh. This method, however, depended on topology correction. We hypothesize that an explicit mesh reconstruction can evade this issue.

Due to its robustness, automatic software like FreeSurfer and CAT12 is widely utilized for cortical surface reconstruction in the research community. However, the lengthy runtimes of these tools, several hours per MRI scan, make their application in clinical routine impractical. Deep learning methods, with their potential to accelerate this process significantly, could provide a feasible solution.

But the path toward developing deep learning for cortical surface reconstruction is challenging. Firstly, there is a lack of manual ground truth labels, necessitating the reliance on third-party software like FreeSurfer as a silver standard.

Moreover, these surfaces need to be exceedingly precise. Subvoxel accuracy is a requirement, as the errors ideally should be significantly less than 1mm. This precision level is vital since cortical thickness changes often fall within this range. The surfaces are used to detect early changes, hence the need for such accuracy.

A final hurdle to consider is the requirement for topological correctness, an essential factor in ensuring that the reconstructed surface correctly represents the complex and unique topology of the cerebral cortex. Balancing these factors — speed, accuracy, and topological correctness — presents a challenging task for applying deep learning methods in cortical surface reconstruction.

Alongside speed, accuracy, and topological correctness, another feature that enhances the usability of tools like FreeSurfer is their ease of performing group comparisons. FreeSurfer provides the FsAverage template, which provides a standard anatomical space allowing data from different individuals to be analyzed collectively. Therefore, a desirable trait for a deep learning method is its ability to emulate these features. The development of tools that can combine the speed and efficiency of deep learning with such practical features, like the ease of comparison across groups, will make them more appealing and suitable for use in a clinical setting.

10.6 Datasets

Throughout this part, we use the same datasets and pre-processing, which I will describe here.

10.6.1 ADNI

Data utilized for the crafting of this thesis were sourced from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database, which is accessible at `adni.loni.usc.edu`. Initiated in 2003, ADNI emerged as a collaboration between public and private sectors, under the leadership of Principal Investigator Dr. Michael W. Weiner. The core objective of ADNI is to explore whether the integration of serial magnetic resonance imaging (MRI), positron emission tomography (PET), other biological markers, and clinical as well as neuropsychological assessments can effectively gauge the progression of mild cognitive impairment (MCI) and early-stage Alzheimer's disease (AD). Please visit `www.adni-info.org` for the most current information.

The ADNI database furnishes MRI T1 scans for individuals diagnosed with Alzheimer's Disease, Mild Cognitive Impairment, as well as for healthy subjects. We first removed data with processing artifacts, e.g. segmentation errors by FreeSurfer, detected by the UCSF quality control measures [55]. We then partitioned the dataset into training, validation, and testing subsets, ensuring a balanced representation in terms of diagnosis, age, and gender. Given that ADNI is conducted as a longitudinal study, only each participant's first (baseline) scan was employed.

For the experiments, two distinct partitions of the ADNI data were utilized. The first partition, termed as $\text{ADNI}_{\text{small}}$, encompasses 299 subjects in the training set, and 60 subjects each in the validation and testing sets. This subset was instrumental for the architecture ablation study experiments, as detailed in Chapter 11. The second partition, named $\text{ADNI}_{\text{large}}$, is more extensive, containing 1,155 subjects for training, 169 for validation, and 323 for testing.

10.6.2 OASIS

The OASIS-1 dataset [101] comprises MRI T1 scans from a total of 416 subjects, out of which 100 subjects are diagnosed with Alzheimer's disease ranging from very mild to moderate stages. The data was divided, ensuring balance based on diagnosis, age, and gender, which led to the allocation of 292 subjects for the training set, 44 for the validation set, and 80 for the testing set.

10.6.3 Test-Retest

Test-Retest dataset (TRT) [98] comprises 120 T1w MRI scans from three subjects, with each subject scanned twice on 20 different days. This dataset is primarily employed to assess the consistency of our models. We compare our model's predictions on scans taken on the same day, as we anticipate no structural changes in the brain and only minor variations due to patient positioning. A robust and consistent model should yield highly similar predictions for scans from the same day. Moreover, we use this dataset to evaluate the transfer learning capabilities of our models, examining how well models trained on other datasets adapt to this previously unseen data.

10.6.4 MALC

The MALC dataset [83] is used in chapter Chapter 11 for initial experiments and hyperparameter tuning. Here we split the dataset into 15 training scans, seven validation scans, and 8 test scans. But we only present results on the validation set.

10.6.5 J-ADNI

The Japanese ADNI Project (J-ADNI) (`https://www.j-adni.org/`) provides MRI T1w scans from healthy subjects and subjects with Alzheimer's disease from Japan. We use 502 baseline scans for testing models trained on ADNI for their generalizability to unseen data.

10.6.6 Mindboggle

We further test generalization to the Mindboggle-101 dataset [77] 101 MRI T1w scans. This dataset also contains manual parcellations; therefore, it is also used to evaluate parcellation accuracy.

10.6.7 Pre-Processing

In this part, the data was processed using FreeSurfer v5.3 or v7.2 [39]. We utilized orig.mgz files, along with white matter (WM) and pial surfaces (lh.white, lh.pial, rh.white, and rh.pial), generated by FreeSurfer. The MRI scans in orig.mgz format possess dimensions of $256 \times 256 \times 256$ with a voxel resolution of 1mm.

The pre-processing pipeline proposed by [21] was employed, which involved aligning the MRI scans to the MNI152 template [102] through rigid followed by affine registration. MNI152 is a widely recognized brain template crafted by averaging 152 healthy brain scans.

NiftyReg [111, 121] was utilized for the registration process. Excluding Chapter 13, when FreeSurfer meshes were employed as supervised training labels, they were simplified to approximately 40,000 vertices per surface via quadric edge collapse decimation [44]. This simplification was primarily conducted to economize memory usage.

After the registration to MNI space, the input MRI images exhibited dimensions of $182 \times 218 \times 182$ with a voxel resolution remaining at 1mm. For most experiments, unless explicitly stated otherwise, padding was applied to input images to achieve dimensions of $192 \times 208 \times 192$, followed by downsampling to $128 \times 144 \times 128$ voxels to save memory. Lastly, intensity values underwent min-max normalization, scaling them to fall within the $[0, 1]$ range.

11 Fast Explicit Reconstruction of Cortical Surfaces

Contents

11.1 Introduction

It is necessary to reconstruct the cortical surface as a triangular mesh to obtain accurate cortical thickness and volume measurements. While voxel-based segmentations can be used to obtain mesh-based surface representations, the resulting meshes often suffer from staircase artifacts and topological defects, which require post-processing steps to correct, see Figure 11.1. Traditional methods, such as FreeSurfer [39], provide accurate and topologically correct cortical surface meshes but are time-consuming and impractical for fast predictions in clinical use.

Recent research has introduced deep learning-based algorithms that can produce explicit surface representations of organs from Magnetic Resonance Imaging (MRI) scans without the need for post-processing [174]. However, at the time of publication, these methods had not been applied to reconstruct shapes with complex folding patterns, such as the cerebral cortex.

To address this gap, we created Vox2Cortex, a deep learning network that integrates convolutional and graph convolutional networks. By deforming an initial template, it is able to directly recreate explicit meshes of cortical surfaces from MRI images of the brain. Four output meshes are predicted by our network, including the white matter and pial surfaces of both hemispheres. The cortical meshes' spherical topology is maintained by the deformation of

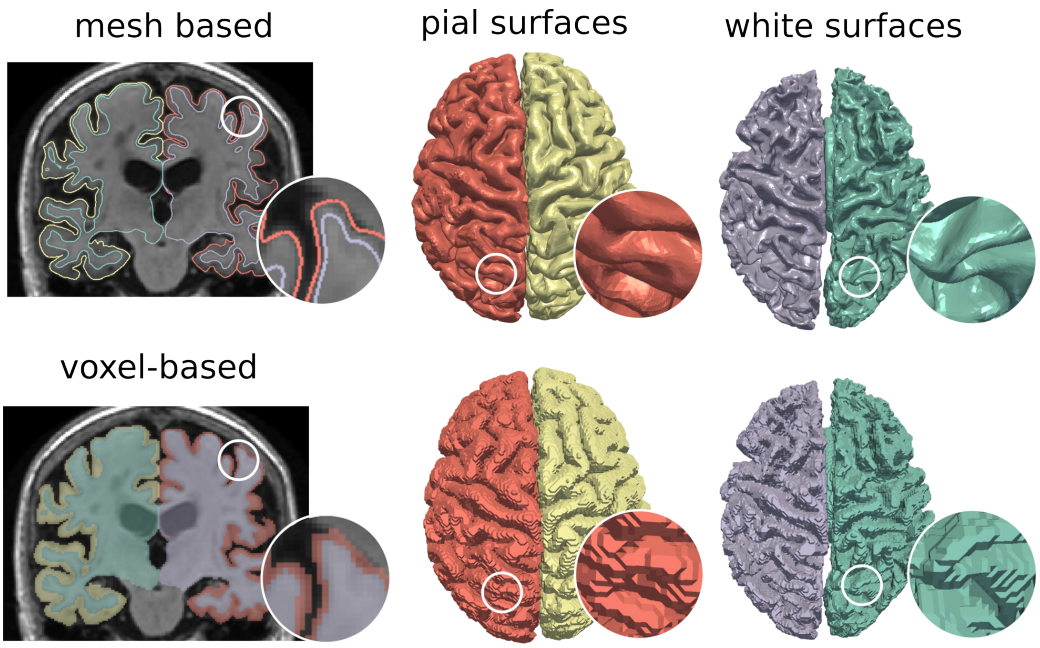

Figure 11.1. Brain MRI scans in the coronal plane, with overlays highlighting both mesh-based and voxel-based segmentations, and three-dimensional visualizations of the corresponding pial and WM surfaces. In the top row, one can observe the meshes that have been generated through our proposed method, while the bottom row displays meshes that were produced using the marching cubes algorithm applied to the voxel segmentation. Notably, the meshes in the bottom row exhibit distinct staircase-like irregularities, which are artifacts inherent to the marching cubes method. This figure has been adapted and modified from Figure 1 in [10] © IEEE 2022

an initially spherical surface. Consequently, our procedure prevents the formation of mesh openings or handles. We train our model on meshes with up to 168,000 vertices per mesh to accurately represent the extremely complex folding patterns of the cortex.

We have made several significant contributions through our work. Firstly, we have devised a rapid and highly precise method for recreating the cortex by integrating a convolutional neural network with a graph neural network. Secondly, we ensured that the generated meshes possess a spherical topology, which was achieved by deforming a template mesh with a fixed topology that can have any resolution. Further, we considered the interconnected nature of the white matter (WM) and pial surfaces and facilitated information exchange between them. Lastly, we introduced a novel loss function for training methods focused on explicit reconstruction. This function takes advantage of local curvature, which is used to assign weights to Chamfer distances.

Moreover, our technique demonstrates performance that is either equivalent to or surpasses the existing reconstruction methods, that rely on implicit representations or are voxel-based, concerning accuracy and consistency. Notably, our method is also 25 times faster during the inference phase. The practical applicability of our method has been demonstrated on downstream tasks, such as the measurement of cortical thickness and surface area.

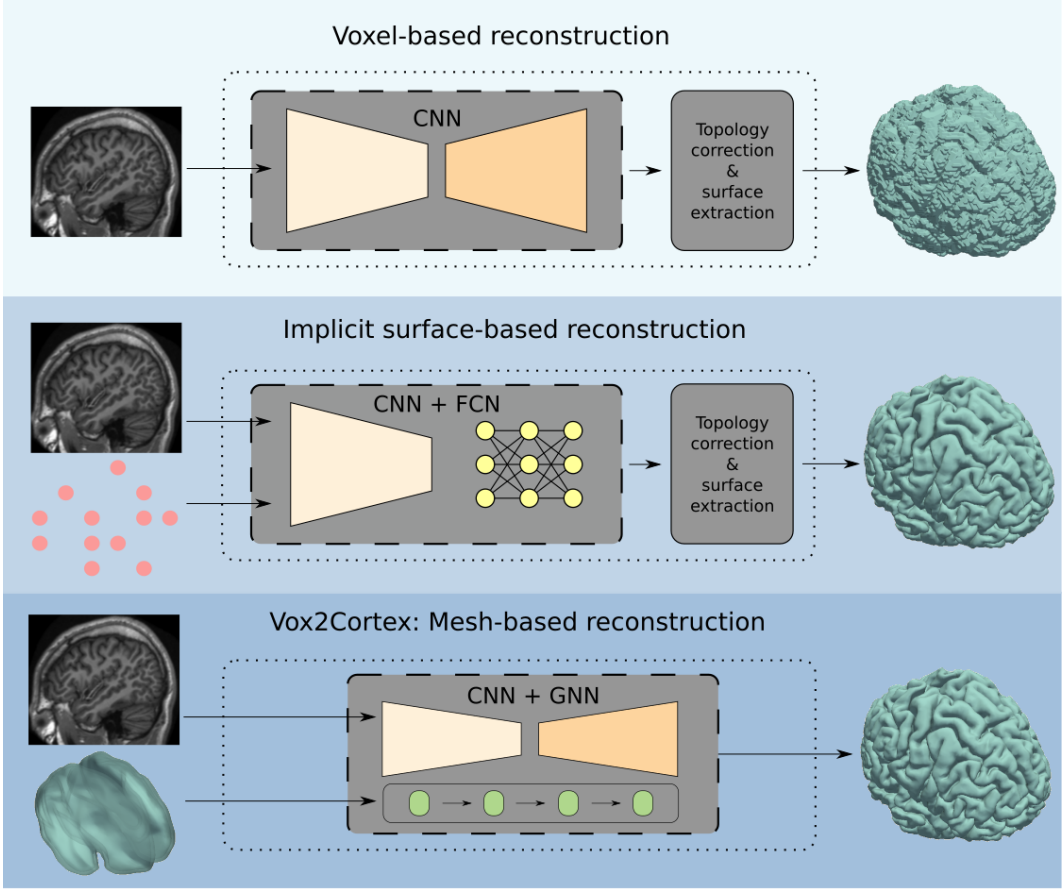

Figure 11.2. This figure illustrates that the prevailing deep learning-driven methods for reconstructing the cortical surface from MRI scans predominantly employ an implicit or voxel-based approach. Such approaches necessitate the execution of complex post-processing operations, which include correcting the topology and utilizing the marching cubes algorithm [87] to create explicit surface representations, such as triangular meshes. These meshes are indispensable for subsequent applications, like evaluating the thickness of the cortex. However, in stark contrast, our proposed model efficiently generates highly precise meshes of the white matter (WM) and the pial surfaces directly. The figure has been sourced from [10] © IEEE, 2022

11.2 Related Work

Traditionally, processing pipelines for brain MRI scans have been composed of multiple stages, such as the alignment of images (registration), dividing the images into segments (segmentation), and the extraction of cortical surfaces [24, 41, 156]. However, these processes require significant computational resources, leading to delays in obtaining measurements of the cortex following the completion of the scans. As an alternative, deep learning approaches for cortical surface reconstruction focused on voxel-based or implicit surface reconstruction methods when we developed Vox2Cortex, which can be categorized as a template-deformation approach. For a visual summary that compares the different methodologies, refer to Figure 11.2. Additionally, in the subsequent sections, we delve into a more comprehensive explanation of the methods related to computer vision and cortical surface reconstruction that were prevalent when this work was published.

11.2.1 Voxel-Based Surface Reconstruction

The idea of voxel-based surface reconstruction is first to segment an object, like the cerebral cortex, by voxel-based segmentation methods, e.g., Convolutional Neural Networks (CNNs), and then use a post-processing step for mesh extraction. A common algorithm to use is marching cubes [87], which requires additional topology correction. The marching cubes algorithm is explained in subsection 10.2.4. FastSurfer [59] is a deep learning approach, similar to QuickNAT [143], that focuses on cortex parcellation. For both subcortical and cortical segmentation, the FreeSurfer pipeline is sped up using a CNN; nonetheless, surface creation and topology correction still need marching cubes. For combined segmentation and surface reconstruction, SegRecon [48] presents a 3D CNN, which learns a 3D signed distance function and still needs marching cubes and topology correction.

11.2.2 Deep Implicit Representations

Deep implicit representations have recently gained significant traction as a field of study within 3D computer vision [105, 122, 153, 179]. The central concept involves training a function to associate 3D coordinates with a continuous implicit representation of a shape, usually in the form of a signed distance function (SDF) or an occupancy value. These functions can be learned by Multi-Layer Perceptrons (MLPs) either for an individual shape or a group of shapes, usually conditioned on an image. Early work in this field of research was DeepSDF [122], OccNet [105], or DISN. DeepCSR [21] has been proposed for cortical surface reconstruction. While the shape representation is continuous, the need for topology correction of the implicit field and marching cubes for mesh generation remains a drawback.

11.2.3 Template-Deformation Approaches

Template-deformation networks [79, 80, 169, 173, 174] deform a mesh iteratively by learning the deformation field of the vertices using an input image and a template mesh. Voxel2Mesh [174] and MeshDeformNet [79, 80] have shown promising results when applied to medical data, though they were mainly applied to simpler shapes like hippocampus or liver. While it remains to be evaluated if this approach can work well on cortex reconstruction, deformation-based approaches for surface reconstruction have an explicit shape representation that typically doesn't need post-processing. PialNN, as cited in [97], is designed to morph an initial WM surface into a pial surface through a sequence of graph convolutions. However, it necessitates using FreeSurfer to generate the WM surface.

11.3 Method

This section describes our proposed Vox2Cortex architecture, which achieves fast cortical surface reconstruction from MRI scans. Our model comprises a convolutional branch (voxel network) and a graph convolutional branch (mesh network), which we describe in detail in the following.

11.3.1 Vox2Cortex Architecture

Vox2Cortex utilizes both a 3D brain MRI scan and template meshes corresponding to each hemisphere's white and pial surfaces as inputs. The network concurrently calculates the deformation for all four of these template surfaces. As output, it produces the modified surfaces along with a voxel segmentation of the brain, specifically targeting the gray matter and the tissue surrounded by the gray matter. The design of Vox2Cortex is influenced by preceding methods [79, 80, 168, 174] and comprises two connected sub-networks. The first sub-network is a CNN, which processes voxels, and the second is a Graph Neural Network (GNN) that takes charge of the deformation of the meshes. These networks are connected by feature-sampling modules, which are responsible for mapping the features extracted by the CNN to the vertex locations found within the meshes. Figure Figure 11.3 visually represents the entire architecture, alongside examples of inputs and outputs. In the following sections, we elaborate on the various components of this system: the network that handles voxels, the network responsible for mesh deformation, and the mechanism for exchanging information between these networks.

Voxel Network (CNN)

We use a residual 3D UNet architecture [17, 68, 139, 182] for image-feature extraction. This encoder-decoder Fully-Convolutional Neural Network (F-CNN) solely takes the 3D brain scan as input and outputs a binary segmentation of the brain. The UNet's encoder comprises an initial convolutional layer that preserves the shape, four residual convolution blocks, and four downsampling layers. Each residual convolution block consists of two convolutional layers with a $3 \times 3 \times 3$ kernel size, followed by batch normalization and Rectified Linear Unit (ReLU) activation. The input is added to the residual output [57] before the final ReLU activation. To ensure that the residual input's channel number matches that of the residuum, we use a $1 \times 1 \times 1$ convolutional layer. The feature maps are downsampled using a $2 \times 2 \times 2$ convolution with a stride of 2 after each residual block.

The decoder of the UNet follows the same architecture as the image encoder, consisting of four residual convolution blocks. Before each block, the input is upsampled using transposed convolutions with a kernel size of $2 \times 2 \times 2$ and stride 2, matching the downsampling layers of the encoder. The feature maps from the corresponding size in the encoder are then added to the upsampled feature maps and fed to the residual decoder block. To allow for deep supervision, branches are created following the approach in [181] to propagate the segmentation loss to lower decoder layers. For the segmentation output of the final and deep supervision layers, a $1 \times 1 \times 1$ convolution and subsequent sigmoid classifier are used. The architecture of the voxel network is depicted in detail in Figure 11.3.

Mesh Network (GNN)

Our architecture comprises a GNN, which inputs a template mesh. This approach is inspired by related work [79, 80, 168, 174]. The choice of a GNN in comparison to an MLP [51, 170] is verified by our ablation study in Table11.3.

The GNN employs a series of four mesh-deformation steps to deform the templates, as depicted in Figure Figure 11.3. Within each of these steps, the GNN computes displacement vectors for

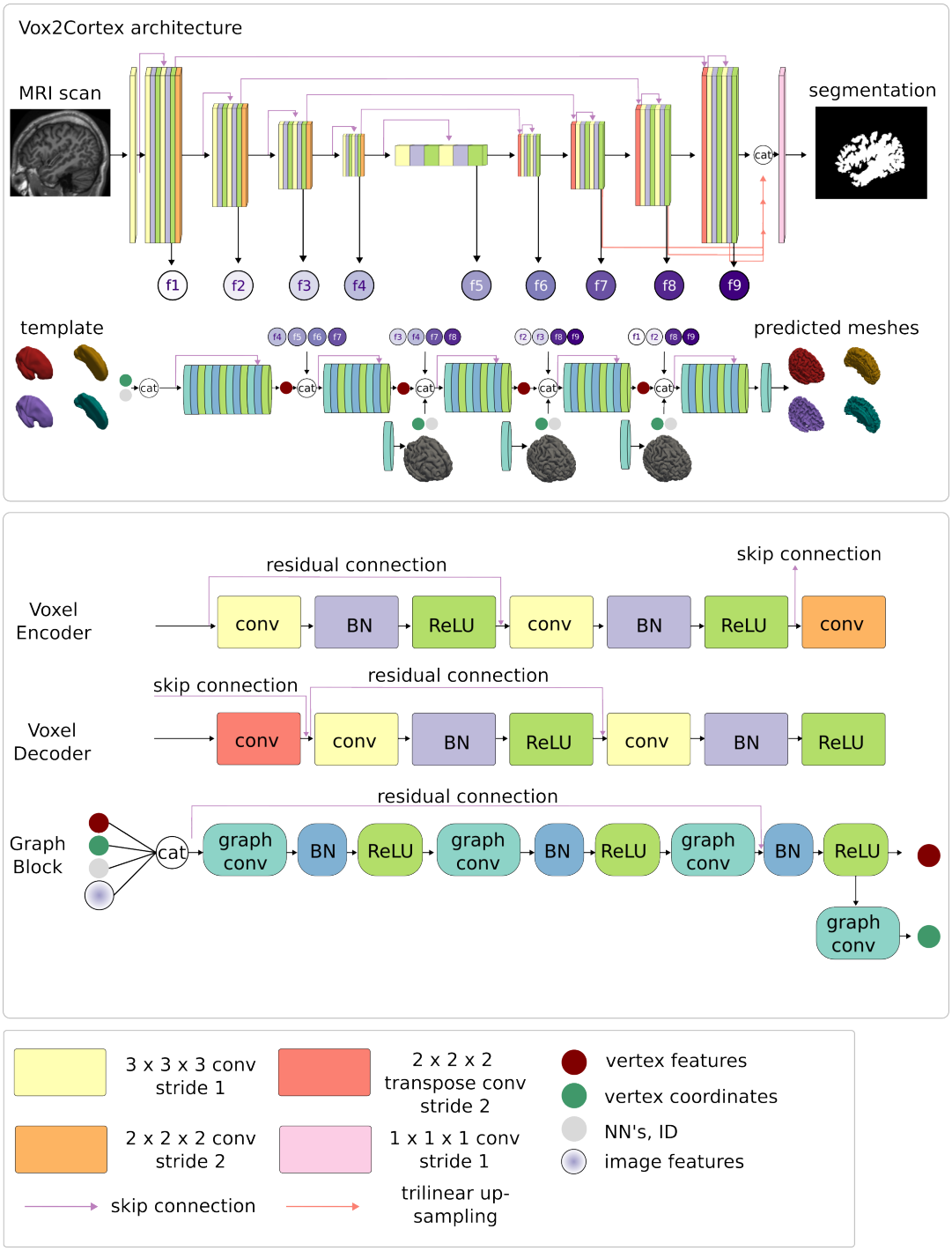

Figure 11.3. Illustration of the Vox2Cortex pipeline. Given a brain MRI scan and a mesh template as inputs, our network generates a voxel level segmentation alongside cortical surface meshes. The architecture is fundamentally built upon a voxel network, visualized as the U-Net style architecture at the top, and a GNN, visualized at the bottom. The GNN is tasked with deforming the initial template through four steps, employing features that describe both the image and shape to produce the final output meshes.

each vertex relative to the mesh produced in the preceding step. The architecture of the GNN has notable parallels with the voxel network in design, incorporating several residual blocks that enable learning based on residuals, as mentioned in [57, 80]. Each graph-residual block comprises three graph convolutions, followed by batch normalization and a ReLU activation.

The input residuum is added before the last ReLU activation. The vertices features are initially comprised of the three vertex coordinates and information on the neighbors, which is explained in more detail in subsection 11.3.1. As the length of these vertex feature vectors is much smaller than the image feature vectors, the initial graph-residual block has more channels than the later blocks to upscale the vertex features, as proposed in [80]. Each of the four mesh deformation blocks outputs the new vertex features, illustrated as red dots in Figure 11.3, as well as the displacement vectors for each vertex at the current deformation stage. The new vertex coordinates after deformation, illustrated as green dots in Figure 11.3, are then concatenated to the vertex features as input to the next mesh deformation block.

To implement our GNN, we utilize spectrum-free graph convolutions based on message-passing operations [12], implemented by the framework proposed in [130].

Combining CNN and GNN

Although CNNs and GNNs are well-established network architectures, there have been few attempts to combine them, and the optimal way to exchange information between them remains unanswered [79, 80, 168, 174].

In Vox2Cortex, we feed information from the voxel network into the graph network to condition the template deformation process on the image. We achieve this by sampling from multiple CNN feature maps at locations determined by the vertex coordinates of the mesh prediction of the current GNN block. Trilinearly interpolating feature maps from discrete voxels is necessary to obtain features in continuous 3D space. These image features are then concatenated with the vertex features in the mesh network. This process is illustrated as circles f1 to f9 in Figure 11.3.

While it may seem intuitive to use high-level image features, such as those from the CNN bottleneck, for early deformation stages and more granular features for later stages, previous approaches have extracted this information either only from the encoder or only from the decoder. We argue that concatenating features from both the encoder and the decoder at multiple resolutions is more reasonable as it allows the network to optimize its decision about which information to use during training.

Interdependence of White and Pial Surfaces

At the time of publication, template deformation approaches have been mainly applied to reconstruct single objects. However, we deal with multiple surfaces, which are interdependent, as the inner and outer brain surfaces are always aligned. To enhance the reconstruction quality, we incorporate information exchange between the meshes to model their interdependence.

For each surface vertex, we design an additional feature vector. This feature vector contains the coordinates of the five nearest vertices on the other surface, meaning for a white matter vertex, we find the nearest vertices on the corresponding pial surface and vice versa, and a surface identifier (value between 0 and 3). This feature vector is depicted as gray dots in Figure 11.3 and concatenated to the existing vertex features. This enables the network to incorporate the spatial relationships between the different surfaces. This is in accordance with methods like

FreeSurfer [24], which inherently model the inter-dependency by extracting the pial surface based on the white one.

Mesh Templates

Existing template deformation methods in the literature commonly use simple mesh templates like spheres or ellipsoids[79, 80, 168, 174]. However, we found that the accuracy of our reconstructed surfaces was not satisfactory using the spherical or ellipsoid templates (cf. subsection 11.4.2). Therefore, we propose to use a more realistic brain-shaped template for surface reconstruction. To create the new templates, we randomly selected FreeSurfer meshes from the MALC dataset [83] and applied Laplacian smoothing [166] until no further changes occurred. The resulting templates, depicted in Figure 11.3, were used as input to the network during training. Interestingly, we observed that we could use a higher resolution template during testing than training, thereby increasing the surface accuracy of a trained model. This allows us to choose the desired mesh resolution independently of the template used during training. During training, we used about 42,000 vertices per surface, while during testing, we increased this number to approximately 168,000 vertices per surface. This results in a total of over 672,000 vertices across all four surfaces.

11.3.2 Loss Functions

In this section, we outline the construction of the loss function used to train the Vox2Cortex network, focusing on the innovative aspect of our curvature-weighted Chamfer loss.

Let $\hat{\mathcal{Y}} = \{\hat{M}_{s,c}, \hat{\mathbf{\Delta}}_{s,c}, \hat{\mathbf{B}}_l | s = 1, \dots, S;, l = 1, \dots, L; c = 1, \dots C\}$ be the model's prediction. Here, $\hat{M}_s = (\hat{\mathbf{V}}_s, \hat{\mathbf{F}}_s, \hat{\mathbf{E}}_s)$ represents the meshes that are predicted at S distinct stages of deformation, and each stage is comprised of C different surfaces. In our implementation, C is set to four, accounting for each hemisphere's white and pial surfaces. $\hat{\mathbf{\Delta}}_s$ stands for the predicted displacement vectors compiled into a tensor, and $\hat{\mathbf{B}}_l \in [0, 1]^{HWD}$ signifies the voxel-by-voxel binary segmentation maps, which is the final output of the CNN and includes $L - 1$ deep-supervision outputs.

Training our model follows a supervised approach, and we presume the availability of corresponding ground-truth meshes and segmentation maps, represented as $\mathcal{Y} = \{M, \hat{\mathbf{B}}\}$, for each example in the training dataset. The loss function used to train Vox2Cortex is composed of two main components: a voxel part, $\mathcal{L}_{\text{vox}}$, and a mesh part, $\mathcal{L}_{\text{mesh}}$:

$$\mathcal{L}(\hat{\mathcal{Y}}, \mathcal{Y}) = \mathcal{L}_{\text{vox}}(\hat{\mathcal{Y}}, \mathcal{Y}) + \mathcal{L}_{\text{mesh}}(\hat{\mathcal{Y}}, \mathcal{Y}). \tag{11.1}$$

Voxel Loss

Let $\mathcal{L}_{\text{BCE}}(\hat{\mathbf{B}}_l, \mathbf{B})$ denote the binary cross-entropy loss between a predicted segmentation map $\hat{\mathbf{B}}_l \in [0, 1]^{HWD}$ and the corresponding ground truth segmentation map $\mathbf{B} \in 0, 1^{HWD}$.

Considering that we have L segmentation outputs, which include the final segmentation of the CNN and $L-1$ deep-supervision outputs, the voxel loss is computed as follows:

$$\mathcal{L}_{\text{vox}}(\hat{\mathcal{Y}}, \mathcal{Y}) = \sum_{l=1}^{L} \mathcal{L}_{\text{BCE}}(\hat{\mathbf{B}}_l, \mathbf{B}). \tag{11.2}$$

The purpose of the voxel network is to facilitate the learning of meaningful feature maps, and one way to achieve this is by learning the segmentation of the brain. Since the segmentation is binary, we discovered that using a simple segmentation loss such as the $\mathcal{L}_{\text{BCE}}$ was effective. Therefore we did not utilize a more complex segmentation loss as we did in the previous part.

Mesh Loss

Defining an appropriate loss function for two meshes is more difficult than for voxel loss, primarily due to the absence of point correspondences between predicted meshes and labels. This makes creating a loss function that enforces an accurate cortical surface mesh challenging. Drawing inspiration from previous template deformation approaches [80, 168, 174], we employ a combination of geometry-consistency and regularization losses:

$$\mathcal{L}_{\text{mesh}}(\hat{\mathcal{Y}}, \mathcal{Y}) = \mathcal{L}_{\text{mesh},\, cons}(\hat{\mathcal{Y}}, \mathcal{Y}) + \mathcal{L}_{\text{mesh},\, reg}(\hat{\mathcal{Y}}). \tag{11.3}$$

The geometry-consistency loss comprises our novel curvature-weighted Chamfer loss $\mathcal{L}_{\text{C}}$, detailed in the next section, and an inter-mesh normal consistency (also referred to as normal distance [46]) $\mathcal{L}_{\text{n}}$:

$$\begin{aligned} \mathcal{L}_{\text{mesh},\, cons}(\hat{\mathcal{Y}}, \mathcal{Y}) = \sum_{s=1}^{S} \sum_{c=1}^{C} \Big[& \lambda_{1,c}, \mathcal{L}_{\text{C}}(\hat{M}_{s,c}, M_c) \\ & + \lambda_{2,c}\, \mathcal{L}_{\text{n}}(\hat{M}_{s,c}, M_{s,c}) \Big]. \end{aligned} \tag{11.4}$$

The regularization loss consists of Laplacian smoothing of the displacement fields $\mathcal{L}_{\text{Lap}}$, intra-mesh normal consistency $\mathcal{L}_{\text{n},\, intra}$, and edge length $\mathcal{L}_{\text{edge}}$

$$\begin{aligned} \mathcal{L}_{\text{mesh},\, reg}(\mathcal{Y}^{\text{p}}) = \sum_{s=1}^{S} \sum_{c=1}^{C} \Big[& \lambda_{3,c}\, \mathcal{L}_{\text{Lap}}(\hat{M}_{s,c}, \mathbf{\Delta}^{\text{p}}_{s,c}) \\ & + \lambda_{4,c}\, \mathcal{L}_{\text{n},\, intra}(\hat{M}_{s,c}) \\ & + \lambda_{5,c}\, \mathcal{L}_{\text{edge}}(\hat{M}_{s,c}) \Big]. \end{aligned} \tag{11.5}$$

More details about the loss hyperparameters can be found in the Results section. We describe each loss function in detail in the following.

Curvature-Weighted Chamfer Loss

The Chamfer distance has become an essential tool in learning deformable shape models for surfaces [36, 168]. Nonetheless, it's imperative to couple it with regularization terms to avoid edge intersections that may occur otherwise [54]. Specifically, in regions characterized by

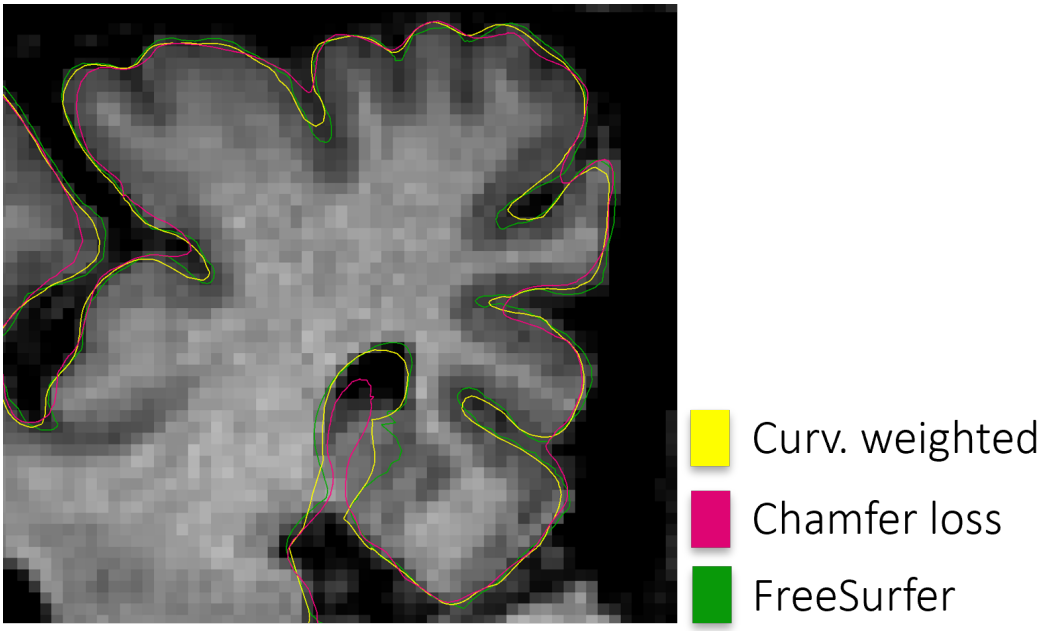

Figure 11.4. This figure demonstrates the impact of employing the curvature-weighted Chamfer loss for training Vox2Cortex in contrast to using the conventional Chamfer loss, along with a comparison to the FreeSurfer ground truth. Utilizing the curvature-weighted Chamfer loss results in surface meshes of higher fidelity, with the cortical folds being more precisely captured.

pronounced curvature, such as the intricate folding patterns of the cortex, the smoothing effect brought about by the regularization terms can cause a decrease in geometric precision.

To tackle this problem, we introduce a curvature-weighted Chamfer loss function that emphasizes regions with high curvature, thus improving the reconstruction of areas with a dense folding pattern, as depicted in Figure Figure 11.4. Further, we provide a proof in the Appendix that substantiates how this loss function actively nudges the predicted points in high-curvature zones closer to their true positions compared to those in low-curvature areas, given certain mild assumptions.

It is critical to highlight that the curvature weights are exclusively derived from ground-truth data points, as reliance on the curvature from a prediction might be unreliable and steer the model in the wrong direction.

We take into account a curvature function, denoted as $\kappa(\mathbf{p}) \in \mathbb{R}_{\geq 0}$, which assigns a curvature value to a given point $\mathbf{p}$. With this, the curvature-weighted Chamfer loss is defined as follows:

$$\begin{aligned} \mathcal{L}_{\mathrm{C}}(\hat{M}_{s,c}, M_c) = &\frac{1}{|\mathcal{P}_c|} \sum_{\mathbf{u} \in P_c} \kappa(\mathbf{u}) \min_{\mathbf{v} \in \hat{\mathcal{P}}_{s,c}} \|\mathbf{u} - \mathbf{v}\|^2 \\ &+ \frac{1}{|\hat{\mathcal{P}}_{s,c}|} \sum_{\mathbf{v} \in \hat{\mathcal{P}}_{s,c}} \kappa(\tilde{\mathbf{u}}) \min_{\mathbf{u} \in \mathcal{P}_c} \|\mathbf{v} - \mathbf{u}\|^2, \end{aligned} \tag{11.6}$$

where $\tilde{\mathbf{u}} = \underset{\mathbf{r}\in\mathcal{P}_c}{\arg\min}\|\mathbf{v}-\mathbf{r}\|^2$. In practice, we found that

$$\kappa(\mathbf{p}) = \min\{1+\bar{\kappa}(\mathbf{p}),\, \kappa_{\max}\}. \tag{11.7}$$

In this case, $\bar{\kappa}(\mathbf{p})$ represents the discrete mean curvature [106, 114], and it has proven to be a suitable choice (with $\kappa_{\max} = 5$ in our experiments). $\mathcal{P}_c$ and $\hat{\mathcal{P}}_{s,c}$ denote sampled point clouds from the surfaces of the ground truth and predicted mesh. During the training phase, each $\mathcal{P}_c|c = 1,\ldots,C$ contains a number of vertices equal to that in the smallest ground-truth surface in the training dataset. Further, the point clouds $\hat{\mathcal{P}}_{s,c}|c = 1,\ldots,C; s = 1,\ldots,S$ are sampled from the surface of the predicted meshes in a differentiable manner [46, 154]. This ensures that the sampled point clouds have the same number of points as the reference meshes.

Inter-mesh normal consistency loss

The Chamfer distance is primarily concerned with the spatial alignment of two meshes, making sure that the points on the surfaces are positioned accurately. In contrast, the cosine distance considers the meshes' orientation. Typically, the cosine distance can be computed within a single mesh, termed intra-mesh normal consistency, or between two meshes, referred to as inter-mesh normal consistency. The computation of the inter-mesh normal consistency loss involves using the normal vectors of the nearest points between the predicted and ground-truth meshes. Suppose $\hat{\mathcal{P}}_{s,c}$ and $\mathcal{P}_c$ are the predicted and ground-truth point clouds, with associated normals $\hat{\mathbf{N}}_{s,c} = \mathbf{n}(\mathbf{p})|\mathbf{p}\in\hat{\mathcal{P}}s, c$ and $\mathbf{N}_c = \mathbf{n}(\mathbf{p})|\mathbf{p}\in\mathcal{P}_c$, respectively. The inter-mesh normal consistency loss can then be expressed as follows:

$$\begin{aligned}\mathcal{L}_{\mathrm{n}}(\hat{M}_{s,c}, M_c) = &\frac{1}{|\mathcal{P}_c|}\sum_{\mathbf{u}\in\mathcal{P}_c} 1-\cos(\mathbf{n}(\mathbf{u}),\mathbf{n}(\tilde{\mathbf{v}})) \\ &+\frac{1}{|\hat{\mathcal{P}}_{s,c}|}\sum_{\mathbf{v}\in\hat{\mathcal{P}}_c} 1-\cos(\mathbf{n}(\mathbf{v}),\mathbf{n}(\tilde{\mathbf{u}})),\end{aligned} \tag{11.8}$$

where $\tilde{\mathbf{v}} = \underset{\mathbf{r}\in\hat{\mathcal{P}}_{s,c}}{\arg\min}\|\mathbf{u}-\mathbf{r}\|^2$ and $\tilde{\mathbf{u}} = \underset{\mathbf{r}\in\mathcal{P}_c}{\arg\min}\|\mathbf{v}-\mathbf{r}\|^2$.

For every specific point $\mathbf{p}$, its normal vector is contrasted with the normal of the closest neighboring point in the corresponding point set. Given that the nearest-neighbor correspondences are also essential for the computation of the Chamfer loss, the same point sets $\hat{\mathcal{P}}$ and $\mathcal{P}$ are used in our implementation to save computation time.

Intra-Mesh Normal Consistency Loss

Instead of using the cosine distance for comparing the normals of two separate meshes, a different methodology involves analyzing the normal vectors of two neighboring faces within the same mesh. Faces are considered adjacent if they possess a shared edge. We term this metric of mesh smoothness intra-mesh normal consistency ($\mathcal{L}_{\mathrm{n},intra}$), and it is defined as follows:

$$\mathcal{L}_{\mathrm{n},\,intra}(\hat{M}_{s,c}) = \frac{1}{|\hat{\mathbf{E}}_{s,c}|}\sum_{a,b\in\mathbf{E}_{s,c}} (1-(\hat{\mathbf{n}}_a\cdot\hat{\mathbf{n}}_b))^2, \tag{11.9}$$

where $\hat{\mathbf{n}}i$ represents the unit normal of the i-th face of $\hat{M}_{s,c}$. This loss attains its minimum for meshes with no curvature, leading to smoother surfaces. Since this loss is calculated solely using a predicted mesh and doesn't consider any ground truth, it falls under the category of mesh-regularization losses.

Laplacian Loss

The smoothness of a mesh can also be assessed through the employment of the uniform Laplacian operator $\mathbf{L} = \mathbf{D}^{-1}\mathbf{A} - \mathbf{I}$, in which $\mathbf{D}$ signifies the degree matrix and $\mathbf{A}$ represents the adjacency matrix of the mesh. Laplacian smoothing, in particular, is defined as:

$$\mathcal{L}_{\text{Lap}}(\hat{M}_{s,c}) = \frac{1}{|\hat{\mathbf{V}}_{s,c}|} \sum_{i=1}^{|\mathcal{V}_{s,c}|} \|(\hat{\mathbf{L}}_{s,c}.\boldsymbol{\Delta}_{s,c})_i\| \tag{11.10}$$

The technique is renowned for its efficacy in creating smooth meshes [114]. Numerous research [80, 168, 174] generally apply mesh smoothing to vertex coordinates $\hat{\mathbf{V}}_{s,c}$, but, drawing inspiration from [183], our method employs the Laplacian operator to the displacement field $\boldsymbol{\Delta}_{s,c}$. Our ablation study confirms the effectiveness of this approach. More precisely, $\boldsymbol{\Delta}_{s,c}$ are the displacement vectors which morve the vertices $\hat{\mathbf{V}}_{s-1,c}$ to $\hat{\mathbf{V}}_{s,c}$, implying that the equation $\hat{\mathbf{V}}_{s,c} = \hat{\mathbf{V}}_{s-1,c} + \boldsymbol{\Delta}_{s,c}$ transforms the mesh $\hat{M}_{s-1,c}$ into $\hat{M}_{s,c}$.

Although a Laplacian loss cannot ensure that the predicted meshes are entirely free of self-intersections, it does promote smoother surfaces with fewer self-intersections. It is essential to mention that in Equation (11.10), $\hat{\mathbf{L}}_{s,c}$ is treated as a constant, and therefore the loss is not propagated back through the creation of $\hat{\mathbf{L}}_{s,c}$.

Edge Loss

Another loss function that can be used to regularize predicted meshes is the edge loss, which is defined as follows for a predicted mesh $\hat{M}_{s,c}$:

$$\mathcal{L}_{\text{edge}}(\hat{M}_{s,c}) = \frac{1}{|\hat{\mathbf{E}}_{s,c}|} \sum (i,j) \in \hat{\mathbf{E}}_{s,c} \|\mathbf{v}_i - \mathbf{v}_j\|^2. \tag{11.11}$$

The edge loss encourages the predicted meshes to have homogeneous edge lengths, resulting in a more even distribution of vertices on the surface.

11.4 Experiments and Results

11.4.1 Implementation Details

PyTorch v1.7.1 `https://pytorch.org/` and pytorch3D v0.4.0 `https://pytorch3d.readthedocs.io` were used to create our implementation. One NVIDIA Quadro GPU and one NVIDIA Titan RTX GPU, both with 24GB of RAM, were used in the studies. We used Python 3.88, CUDNN 7.6.5, and CUDA 10.2.89. We also used the Voxel2Mesh [174] and DeepCSR [21] repositories `https://bitbucket.csiro.au/projects/CRCPMAX/repos/deepcsr/browse` and `https:`

Table 11.1. Hyperparameters used in our experiments.

Optimizer	CNN learning rate	GNN learning rate	Batch size	Mixed precision	CNN channels	GNN channels	Gradient clipping
Adam [76] $\beta_1 = 0.9$, $\beta_2 = 0.999$	$1e^{-4}$	$5e^{-5}$	2 (1 for OASIS)	yes	16, 32, 64, 128 , 256, 64, 32, 16, 8	255, 64, 64, 64, 64	$2e^{5}$

Table 11.2. Hyperparameters weighting the different mesh loss functions: Chamfer loss, inter-mesh normal consistency, laplacian loss, intra-mesh normal consistency, and edge length loss, for white and pial surfaces.

Surface	chamfer $\lambda_{1,c}$	inter-mesh NC $\lambda_{2,c}$	Laplacian $\lambda_{3,c}$	intra-mesh NC $\lambda_{4,c}$	edge $\lambda_{5,c}$
$c =$ wm	1.0	0.01	0.1	0.001	5.0
$c =$ pial	1.0	0.0125	0.25	0.00225	5.0

`//github.com/cvlab-epfl/voxel2mesh/blob/master/README.md`, respectively. Our code is available at `https://github.com/ai-med/Vox2Cortex` for easy access and replication.

Hyperparameters

The hyperparameters for the model are outlined in Table Table 11.1. The models were trained for 100 epochs on the OASIS and $\text{ADNI}_{\text{small}}$ datasets and 40 epochs on the $\text{ADNI}_{\text{large}}$ dataset. The best model was selected based on the respective validation set, considering voxel Intersection over Union (IoU) and Hausdorff distance as evaluation metrics.

To enhance the reconstruction quality, we incorporated surface class-specific mesh-loss weights, despite the increased number of hyperparameters. Specifically, we found that different weights were necessary for the white matter and pial surfaces. To determine the optimal mesh-loss weights, we independently fine-tuned them for the white matter and pial surfaces using the small MALC dataset [83]. This tuning process focused on one hemisphere at a time and disregarded the corresponding other surfaces. Table Table 11.2 provides the resulting loss weights. Typically, the regularization weights for pial surfaces were higher, because the ground truth white surfaces are less smooth and contain regions of higher curvature.

Initially, a grid search was performed to fine-tune the mesh-loss function weights for inter-mesh normal consistency ($\mathcal{L}_{\text{n}}$), intra-mesh normal consistency ($\mathcal{L}_{\text{n},intra}$), and Laplacian smoothing ($\mathcal{L}_{\text{Lap}}$). The grid search involved testing values of 0.1, 0.01, and 0.001. Subsequently, the best values obtained from the grid search were further refined using values of $x + 0.5x$, x, and $x - 0.5x$, where x represented the respective best value from the initial tuning. During this process, the weights for the Chamfer and edge losses were kept constant at 1, while the edge-loss weight was separately fine-tuned with values of 1, 5, and 10.

Table 11.3. The ablation study's outcomes are summarized in the table below using the Average Symmetric Surface Distance (ASSD) and 90th-percentile Hausdorff Distance (HD-90) metrics. The use of a reduced-vertex-count (42,000 as opposed to 168,000) test template is denoted by an asterisk (*). For ease of understanding, we have bolded the best values. For a detailed discussion of the various variants tested, see Section 11.4.2. Millimeters (mm) are used for all measurements.

	Left WM Surface		Right WM Surface		Left Pial Surface		Right Pial Surface	
	ASSD	HD-90	ASSD	HD-90	ASSD	HD-90	ASSD	HD-90
Vox2Cortex	**0.401 ±0.065**	**0.894 ±0.177**	**0.403 ±0.057**	**0.896 ±0.142**	**0.375 ±0.055**	0.965 ±0.210	**0.378 ±0.060**	1.012 ±0.248
Vox2Cortex*	0.455 ±0.063	1.057 ±0.195	0.457 ±0.056	1.055 ±0.145	0.467 ±0.057	1.316 ±0.278	0.470 ±0.0611	1.371 ±0.281
Voxel2Mesh* [174]	0.528 ±0.222	1.209 ±0.732	0.528 ±0.197	1.186 ±0.625	0.486 ±0.114	1.457 ±0.398	0.476 ±0.108	1.440 ±0.384
Encoder features	0.453 ±0.072	0.984 ±0.177	0.456 ±0.054	1.007 ±0.144	0.432 ±0.067	1.057 ±0.211	0.430 ±0.059	1.040 ±0.174
Classic Chamfer	0.852 ±0.081	2.175 ±0.340	0.985 ±0.074	2.282 ±0.313	0.716 ±0.063	1.906 ±0.282	0.913 ±0.056	2.391 ±0.160
w/o inter-mesh NNs	0.444 ±0.063	0.960 ±0.174	0.438 ±0.052	0.958 ±0.142	0.390 ±0.051	**0.892 ±0.146**	0.396 ±0.049	0.946 ±0.168
Ellipsoid template	0.459 ±0.065	0.970 ±0.145	0.452 ±0.071	0.954 ±0.140	0.407 ±0.044	0.948 ±0.145	0.412 ±0.053	0.983 ±0.201
w/o voxel decoder	0.413 ±0.069	0.914 ±0.168	0.424 ±0.065	0.928 ±0.150	0.392 ±0.057	0.916 ±0.147	0.400 ±0.059	**0.942 ±0.180**
Lap. on abs. coord.	0.467 ±0.075	0.958 ±0.150	0.444 ±0.065	0.952 ±0.140	0.414 ±0.050	1.102 ±0.182	0.425 ±0.050	1.057 ±0.178
MLP deform	0.538 ±0.062	1.237 ±0.195	0.542 ±0.057	1.228 ±0.181	0.533 ±0.057	1.472 ±0.227	0.566 ±0.055	1.447 ±0.218

In the subsequent sections, we present a comparative analysis of Vox2Cortex in relation to other relevant approaches across multiple datasets. Furthermore, we conduct an extensive ablation study to assess the individual contributions of the main building blocks of Vox2Cortex.

11.4.2 Results

Ablation Study

We focus on evaluating the design choices made in Vox2Cortex and examine the characteristics of template deformation approaches, which include the choice of the initial template, our proposed inter-mesh neighbor features, the proposed curvature weighted chamfer loss, and our choice of regularizing the deformation field instead of vertex coordinates. One of the key questions raised by combining a CNN and a GNN is how to transfer information in the form of features from one subnetwork to the other. Existing literature does not extensively study the impact of this and other choices. To address this, we train multiple models that are created as described below and compare their final performance in terms of Average Symmetric Surface Distance (ASSD) and 90th-percentile Hausdorff Distance (HD-90) with respect to the FreeSurfer pseudo ground truth labels in Table 11.3.

Voxel2Mesh: In this variation, we adopt the Voxel2Mesh network [174] as our architecture replacement. Notable differences from Vox2Cortex are as follows: we exclusively sample the CNN features from the voxel-decoder stage at each mesh deformation step rather than sampling from both the encoder and decoder. To ensure a fair comparison, we double the number of voxel-decoder channels at each stage, as these channels are the only ones passed to the GNN in Voxel2Mesh. This adjustment ensures that the length of the feature vectors passed to the GNN remains consistent between Vox2Cortex and Voxel2Mesh. Additionally, we employ learned neighborhood sampling instead of trilinear interpolation at the vertex locations, following the approach described in [174].
Encoder features: In this setup, as shown in works such as [79, 80], we only sample CNN features from the relevant voxel-encoder step at each mesh-deformation stage.
Classic Chamfer: Here, we train our architecture using the classic Chamfer loss, which is equivalent to setting $\kappa(\cdot) \equiv 1$ in Equation 11.6.

W/o inter-mesh NNs means that the white and pial surfaces do not share their nearest-neighbor vertex locations.
Ellipsoidal template: Instead of starting the deformation process with our smoothed cortical template, we use an ellipsoidal template.
W/o voxel decoder: In this configuration, we skip the voxel decoder altogether and instead draw CNN features directly from the encoder at each level, as in [168].
Lap. on abs. coord.: Here, the Laplacian loss is computed based on absolute vertex coordinates rather than relative displacements. In other words, we smooth the mesh itself instead of focusing on the displacement field.
MLP deform: In this scenario, each layer of the GNN is replaced with a linear layer. This setup allows us to evaluate the choice of using a GNN for mesh deformation.

The detailed results of our ablation study can be found in Table 11.3. Notably, we observe that despite being designed for medical applications, Voxel2Mesh [174] does not yield cortical surfaces with sufficient accuracy. Our study shows that the combination of design choices in Vox2Cortex results in the best performance in terms of ASSD for all four surfaces. HD-90 is lowest for Vox2Cortex when it comes to white surfaces, while for pial surfaces, the model without inter-mesh neighbor features and the model without a voxel decoder outperform. As these models only excel over Vox2Cortex in a single mesh for one metric, we argue that the Vox2Cortex architecture is more robust across surfaces, making it a superior choice.

Among Vox2Cortex's design choices, the curvature-weighted Chamfer loss is the most critical factor, as the reconstruction accuracy declines most when replaced by the standard Chamfer loss. When examining the voxel network design, it's interesting to observe that omitting the voxel decoder entirely has minimal impact on the overall performance. However, the performance slightly decreases when sampling the features only from the encoder. One possible explanation is that when the network only has an encoder, it learns features more relevant to template deformation. In contrast, when it has a decoder and segmentation output, the encoder focuses more on learning features relevant to segmentation.

Interestingly, sampling image features from the encoder or decoder does not seem overly important. This could explain why both approaches have been successful in previous work [79, 174]. When using an ellipsoid template instead of our smoothed brain template, the performance declines, validating our choice of a template closer to the desired shape. Applying the Laplacian regularization on vertex coordinates rather than the displacement field results in poorer performance, which might oversmooth the mesh and decrease geometric accuracy. Replacing the graph convolutional layers with linear layers also leads to worse performance. This outcome not only supports the choice of a GNN in Vox2Cortex and the importance of using local neighborhood information even in a fixed mesh but also differentiates our novel recombination of a CNN and a GNN from previous MLP-based approaches [51, 170]. We also observe that it is possible to use a template with lower resolution, i.e., fewer vertices during test time. This leads to slightly lower performance but comes with the benefit of faster processing time and lower memory demand and, therefore, might be of value for some users.

Table 11.4. On the $\text{ADNI}_{\text{large}}$ and OASIS datasets, we compare the performance of Vox2Cortex, DeepCSR, and nnU-Net over all four surfaces. We report Average Symmetric Surface Distance (ASSD) and 90th-percentile Hausdorff Distance (HD-90) in millimeters (mm).

		Left WM Surface		Right WM Surface		Left Pial Surface		Right Pial Surface	
Data	Method	ASSD	HD-90	ASSD	HD-90	ASSD	HD-90	ASSD	HD-90
ADNI large	Vox2Cortex	**0.345** ±0.056	**0.720** ±0.125	**0.347** ±0.046	**0.720** ±0.087	**0.327** ±0.031	**0.755** ±0.102	**0.318** ±0.029	**0.781** ±0.102
	DeepCSR [21]	0.422 ±0.058	0.852 ±0.134	0.420 ±0.058	0.880 ±0.156	0.454 ±0.059	0.927 ±0.243	0.422 ±0.053	0.890 ±0.197
	nnU-Net [68]	1.176 ±0.345	1.801 ±2.835	1.159 ±0.242	1.739 ±1.880	1.310 ±0.292	3.152 ±2.374	1.317 ±0.312	3.295 ±2.387
OASIS	Vox2Cortex	**0.315** ±0.039	**0.680** ±0.137	**0.318** ±0.048	0.682 ±0.151	**0.362** ±0.036	**0.894** ±0.141	**0.373** ±0.041	**0.916** ±0.137
	DeepCSR [21]	0.360 ±0.042	0.731 ±0.104	0.335 ±0.050	**0.670** ±0.195	0.458 ±0.056	1.044 ±0.290	0.442 ±0.058	1.037 ±0.294

Comparison with Related Work

We compared Vox2Cortex and DeepCSR [21] on the $\text{ADNI}_{\text{large}}$ and OASIS datasets. For DeepCSR, we obtained high-resolution predictions by sampling points at a distance of 0.5mm. To assess the performance against a voxel-based segmentation method, we selected 3D nnU-Net [68], a state-of-the-art segmentation model known for its effectiveness across various segmentation tasks.

We follow the approach by Santa Cruz et al. [21] to extract mesh representations of the cortical surfaces from the nnU-Net segmentation. Here we apply topology correction and marching cubes [87] to obtain topologically correct meshes that enable a fair comparison to other methods. As illustrated in Table 11.4, nnU-Net achieves mesh predictions that are less accurate compared to the other methods. This was expected as nnU-Net is a voxel based method and therefore the resolution of the generated meshes is dependent on the image resolution. As the image resolution is 1mm cubic, we can not expect the surface distance to get below 1mm. Moreover, staircase artifacts in the meshes resulting from voxel-based segmentation and marching cubes extraction contribute to higher surface distances when comparing the meshes to the smoother FreeSurfer ground truth meshes. Vox2Cortex outperforms DeepCSR in nearly all performance metrics across both datasets.

We also compared our method's predictions to DeepCSR on the OASIS test set, both with and without topology adjustment (see Table 11.5 for details). The starting template for Vox2Cortex determines the maximum allowed number of faces and vertices for the meshes to contain. Without topology correction, DeepCSR meshes contain multiple connected components, which results in a higher genus than zero. After topology correction, the topology becomes spherical, as expected. The number of vertices and faces of DeepCSR meshes is much larger compared to Vox2Cortex, which is due to the small grid of 0.5mm. However, our results in Table 11.4 demonstrate that the number of vertices and faces of Vox2Cortex meshes is sufficient and even leads to higher surface accuracy. As we visually display in Figure 11.5, topology correction might lead to geometric errors.

Consistency

In this experiment, we evaluate the robustness and consistency of our method. We use models trained on $\text{ADNI}_{\text{large}}$ and test them to the Test-Retest dataset (TRT) dataset [98]. We predict the cortical surfaces of the two brain scans from the same day and compare them in terms of ASSD and HD-90 to evaluate the consistency of the methods.

Table 11.5. In this table, we compare our Vox2Cortex to a state-of-the-art method DeepCSR [21], in terms of mesh complexity, as assessed by the number of faces and vertices, and topological metrics, such as the number of connected components (CC) and the genus.

	Pial Surfaces				WM Surfaces			
Method	CC	genus	# faces	# vertices	CC	genus	# faces	# vertices
Vox2Cortex	1	0	336112	168058	1	0	336112	168058
DeepCSR	48.6	152.4	1341838.3	670711.5	18.3	15.8	1209313.5	604661.7
DeepCSR + top. corr.	1	0	1291385.5	645694.8	1	0	1160980.8	580492.4

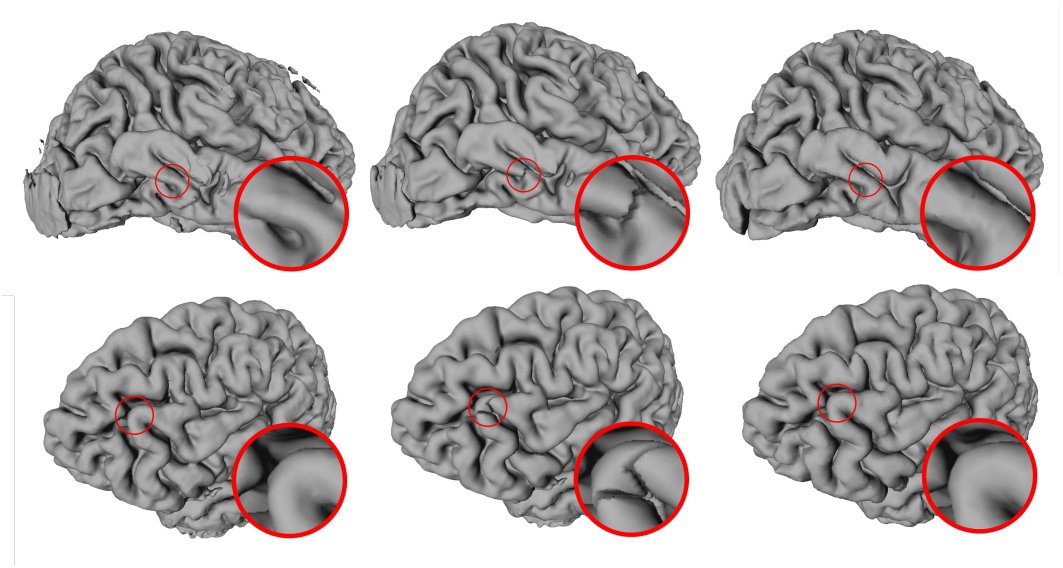

Figure 11.5. Visualizations depicting incorrect anatomy resulting from topology correction on DeepCSR [21] meshes. The images display pial surfaces from two distinct patients from the OASIS dataset. On the left, we showcase DeepCSR's prediction before topology correction. In the middle, we display the outcome after applying topology correction. On the right, we provide the FreeSurfer pseudo ground truth for reference. The zoomed-in regions show how the topology correction introduced cracks in a fold, which is anatomically not plausible. © IEEE,2022

Reconstructions should be similar, with occasional differences attributable to the imaging method, because brain morphology does not vary across scans on the same day. In this experiment, we utilize Iterative Closest-Point algorithm (ICP) to align images before doing comparisons, as recommended in [21]. Vox2Cortex outperforms DeepCSR and FreeSurfer regarding repeatability, as shown by the data in Table 11.6. This experiment demonstrates that Vox2Cortex can generalize effectively to unknown data, as the models were not trained on the TRT dataset. When we look into how quickly each approach can predict a target, we discover that Vox2Cortex is around 25 times quicker than DeepCSR.

Cortical Thickness

Surface reconstruction is crucial in determining cortical thickness, an important biomarker for assessing cortical atrophy in neurodegenerative diseases. In this experiment, we utilize Vox2Cortex surfaces to derive cortical thickness measurements and compare them to the thickness measurements obtained from FreeSurfer. The thickness calculation involves finding the closest-point correspondences between the WM and pial surfaces [107]. On the OASIS test set, the median cortical thickness error per vertex, compared to the closest vertices from

Table 11.6. Comparison of Vox2Cortex with DeepCSR and FreeSurfer in terms of reconstruction consistency using the ASSD and HD-90 on the TRT dataset. Additionally, we provide the percentage of points with a surface distance over 1 and 2mmm and the inference time for each method per 3D scan. We highlight the best performing methods for each metric in bold font. An asterisk (*) denotes the utilization of smaller templates, with approximately 42,000 vertices per surface instead of the usual approximately 168,000 vertices.

Method	ASSD (mm)	HD-90 (mm)	$>$ 1mm	$>$ 2mm	Inference time
Vox2Cortex	0.228 ±0.048	0.478 ±0.101	0.80%	**0.06%**	18.0s
Vox2Cortex*	**0.225** ±0.049	**0.471** ±0.103	**0.77%**	**0.06%**	2.1s
DeepCSR	0.357 ±0.284	0.739 ±0.595	5.82%	2.23%	445.7s
FreeSurfer	0.291 ±0.133	0.605 ±0.279	2.87%	0.67%	>4h

FreeSurfer, is 0.305mm [lower quartile: 0.140mm, upper quartile: 0.564mm]. To provide context, regions affected by atrophy in Alzheimer's disease often exhibit thickness reductions of one millimeter or more. As shown in Figure 11.6, the thickness measurements on Vox2Cortex meshes closely align with those obtained from FreeSurfer pseudo-ground-truth meshes, further validating the accuracy of Vox2Cortex.

Using a group analysis, we examined the vertices with reduced cortical thickness in people with Alzheimer's disease (n=50) and healthy controls (n=124) in the $\text{ADNI}_{\text{large}}$ test-split. Our surfaces were registered to the FreeSurfer Fsaverage template so that all thickness

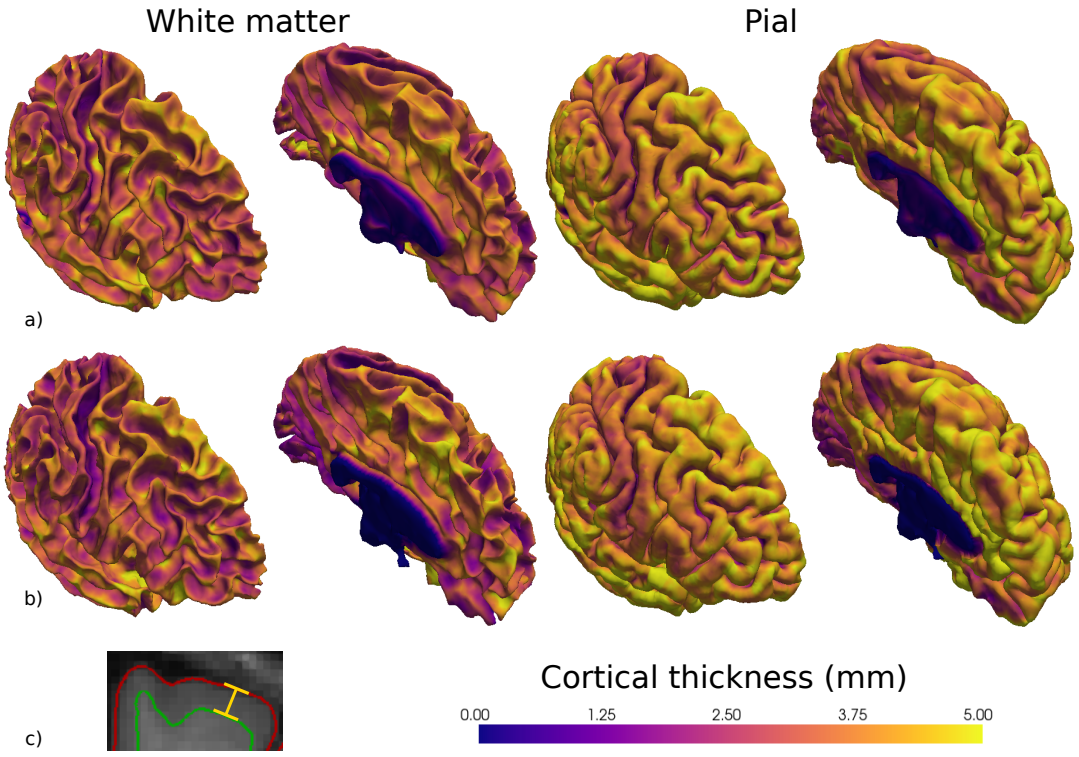

Figure 11.6. Predicted meshes of an example subject from the OASIS dataset, with each vertex color-coded to represent cortical thickness in millimeters. Row a) showcases the Vox2Cortex meshes, while row b) displays the FreeSurfer meshes. In row c), the cortical thickness is visualized between the white matter (green) and pial (red) surfaces. © IEEE,2022

Table 11.7. Results of AD classification (95% confidence interval bootstrapped) based on cortical thickness biomarkers.

	balanced acc.			roc-auc			avg. prec.		
	est.	low	high	est.	low	high	est.	low	high
Vox2Cortex	0.812	0.737	0.875	0.890	0.798	0.940	0.840	0.722	0.906
FreeSurfer	0.816	0.740	0.874	0.915	0.860	0.950	0.825	0.697	0.896
DeepCSR	0.787	0.717	0.847	0.838	0.752	0.897	0.705	0.554	0.809

measurements would be in the same domain, allowing for a more reliable comparison. Cortical thickness was computed using FreeSurfer meshes based on closest-point correspondences rather than the thickness values provided by FreeSurfer. Figure 11.7 displays the p-values (one-sided t-test) obtained for both hemispheres. Vox2Cortex is suited for group analysis of cortical thickness, as seen by the striking resemblance between the significance maps created by Vox2Cortex and FreeSurfer.

We were further interested if thickness measurements from our predicted surfaces can be used for Alzheimer's disease prediction and how it compares to FreeSurfer and DeepCSR. For this, we take the per-vertex thickness measurements and use PCA to reduce the dimension to 64. We then trained a gradient-boosted regression tree to classify AD vs. healthy control subjects. Table 11.7 shows the classification results, where Vox2Cortex achieves comparable results to FS and outperforms DeepCSR. This experiment shows that our surfaces are geometrically accurate, and biomarkers derived from our meshes can be used for clinically relevant tasks.

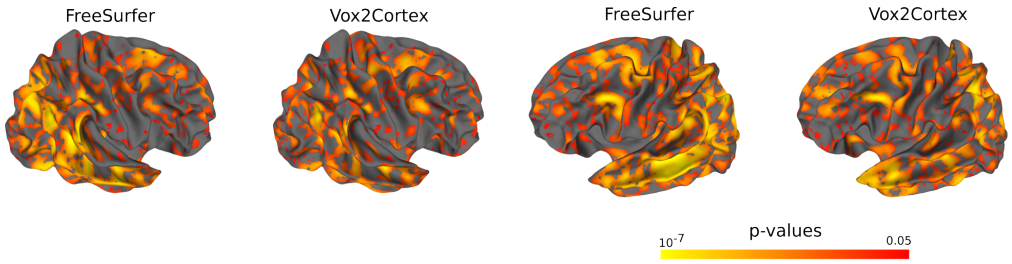

Figure 11.7. Cortical atrophy in the right and left hemispheres: a group comparison using the $\text{ADNI}_{\text{large}}$ test-split of patients with Alzheimer's disease and healthy controls. P-values obtained from one-sided t-tests are visualized on the FsAverage meshes, providing insights into the statistical significance at each vertex. Figure adapted from figures in [10] © IEEE,2022

Visual Analysis of FreeSurfer Fails

Within our $\text{ADNI}_{\text{large}}$ dataset, we carefully excluded cases where FreeSurfer encountered failures during the surface pipeline. For the lack of automatic quality control for the surface stream, we follow the hypothethis that errors in surface reconstruction are related to errors in the voxel-based segmentation stream. Therefore we remove all scans that exhibited segmentation failures in one or more structures, as determined by the UCSF quality control guidelines [55]. As the voxel segmentation often fails in cases where contrast between structures is low or general image quality is low, we believe this will translate well to the surface stream. Subsequently, we applied our trained model to the cases where FreeSurfer failed, which we had previously

excluded from the training and testing sets. To enhance visibility, we focused on visualizing the results for the pial surfaces in Figure 11.8. In the first case, which exhibited relatively mild issues, FreeSurfer generated four surfaces; however, the left pial surface extended into the dura, resulting in artifacts. Our model, on the other hand, successfully avoided these artifacts. Additionally, we present a more extreme case in which FreeSurfer failed to generate surfaces for the right hemisphere and incorrectly segmented parts of the left temporal lobe.

This evaluation is limited since we only visually inspected two subjects. For instance, a large-scale comparison between Vox2Cortex and FreeSurfer on a dataset like the UK Biobank would be interesting. However, to the best of our knowledge, no automatic quality control algorithms are available for such an analysis.

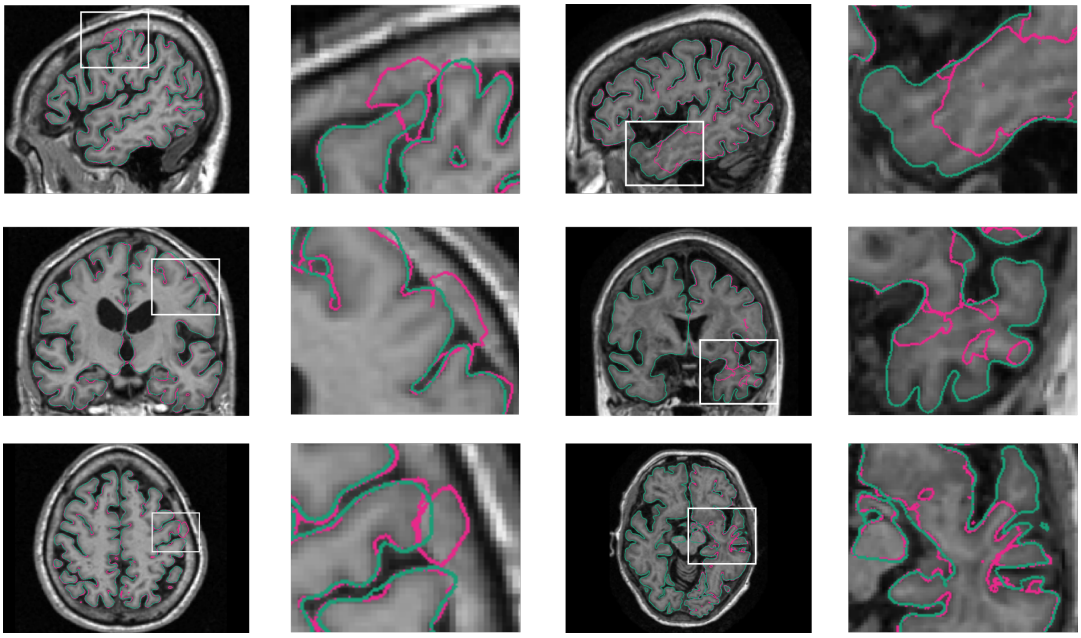

Figure 11.8. The MRI scans in this image show the pial surfaces created by Vox2Cortex, shown in green and FreeSurfer, displayed in pink. The images are arranged from top to bottom, including two patients' sagittal, coronal, and axial slices. We notably emphasize regions where FreeSurfer failed in the zoomed-in parts. For the first subject (left), an observation can be made regarding the FreeSurfer pial mesh (pink) extending into the dura. In the second example (right), FreeSurfer failed to generate a mesh for the right hemisphere and exhibited numerous errors in the temporal lobe of the left hemisphere. In comparison, Vox2Cortex demonstrates superior visual performance in these particular cases, with fewer artifacts and inaccuracies.

11.5 Limitations and Potential Negative Impact

It should be noted that the findings shown here are the raw output of our model. While we can make predictions, we can't guarantee that the resulting meshes won't have self-intersections, which might be essential in some cases. But techniques like MeshFix [3] can fix the problem of self-intersections. Furthermore, it should be noted that the absence of manually generated ground-truth surfaces for our dataset led us to employ FreeSurfer surfaces as a pseudo-ground-truth, which is a common practice in the field [21, 97]. Although efforts were made to

eliminate noisy labels from our training set, there may still be some residual noise in the labels.

Importantly, these predictions should not be used to make clinical decisions, even though our technique can assist radiologists in observing brain surfaces and rapidly computing metrics such as cortical thickness. Since our model has only been evaluated using the datasets described in this chapter, we cannot guarantee its performance with previously unseen data or data from other fields. In addition, the data analyzed in this study only include healthy subjects and those with dementia, so the trained model may generate inaccurate predictions when presented with other morphological brain alterations (e.g., tumors).

We attempted to balance our data splits concerning patient age and sex to minimize bias. However, our model may still lack fairness and potentially discriminate against underrepresented groups within the given datasets. Therefore, expanding the diversity of the training data and addressing potential biases could help mitigate this limitation. Further, the combination of reconstruction losses in this work requires tedious tuning of several loss weights. Therefore, exploring ways to reduce the number of loss hyperparameters in future work would be interesting, which will be addressed in Chapter 13.

Although Vox2Cortex could potentially replace FreeSurfer's time-consuming surface reconstruction, it does not have the full functionality of FreeSurfer's surface stream. For example, FreeSurfer provides parcellations of the surface and registration to the FsAverage template, which is useful for group comparisons and visualization. The next two chapters will tackle these problems.

11.6 Conclusion

In this chapter, we introduced Vox2Cortex, a new approach to reconstructing white matter and pial surfaces from brain MR images in a single step. Our method incorporates a number of novel elements, including a generic brain template that is iteratively deformed by combining convolutional and graph convolutional neural networks. Using a unique curvature-weighted Chamfer loss function, we successfully included ground-truth curvatures, allowing for accurate reconstructions, even in heavily folded locations. This loss function might be useful in a number of fields where complex 3D geometries are common, in addition to the medical industry. Extensive experiments show that our suggested combination of loss functions permits the development of state-of-the-art cortical surfaces utilizing an end-to-end trainable architecture while drastically reducing computing time. In addition, we have demonstrated our method's ability to generate accurate cortical thickness maps, which facilitates the study of atrophy in Alzheimer's disease.

Overall, Vox2Cortex represents a significant advancement in brain surface reconstruction, offering a comprehensive and efficient solution that holds promise for medical and non-medical applications involving complex 3D geometries. The end-to-end trainable architecture and efficient computation time make it a powerful tool for generating state-of-the-art cortical surfaces.

Looking ahead, the next chapter Chapter 12 focuses on learning joint parcellation and surface reconstruction. This integration enables comparisons of thickness measures at a regional level, providing valuable insights for studies such as the analysis of atrophy in Alzheimer's disease. This approach enhances the interpretability of group-level analyses by considering parcels instead of individual vertices. In the subsequent chapter Chapter 13, we address the challenge of vertex correspondence between the input template and the output reconstructed surface. By eliminating the need for registration to a group template, our method simplifies group studies and facilitates direct comparisons between cortical surfaces.

12 Joint Cortical Surface Reconstruction and Parcellation

Contents

12.1 Introduction

In the last chapter Chapter 11, we have presented Vox2Cortex (V2C), a deep learning technique for cortical surface reconstruction. It was one of the first of these methods, along with [21, 84, 97], offering significant speed improvements over FreeSurfer by reconstructing cortical surfaces from Magnetic Resonance Imaging (MRI) scans in a matter of seconds. However, these methods cannot currently provide surface parcellations. This chapter will discuss our publication 'Joint cortical surface reconstruction and parcellation' [132], proposing two methods for joint parcellation and surface reconstruction extending our previously discussed Vox2Cortex network.

Modern parcellation methods [22, 47] typically depend on FreeSurfer for surface mesh extraction. An exception is SegRecon [48], but it falls short regarding surface accuracy.

In this chapter, we bridge this gap by integrating two distinct parcellation techniques in a way that permits end-to-end training into two state-of-the-art methods of cortical surface reconstruction (CSR): our own Vox2Cortex [10], that was introduced in the previous Chapter 11, and the simultaneously released CorticalFlow [84]. Specifically, we use a graph classification network to improve CSR networks. Alternatively, we use a novel class-based reconstruction loss to pass template parcellation labels through the CSR network. Figure 12.1 illustrates both of these methodologies. Our results show that using these methods produces state-of-the-art cortical surfaces and accurate segmentations of the cerebral cortex.

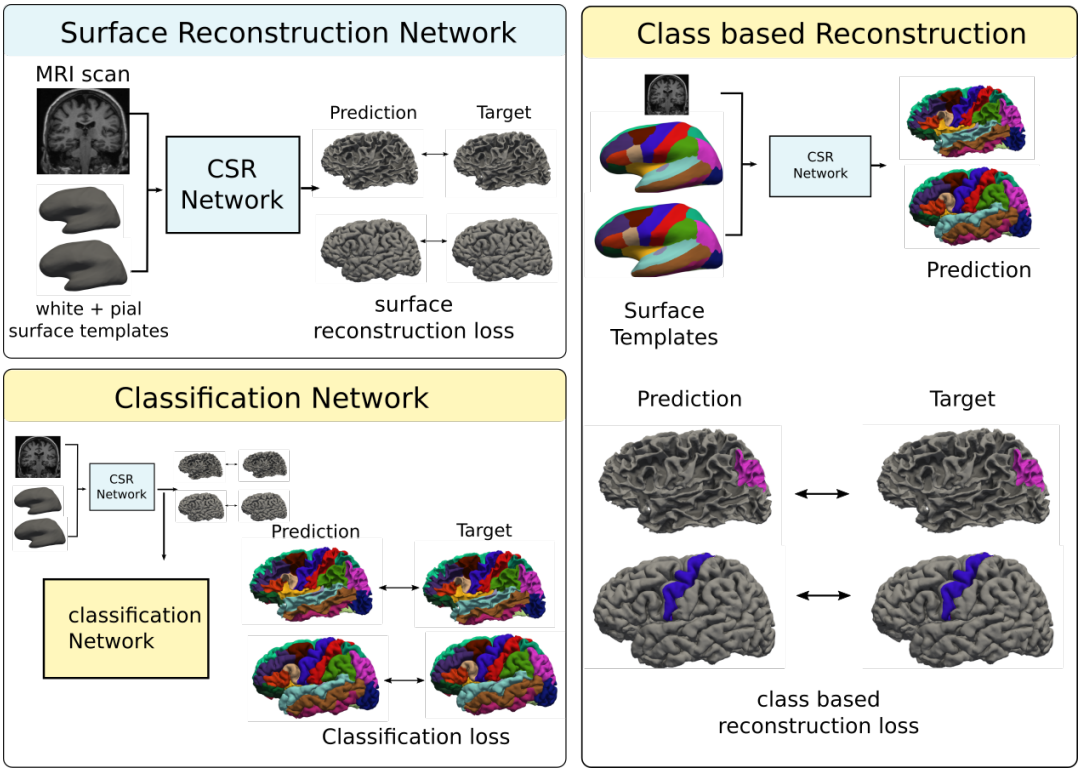

Figure 12.1. Here, we show our improvements to cortical surface reconstruction (CSR) networks for cortical parcellation schematically. Top left: Typical CSR network that creates white matter and pial meshes based on input MRI scan and template meshes. It is trained with a surface reconstruction loss like the chamfer loss. Bottom left: after the CSR network, we add an additional classification network, that predicts a class for each vertex of the generated mesh. We employ an additional classification loss to train the classification network. Right: Here we use as input a template mesh with vertex features that represent the parcellation class. These labels are passed throughout the mesh deformation and instead of computing a global surface reconstruction loss, we compare vertices of the same class only in a class based reconstruction loss. © Springer,2022

12.2 Related Work

This section briefly reviews past work on cortical parcellation and surface reconstruction. Although we focus on joint reconstruction and parcellation, most existing methods address only one of these tasks at a time.

Fully-Convolutional Neural Networks (F-CNNs) have seen extensive use in medical image segmentation and have demonstrated success in cortex parcellation. FastSurfer [59] replaces FreeSurfer's voxel-based stream with a multi-view 2D F-CNN. Other methods [20, 66] employ 3D patch-based networks. However, applying fully-convolutional segmentation networks is ultimately constrained by the input MRI scans' image resolution, and combining them with FreeSurfer's surface stream is inefficient in inference time.

Deep learning techniques for parcellation that work on given surface meshes have been introduced. For instance, [22] investigates a range of network architectures for segmenting two brain regions, concluding that approaches based on graph convolutions are more apt than multi-layer perceptrons (MLPs). Similarly, [34] employs graph attention networks for

parcellating the whole cortex. Taking a different approach, [47] utilizes spherical graph convolutions, which they found to be superior to the traditional graph convolutions in the Euclidean domain. It is important to note that all these vertex classifiers operate assuming that the surface mesh is already available.

In contrast to implicit methods, template-deformation methods like Vox2Cortex [10] provide explicit cortical surface reconstruction without the need for post-processing. CorticalFlow [84] (CF) is another template-deformation method developed concurrently with Vox2Cortex. Unlike Vox2Cortex's approach to predicting the mesh-deformation field on a per-vertex basis, CF generates a deformation field in image space and maps it onto mesh vertices through interpolation. The deformation process in CF is performed incrementally, using a 3D UNet to predict each sub-deformation. The authors propose an Euler integration scheme for the flow fields to avoid self-intersections. The rationale behind numerical integration is that mesh deformation is guaranteed to be diffeomorphic and intersection-free by selecting a small step size. However, this guarantee is not always achieved due to surface discretization[84].

SegRecon [48] stands alone as a method that concurrently learns the generation of cortical surfaces along with a specialized parcellation. The authors utilized a 3D U-Net for training on a voxel-based Signed Distance Function (SDF) of the interface between white and gray matter and spherical coordinates in atlas space. The parcellation from the atlas may be aligned with the surfaces once the mesh is extracted and the topology is corrected. Although this technique can extract a white matter surface from an MRI (with a reported Hausdorff distance of 1.3 mm), its major focus is parcellation. Consequently, the surface reconstructions it produces don't quite match up to recent algorithms that are tailored specifically for this objective, like Vox2Cortex [10] or CorticalFlow [84]. For a more in-depth discussion regarding related implicit and template deformation techniques for cortical surface reconstruction, refer to the preceding Chapter 11.

12.3 Method

Following recent progress in this area, we suggest extending current approaches to supply the reconstructed surfaces with a jointly learned parcellation. In particular, we build on our previous technique, Vox2Cortex [10], and CorticalFlow [84], both of which are template deformation approaches that have shown state-of-the-art results in extracting cortical surfaces. The white matter and pial surfaces of both hemispheres are computed concurrently using each technique, which takes as inputs a 3D MRI scan and a mesh template.

Multiple atlases are available for parcellation of the human cortex based on various criteria like structural or functional attributes that distinguish different brain regions. Atlases that are widely adopted include the Desikan-Killiany-Tourville (DKT) [27, 78] and Destrieux [28] atlases, both of which can be accessed in FreeSurfer. Our work uses FreeSurfer surfaces as pseudo-ground-truth meshes and parcellation labels derived from the DKT atlas. Additionally, we employ smoothed versions of the FreeSurfer FsAverage template as mesh input for the networks for cortex reconstruction. Compared to Vox2Cortex [10], where we used a smoothed mesh of a random training subject as input, the FsAverage template is more generic, as it

represents an average surface of 40 subjects [38]. In the following, we describe the two proposed extensions of cortical reconstruction networks that we evaluate in this chapter.

12.3.1 Classification Network:

Recent work [22, 34] has showcased the capability of Graph Neural Network (GNN) oriented classification networks in achieving precise cortex parcellations. Building on this, we introduce an enhancement to the CSR networks by incorporating a classification arm. This arm is made up of three residual GNN blocks, with each block comprising three GNN layers. The classification network takes the predicted mesh with vertex features obtained from Vox2Cortex's GNN or simply the vertex coordinates sourced from CorticalFlow, as its input. For every vertex, the output is a vector containing each class's probabilities. We then apply a softmax layer and compute a cross-entropy loss by comparing the predicted classes to the ground-truth classes for the nearest points in the target mesh.

In the case of Vox2Cortex, the classification network is integrated following the final mesh deformation stage. Both the CSR and classification networks are trained in an end-to-end fashion. When combined with CorticalFlow, the classification layer is appended after the final deformation, but the parameters of the U-Nets from prior stages are kept constant. Through experimentation, we found that incorporating the classification network at every step of the iterative optimization introduced instability in training. Consequently, we opted to include the classification network in the final iteration.

12.3.2 Class-Based Reconstruction:

Both CSR methods of interest, Vox2Cortex, and CorticalFlow, are template-deformation techniques, meaning they start from a given template mesh that they iteratively deform. Therefore, we propose a method that starts from a template with parcellation labels and passes these labels through the deformation procedure. Upon deformation, it is critical to ascertain that the vertices align with analogous anatomical locations in the initial template. In essence, this necessitates the establishment of vertex correspondence between the input template and the resultant reconstructed surface. To this end, we advocate the utilization of a class-based reconstruction loss function. Notably, the loss function is insensitive to the specific choice of the reconstruction loss, such as Chamfer distance in CorticalFlow or a combination of curvature-weighted Chamfer distance and normal distance in Vox2Cortex.

Adhering to the notation introduced in the preceding Chapter 11, let's represent the predicted and actual point sets, which are sampled from the meshes $\hat{M}_{s,c}$ and M_c (potentially inclusive of normals), by $\hat{\mathcal{P}}_{s,c}$ and $\mathcal{P}_c$ respectively. Moreover, let $\mathcal{L}_{\text{rec}}(\hat{\mathcal{P}}_{s,c,p}, \mathcal{P}_{c,p})$ symbolize any reconstruction loss between the point clouds of a specific parcellation class, denoted by p where $p \in 1, \ldots, C_p$. The class-based reconstruction loss is then calculated as follows:

$$\mathcal{L}_{\text{rec},class}(\hat{\mathcal{P}}_{s,c}, \mathcal{P}_c) = \frac{1}{|C_p|} \sum_{p=1}^{C_p} \mathcal{L}_{\text{rec}}(\hat{\mathcal{P}}_{s,c,p}, P_{c,p}). \tag{12.1}$$

In essence, the predicted points corresponding to a particular parcellation class are exclusively compared to the ground-truth points that belong to that same class. This concept is also visually represented in Figure 12.1B. By designing this loss function, we align the parcellation of the deformed template and the ground-truth parcellation. A notable advantage of this method over the classification network is that it precludes the emergence of "islands," misclassified diminutive regions, on the reconstructed meshes.

12.4 Experiments & Results

12.4.1 Setup and Hyperparameters

Our models are trained using the OASIS dataset, adhering to the same preprocessing steps outlined in Section 10.6. A singular deviation in the experimental setup compared to Vox2Cortex is utilizing an updated version, v7.2, of the FreeSurfer software pipeline. This updated version is a silver standard in training and evaluating our models. For the computation of reconstruction losses, we employ a differentiable sampling method to acquire 50,000 points from the predicted and ground-truth meshes [46]. In the case of CorticalFlow, the training adheres to the iterative methodology detailed by [84], which entails keeping the parameters of the UNet(s) from steps 1 to $i-1$ fixed while training the UNet at step i. Additionally, we leverage the AdamW optimizer [95] with an exponential weight decay of $1\text{e}{-4}$, combined with a cyclic learning rate schedule [155] for network optimization.

Table 12.1. List of hyperparameters. We always trained until the performance converged on the respective validation sets, unless for CorticalFlow experiments where we trained the first two UNets for a fixed number of 50 epochs.

All experiments			
Optimizer	Lr scheduler	Weight decay	Mixed precision
AdamW [95]	Cyclic [155]	1e−4	Yes

Graph classifier				
Base lr	Max lr	Channels	Loss weight (V2C)	Loss weight (CF)
2.5e−5	2e−4	$(64) \times 3$	1e−2	1e−4

Vox2Cortex								
Base lr UNet	Max lr UNet	Base lr GNN	Max lr GNN	Template size training	Template size test	Loss weights	Channels UNet	Channels graph net
1e−4	4e−4	5e−5	2e−4	41K	164K	see [10]	16,32,64,128,256, 128,64,32,16	$((64) \times 3) \times 4$

Cortical Flow					
Base lr	Max lr	Template sizes	Chamfer weight	Edge weight	Channels per UNet
1e−4	4e−4	41K 164K 164K	1	1	16,32,64,128,256,128,64,32,16 16,32,64,32,16 16,32,64,32,16

As the input source for the deformation networks, we make use of the FsAverage templates derived from FreeSurfer, which are subjected to intensive smoothing via the HC Laplacian smoothing operator accessible in MeshLab [19]. Table 12.1 encompasses a comprehensive list of model parameters, which have been inherited from the Vox2Cortex and CorticalFlow studies. Our code is based on PyTorch [123] and PyTorch3D [130], and the training regimen was executed on an Nvidia Titan RTX GPU.

12.4.2 Results

We show findings for the classification network and class-based reconstruction parcellation methods suggested here by integrating them with V2C and CorticalFlow (CF) as described in the Section 12.3. This results in a total of four methods, labeled as V2C_C, CF_C (classification network), and V2C_T, CF_T (class-based reconstruction via template). We evaluate each reconstruction technique against FastSurfer [59] and two additional baselines.

As a first additional baseline, we train CF and V2C and assume vertex correspondence between the input template and the predictions and map the atlas labels directly to the predicted surfaces (labeled as CF + atlas and V2C + atlas). This baseline serves as an evaluation of the vertex-wise alignment between predicted meshes and ground truth. If each template vertex corresponds to the same anatomical location as the predicted vertex, a direct mapping of the templates' parcellation labels should also yield a correct parcellation of the predicted mesh. This feature is also referred to as vertex correspondence and, in more detail, is addressed in Chapter 13. As a post-processing step, the second additional baseline uses FreeSurfer's spherical registration to register the predicted meshes to FsAverage. Then it maps the FsAverage parcellation labels to the registered meshes. This approach is called CF + FS and V2C + FS). Table 12.2 presents the parcellation accuracy in terms of the average Dice coefficient over all parcellation classes and the surface reconstruction accuracy in terms of Average Symmetric Surface Distance (ASSD) and 90th-percentile Hausdorff Distance (HD-90) in mm.

FastSurfer produces surfaces of remarkable accuracy when compared to the FreeSurfer silver standard. This can be attributed to FastSurfer's use of the FreeSurfer surface pipeline to generate surfaces. The CF and V2C baselines deliver highly precise surface predictions, with CF having a slightly better performance. However, mapping the atlas parcellation directly to the output does not yield an accurate surface parcellation, as we cannot guarantee vertex correspondence between the template and the predictions. However, it is interesting that the mapped parcellations are not completely off, which hints that V2C and CF provide some level of vertex correspondence between the input FsAverage template and the predictions.

Producing the DKT parcellation using FreeSurfer's spherical atlas registration results in an elevated Dice score compared to FastSurfer. We believe this to be due to the superior performance of the mesh-based parcellation compared to an approach based on voxels. It is important to note that FreeSurfer's spherical registration is exclusively designed for white matter surfaces, making it inapplicable for parcellating pial surfaces.

Concerning the quality of surfaces, we notice a degradation in accuracy in the models we propose due to the inclusion of the additional task of cortex parcellation. This reduction is

Table 12.2. Using the OASIS dataset, we assess the surface and parcellation accuracy of our improved Vox2Cortex (V2C) and CorticalFlow (CF) methods. The Average Symmetric Surface Distance (ASSD) and 90th-percentile Hausdorff Distance (HD-90) are two millimeter-based surface reconstruction metrics. All measurements are averaged throughout both hemispheres of the brain and displayed as the mean ± standard deviation over the full dataset.

	White surfaces			Pial surfaces		
	Parcellation	Surface accuracy		Parcellation	Surface accuracy	
Method	Dice↑	ASSD↓	HD-90↓	Dice↑	ASSD↓	HD-90↓
CorticalFlow (CF) [84]						
CF + atlas	0.810 ±0.095	0.244 ±0.040	0.578 ±0.101	0.787 ±0.091	0.302 ±0.039	0.747 ±0.117
CF + FS	0.885 ±0.069	0.244 ±0.040	0.578 ±0.101	n.a.	0.302 ±0.039	0.747 ±0.117
CF_C	0.727 ±0.178	0.471 ±0.047	1.190 ±0.170	0.672 ±0.178	0.355 ±0.040	0.896 ±0.126
CF_T	**0.904** ±0.048	0.323 ±0.048	0.784 ±0.126	**0.877** ±0.049	0.347 ±0.044	0.854 ±0.120
Vox2Cortex (V2C) [10]						
V2C + atlas	0.740 ±0.121	0.282 ±0.034	0.587 ±0.078	0.691 ±0.132	0.341 ±0.037	0.848 ±0.124
V2C + FS	0.876 ±0.076	0.282 ±0.034	0.587 ±0.078	n.a.	0.341 ±0.037	0.848 ±0.124
$V2C_C$	**0.902** ±0.050	0.303 ±0.034	0.641 ±0.082	**0.876** ±0.053	0.362 ±0.038	0.894 ±0.119
$V2C_T$	0.885 ±0.057	0.372 ±0.051	0.823 ±0.108	0.858 ±0.058	0.429 ±0.052	1.066 ±0.182
FastSurfer [59]	0.862 ±0.084	0.138 ±0.057	0.331 ±0.172	0.839 ±0.081	0.240 ±0.065	0.557 ±0.179

especially pronounced in the CF_C and $V2C_T$ models. We hypothesize that the class-based reconstruction loss might clash with the regularizers in V2C, which necessitates several regularization losses. As for the accuracy of parcellation, the best performing methods are the CF_T and $V2C_C$ models, which exhibit an average Dice score surpassing 0.9 for white surfaces and 0.87 for pial surfaces across all parcels. These models also exhibit the least reduction in surface accuracy. The GNN classifier in $V2C_C$ makes use of vertex features from the Vox2Cortex GNN as its input, which are not available in CF_C, where the classification network only receives vertex positions as input. As expected, CF_C achieves a lower parcellation accuracy than $V2C_C$. We infer that combining CF with a GNN classification network is not the most effective.

In Figure 12.2, we visually represent the accuracy of parcellation and surface reconstruction for the left hemisphere's surfaces. Notably, classification errors primarily arise along the borders of the parcels, as observed from the averaged data across the test set. In summary, the GNN classifier appears more compatible with V2C due to the enhanced vertex input features provided by preceding graph convolutions. On the other hand, CF yields superior results when employing the class-based reconstruction loss. We further show detailed per-parcel results of the two best performing methods CF_T and $V2C_C$ in Figure 12.3. Regarding the runtime of the evaluated methods, FastSurfer takes about one hour, the FS parcellation of CF and V2C meshes several hours, and the runtime of the proposed methods is in the range of seconds.

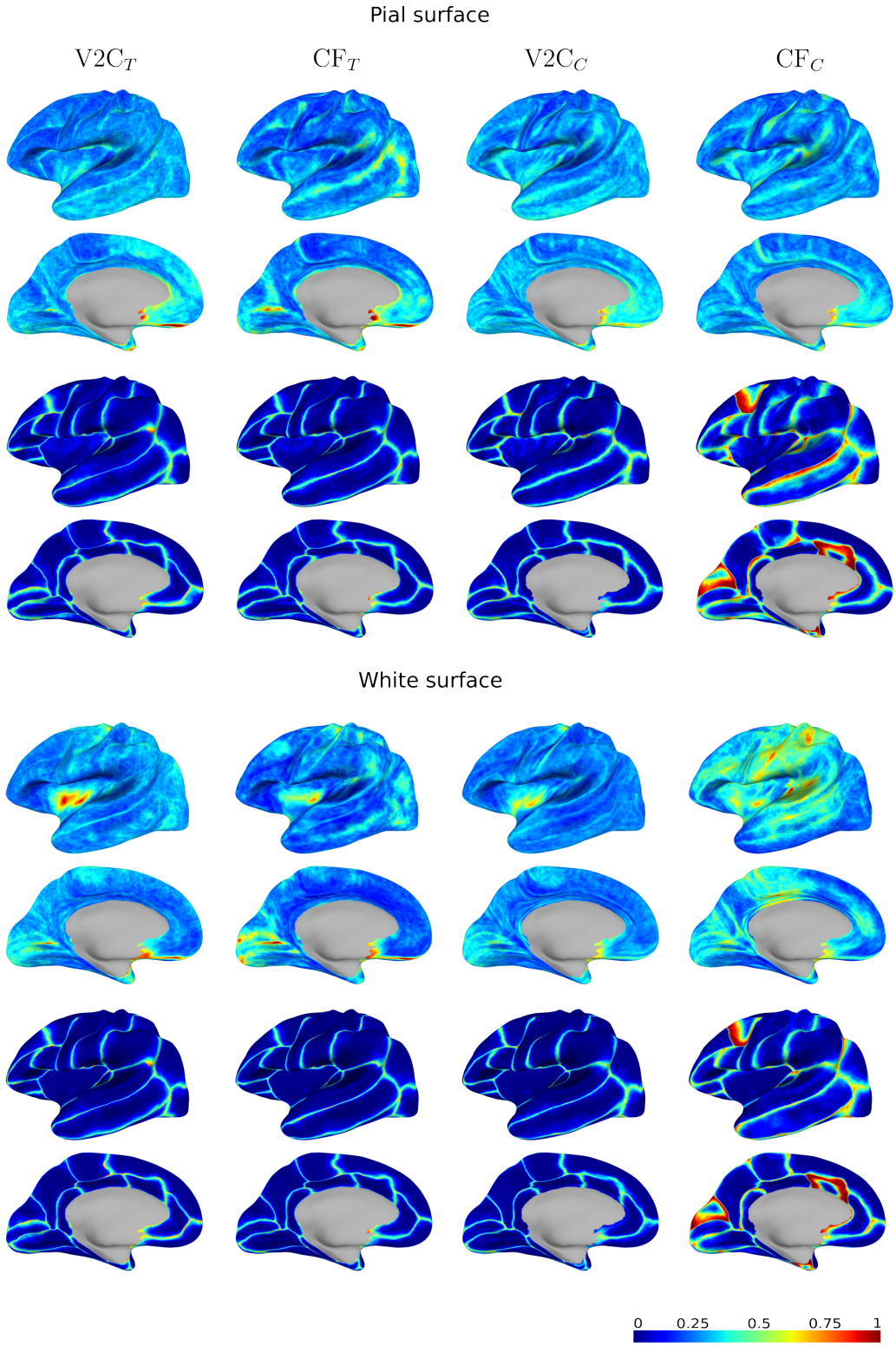

Figure 12.2. Comparison of our four methodologies' parcellation and reconstruction accuracy, averaged across the predicted pial and white matter surfaces in the OASIS test set. The average distance in millimeters is presented in the top rows, depicting how far the predicted surface is from the ground-truth surface. The parcellation error is displayed in the bottom rows, with a value of 0.0 indicating that a vertex is correctly classified for all subjects, and a value of 1.0 indicating that it is wrongly classified for all subjects. The errors are predominantly located at the boundaries of the parcels. © Springer, 2022

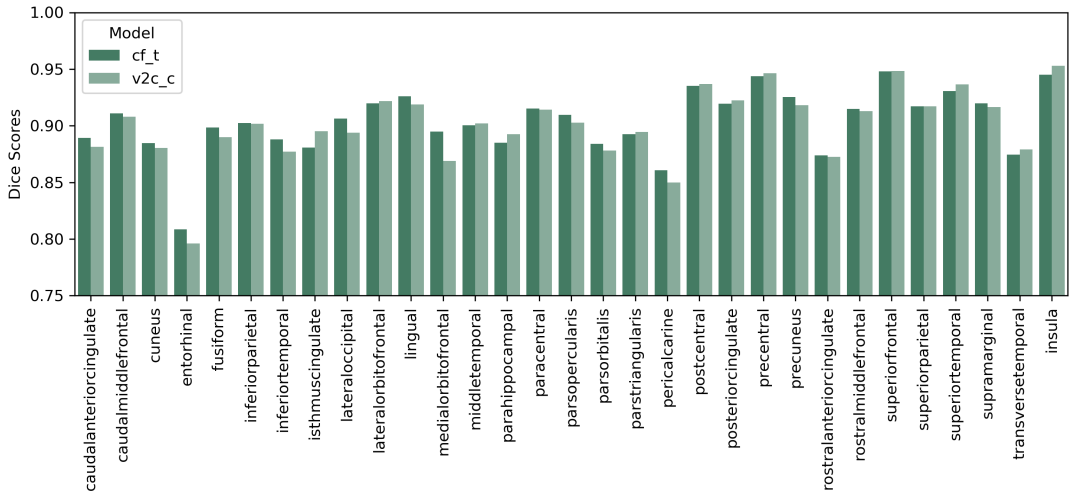

Figure 12.3. Per-class comparison of our best-performing methods CF_T and $V2C_C$ in terms of Dice scores. © Springer, 2022

12.5 Conclusion

In this chapter, we proposed two novel extensions to cortex reconstruction networks that enable joint cortex parcellation. The first extension is based on a graph classifier, and the second one is based on a novel region-based reconstruction loss that is generic and effective. Both methods are well-suited for template deformation networks, which produce accurate surface meshes, allowing for precise parcellation of the surfaces into associated regions. With inference times in the range of seconds, the presented algorithms are high-speed and achieve high parcellation accuracy. These attributes make them ideal for use in large-cohort studies and clinical practice for more fine-grained analysis of brain diseases. However, we have seen in our experiments that the multi-task approach presented in this chapter leads to a slight reduction in surface reconstruction accuracy. The next chapter presents a method for improving vertex correspondence between the input template and the prediction. This facilitates mapping an atlas parcellation from the template to the prediction.

13 Vertex Correspondence in Cortical Surface Reconstruction

Contents

The disadvantage of the previously mentioned joint reconstruction and parcellation approaches is the reduced surface reconstruction accuracy compared to specified surface reconstruction methods like Vox2Cortex. Further, mapping atlas parcellations such as the Desikan-Killiany-Tourville (DKT) atlas directly onto the predicted meshes produces inaccurate parcellations. This chapter focuses on vertex correspondence between the predicted meshes and the input template. Such vertex correspondence would enable direct mapping of parcellations and group comparisons of, e.g., cortical thickness measurements.

13.1 Introduction

The reconstruction of cortical surfaces from brain magnetic resonance imaging (MRI) scans is an essential task in neuroimaging studies. An array of algorithms has been developed and extensively employed for the reconstruction of cortical surfaces, which include, but are not limited to, FreeSurfer [39] and CAT12 [23]. While the surfaces reconstructed from a single individual can be exploited for calculating various metrics such as cortical thickness, curvature, and gyrification, one primary objective in reconstructing cortical surfaces is to facilitate group comparisons. These group comparisons are critical for the identification of structural discrepancies in the brain between patients and healthy control groups.

To execute these comparisons, it is essential to establish correspondence between the vertices of an individual's cortical mesh and a representative group template. This correspondence enables the evaluation of cortical thickness at the vertex level. Moreover, the establishment of vertex correspondence is crucial for the transference of atlas parcellation from the template to individual surfaces, which is instrumental in comparing measurements at a regional scale. This includes the computation of cortical thickness for specific parcels, and it has extensive

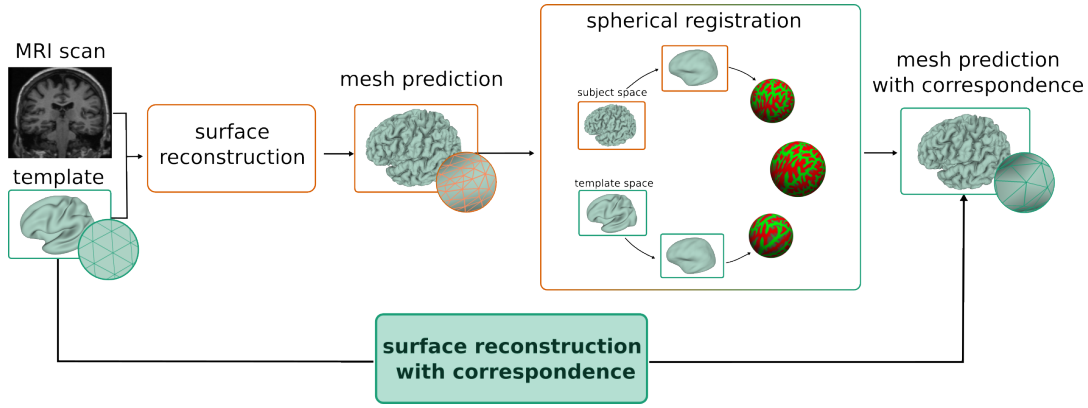

Figure 13.1. Top: Overview of existing cortical surface reconstruction approaches dependent on a cumbersome spherical registration as post-processing to obtain vertex correspondence to a template. Bottom: Our approach directly yields surface predictions with correspondence to the input template and does not require any registration.

applications in research on cortical maturation, as well as in the investigation of age-related and pathological cortical atrophy [86, 138, 150, 152].

Currently, the generation of vertex correspondence is accomplished in a post-processing phase, which constitutes a time-intensive procedure that generally involves aligning and remeshing an individual's surfaces to an atlas [38, 41]. As such, the direct generation of surfaces with integrated vertex correspondence could provide a valuable means for efficient, reliable, and precise cortical surface comparisons.

Recent advancements have given rise to various deep learning methodologies for cortical reconstruction, including DeepCSR [21], Vox2Cortex [10], CorticalFlow [84, 147], CortexODE [96], and Topofit [61].

Among these, template deformation strategies for cortical surface reconstruction employ a template mesh as an input to the deep learning model, and a vertex-centric deformation field is learned, conditioned on 3D MRI data. One of the primary challenges faced by these methods is the generation of smooth and watertight output meshes, which entails ensuring diffeomorphic mapping from the input to the output mesh. This challenge has been tackled through the incorporation of regularization losses [10, 61] or the numerical integration of an ordinary differential equation (ODE) that describes deformation [84, 96, 147]. The connectivity of the output mesh is mainly determined by the template mesh, with potential up- or down-sampling of the mesh resolution. Keeping the mesh resolution constant facilitates comparisons between reconstructed surfaces, as the output mesh has the same number of vertices and vertex connectivity as the input mesh. Despite maintaining constant mesh resolution, Vox2Cortex (V2C) [10], CorticalFlow++ [147] (CFPP), and Topofit [61] have not focused on optimizing or evaluating the accuracy of vertex correspondence.

In this work, we propose a novel surface reconstruction approach that natively provides correspondence to a template without spherical registration; see Fig. 13.1. We achieve this by training on meshes with vertex correspondence instead of meshes that vary in the number of vertices and vertex connectivity. For the network to learn these correspondences, we replace

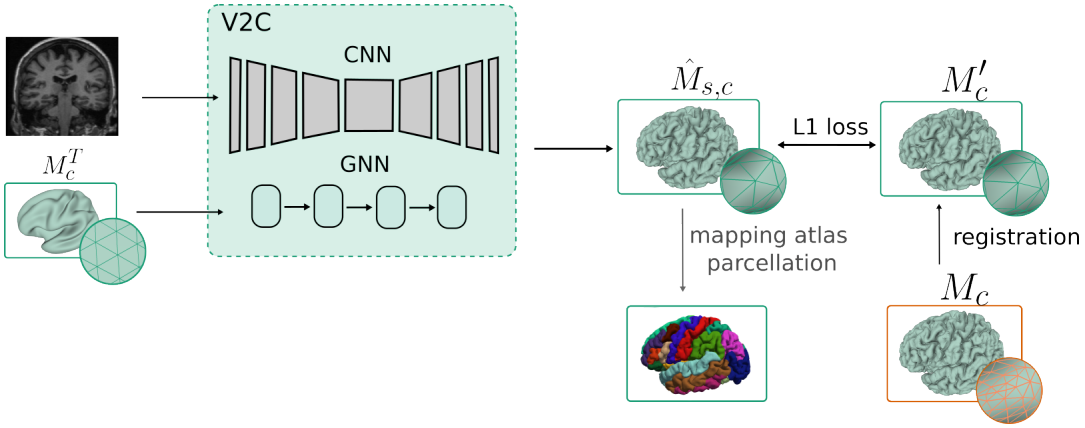

Figure 13.2. Overview of our V2CC method. The ground truth mesh is registered to the template mesh in a preprocessing step, allowing to compute the L1 loss on the vertex locations. As the surface reconstruction network, we use V2C [10]. Vertex correspondence to the template enables direct mapping of an atlas parcellation at inference.

the commonly used Chamfer loss with the L1 loss, which has yet to be used for cortical surface reconstruction. We use V2C as our backbone network as it is fast to train and provides white and pial surfaces for both hemispheres with one network, but our approach is generic. It can also be integrated into other surface reconstruction methods. We term our method Vox2Cortex with Correspondence (V2CC). We demonstrate that template deformation methods trained with the Chamfer loss, such as V2C, Topofit, and CFPP, provide insufficient vertex correspondences for mapping parcellations. Instead, our approach improves inter- and intra-subject vertex correspondence, making it suitable for direct group comparison and atlas-based parcellation.

13.2 Methods

In cortical surface reconstruction, template deformation methods transform a mesh template to match the neuroanatomy of the given patient. Following the notation introduced in Chapter 11, let M_c^T be the triangular mesh template, where $M_c^T = \{\mathbf{V}_c^T, \mathbf{F}_c^T, \mathbf{E}_c^T\}$, contains n vertices, represented as $\mathbf{V}_c^T \in \mathbb{R}^{n\times 3}$, o faces $\mathbf{F}_c^T \in \mathbb{R}^{o\times 3}$, storing the indices of the respective vertices that make up the triangles, and r edges $\mathbf{E}_c^T \in \mathbb{R}^{r\times 2}$, storing the indices of two adjacent faces that share a common edge. The surface reconstruction algorithm computes the displacement $f : \mathbb{R}^{n\times 3} \rightarrow \mathbb{R}^{n\times 3}$ for the set of vertices $\mathbf{V}_c^T$. The index c, ranging from 1 to C, denotes the specific surface under consideration. For our scenario, C is set to four, corresponding to each hemisphere's white and pial surfaces. As a recap of Section 11.3, in V2C, this displacement is computed by a graph convolutional network, conditioned on image features from a convolutional neural network that takes the Magnetic Resonance Imaging (MRI) scan as input. The two sub-networks are connected via feature-sampling modules that map features extracted by the Convolutional Neural Network (CNN) to vertex locations of the meshes. V2C addresses the issue of self-intersections in explicit surface reconstruction methods by incorporating multiple regularization terms into the loss function.

Let $M_c = \{\mathbf{V}_c, \mathbf{F}_c, \mathbf{E}_c\}$ be the ground truth mesh, and $\hat{M}_{s,c} = \{\hat{\mathbf{V}}_{s,c}, \hat{\mathbf{F}}_{s,c}, \hat{\mathbf{E}}_{s,c}\}$ the predicted deformed mesh, where $\hat{\mathbf{V}}_{s,c} \in \mathbb{R}^{n\times 3}$, and $\mathbf{V}_c \in \mathbb{R}^{m\times 3}$. Note that $n \neq m$ and no one-to-one mapping exists for vertex correspondence. The index $s \in 1, \ldots, S$ defines the different deformation stages, as in Chapter 11.

The complete loss function of V2C consists of a loss term for the CNN and a loss term for the mesh reconstruction, with further details in subsection 11.3.2. Here, we focus on the mesh reconstruction loss, which contains a geometric consistency loss and several regularization terms. The geometric consistency loss $\mathcal{L}_C$ is a curvature-weighted Chamfer loss and is defined as:

$$\mathcal{L}_{\text{Chamf}}(\hat{M}_{s,c}, M_c) = \frac{1}{|\mathcal{P}_c|}\sum_{\mathbf{u}\in\mathcal{P}_c}\min_{\mathbf{v}\in\hat{\mathcal{P}}_{s,c}}\|\mathbf{u}-\mathbf{v}\|^2 + \frac{1}{|\hat{\mathcal{P}}_{s,c}|}\sum_{\mathbf{v}\in\hat{\mathcal{P}}_{s,c}}\min_{\mathbf{u}\in\mathcal{P}_c}\|\mathbf{v}-\mathbf{u}\|^2, \tag{13.1}$$

where $\mathcal{P}_c \in \mathbb{R}^{q\times 3}$, and $\hat{\mathcal{P}}_{s,c} \in \mathbb{R}^{q\times 3}$ are point clouds sampled from the surfaces of M and $\hat{M}_{s,c}$ respectively. For simplicity, we have omitted the curvature weights here.

To optimize for vertex correspondence, we propose to use a pre-processing step that registers the mesh M_c to the template mesh M_c^T, resulting in a resampled ground truth mesh $M'_c = \{\mathbf{V'}_c, \mathbf{F'}_c, \mathbf{E'}_c\}$, with $\mathbf{V'}_c \in \mathbb{R}^{n\times 3}$ and $\mathbf{F'}_c \in \mathbb{R}^{o\times 3}$, where each index $i \in 1, \ldots, n$ represents the same anatomical location in both $\mathbf{V}_c^T$ and $\mathbf{V'}_c$. Instead of the Chamfer loss in (Equation (13.1)), we propose the loss function of V2CC as $\mathcal{L}(\hat{M}_{s,c}, M'_c) = \mathcal{L}_1(\hat{M}_{s,c}, M'_c) + \lambda\mathcal{L}_{reg}(\hat{M}_{s,c})$, where $\mathcal{L}_1$ is the mean absolute distance between corresponding vertices in M'_c and $\hat{M}_c$, and $\mathcal{L}_{reg}$ is the normal consistency regularization to avoid self-intersections in $\hat{M}_{s,c}$. $\mathcal{L}_1$ and $\mathcal{L}_{reg}$ are defined as:

$$\mathcal{L}_1(\hat{M}_{s,c}, M'_c) = \frac{1}{n}\sum_{i}^{n}|\mathbf{v}_i - \mathbf{u}_i|, \tag{13.2}$$

$$\mathcal{L}_{reg}(\hat{M}_{s,c}) = \frac{1}{|\hat{\mathbf{E}}_{s,c}|}\sum_{a,b\in\hat{\mathbf{E}}_{s,c}}(1-(\hat{\mathbf{n}}_a\cdot\hat{\mathbf{n}}_b))^2, \tag{13.3}$$

where $\hat{\mathbf{n}}_i$ is the unit normal of the i-th face of $\hat{M}_{s,c}$. Our method relies on only one regularization term, compared to three in V2C. In cortical surface reconstruction, there is an inherent trade-off to consider: surface accuracy, which can be optimized through geometric consistency losses like Chamfer or $\mathcal{L}_1$ loss, versus surface smoothness, which can be enhanced through regularization techniques. During preliminary experiments, we evaluated the regularizers used in V2C, see subsection 11.3.2, including Laplacian loss, intra-mesh normal consistency loss, and edge length loss in combination with our proposed $\mathcal{L}_1$ loss. Our findings suggested that using only the normal consistency regularization resulted in the best outcomes, while Laplacian smoothing didn't bring additional benefits. We decided against employing edge length regularization as it seemed counterintuitive to enforce uniform edge lengths while also using $\mathcal{L}_1$ loss to ensure precise vertex locations. The regularization factor λ needs to be tuned as a hyperparameter. Our proposed V2CC method and the pre-processing step are presented in Figure 13.2.

Table 13.1. Comparison of mesh quality of right hemisphere surfaces by ASSD in mm ±std and mean of percentage of self-intersecting faces (% SIF), and vertex correspondences by mean RMSD ±std of vertex positions and Dice overlap of mapped atlas parcellation. All models were trained on the ADNI data. RMSD values were computed on the TRT dataset, all other metrics on the data specified by the data column. Bold numbers indicate best performing methods. Input template: fsaverage6.

		Right Pial			Right WM			Average
Method	data	RMSD↓	ASSD↓	% SIF↓	RMSD↓	ASSD↓	% SIF↓	Dice↑
V2C [10]	ADNI	1.015 ±0.496	0.437 ±0.0311	1.123	0.961 ±0.447	0.372 ± 0.030	0.185	0.762
CFPP [147]	ADNI	0.884 ±0.353	0.3314 ±0.029	**0.052**	0.778 ±0.294	0.337 ±0.031	**0.013**	0.813
Topofit [61]	ADNI	-	-	-	1.271 ±0.410	**0.180 ±0.030**	0.022	0.838
V2CC only $\mathcal{L}_1$	ADNI	**0.816 ±0.337**	**0.268 ±0.036**	2.880	**0.739 ±0.268**	**0.228 ±0.036**	0.073	**0.921**
V2CC	ADNI	0.825 ±0.360	0.285 ±0.040	1.335	0.748 ±0.285	0.231 ±0.036	0.073	**0.921**

13.3 Experiments and Results

13.3.1 Experiment Setup

To assess the quality of reconstructed cortical surfaces, we employ four metrics. We evaluate the reconstructed surfaces ' quality with the Average Symmetric Surface Distance (ASSD) and the percentage of self-intersecting faces ($\%SIF$). We use two approaches for intra- and inter-subject cases to evaluate vertex correspondence. In *intra-subject cases*, we measure whether the same template vertex moves to the same location when provided with different scans of the same subject. In this case, we use scans acquired within a brief period to avoid structural changes. We calculate the consistency of vertex locations using the Root Mean Square Deviation (RMSD) of vertex positions. In *inter-subject cases*, we assess the ability of our method to map pre-defined parcellation atlases, such as the DKT atlas [27, 78], onto cortical surfaces. This mapping allows for assessing morphological measurements, such as cortical thickness in cortical regions. We directly map vertex classes from the template atlas onto the predicted mesh and calculate the Dice overlap (Dice) to FreeSurfer's silver standard parcellation to evaluate inter-subject vertex correspondence.

Similar to the experiments in the previous chapters, we use the $\text{ADNI}_{\text{large}}$ dataset and pre-processing as described in Section 10.6. We use FreeSurfer version 7.2. We used the Test-Retest dataset (TRT) dataset [98] to evaluate intra-subject correspondence, which contains 120 MRI T1w scans from 3 subjects, where each subject was scanned twice in 20 days. We further tested generalization to the Mindboggle-101 dataset [77] and the Japanese ADNI (J-ADNI).

13.3.2 Implementation Details

We use FreeSurfer's `mri_surf2surf` tool to register surfaces to FsAverage6 (40,962 vertices) and FsAverage (163,842 vertices) template surfaces. We used public implementations of baseline methods [1, 35, 146] and made adaptations so that all methods use the same template for training and testing. All models were trained on NVIDIA Titan-RTX or A100 GPUs. After grid search, the hyperparameter λ was set to 0.003 for white surfaces and 0.007 for pial

surfaces. For baseline models (V2C, CFPP, and Topofit), we used hyperparameters proposed by the original method. Topofit uses an early stopping regime when the validation loss plateaus, leading to a training time of 2600 epochs. CFPP trains each of the three U-Nets for 70000 iterations, which with a training set size of 1155, leads to 60 epochs per U-Net. We use the CFPP model with the highest performance on the validation set.

13.3.3 Results and Discussion:

We compare the proposed V2CC method to state-of-the-art models V2C, CFPP, and Topofit on the $\text{ADNI}_{\text{large}}$ dataset with FreeSurfer's FsAverage6 right hemisphere templates as an input mesh. All methods were trained using the resampled ground truth meshes. We show the results in the top part of Table 13.1. Topofit achieves the highest surface reconstruction accuracy on the white surface, and CFPP has the lowest number of self-intersecting faces pial surfaces. V2C achieves lower surface accuracy compared to CFPP and Topofit and has a higher number of self-intersections. We believe this could be due to longer training time for Topofit and CFPP, 2600 epochs for Topofit, and 60 epochs per U-Net for CFPP. For V2C and V2CC, we have trained models for 100 epochs and used the models with the best validation performance, which for V2CC was epoch 100 and for V2C epoch 90. This suggests that training the V2CC models for longer might lead to better results.

When replacing the loss function in V2C with $\mathcal{L}_1$, we observe an immense boost in surface accuracy and improved inter- and intra-subject correspondence. The disadvantage of using the $\mathcal{L}_1$ alone is seen in an increase of self-intersecting faces, especially on pial surfaces. Self-intersections of pial surfaces can be reduced by introducing the normal consistency regularizer in (Equation (13.2)).

Next, we trained the baseline V2C model, Topofit, and our V2CC model on higher-resolution templates (FsAverage) and images. We present the results on the right hemisphere in the lower section of Table 13.1. Results for the left hemisphere can be found in the supplementary material. We did not train CFPP on high resolution because of the long training process (about four weeks). For experiments on full-resolution (FsAverage) templates, we have trained V2C and V2CC for 140 epochs, where the best validation score for V2CC was at epoch 140 and for V2C at epoch 101. The full-resolution Topofit model stopped training at 2150 epochs. We can observe that all models achieve lower surface reconstruction error (cdist) when trained with higher resolution and more self-intersections. We believe this is partly due to existing self-intersections in the FsAverage templates and the resampled ground truth meshes. For the vertex correspondence metrics, we can observe that both baselines, V2C and Topofit, achieve higher parcellation scores than in the low-resolution experiment but are still outperformed by V2CC. We have further trained a state-of-the-art parcellation model (FastSurfer [59]) on the same dataset, which yields a Dice score of 0.88 ± 0.022, so we can conclude that our atlas-based parcellation can even outperform dedicated parcellation models. We visualize the quality of intra-subject correspondence of V2CC and FreeSurfer in the top box of Figure 13.3, where we display the per-vertex RMSD on each subject's white matter surface of the right hemisphere. We can observe that for all three subjects, our method leads to less variance of vertex positions than FreeSurfer. This is interesting, as FreeSurfer results were registered and resampled to obtain vertex correspondence, and our predictions were not. Further, this shows

Table 13.2. Comparison of mesh quality of right and left hemisphere surfaces by ASSD in mm ±std and mean of percentage of self-intersecting faces (% SIF), and vertex correspondences by mean RMSD ±std of vertex positions and Dice overlap of mapped atlas parcellation. All models were trained on the ADNI data. RMSD values were computed on the TRT dataset, all other metrics on the data specified by the data column. Bold numbers indicate best performing methods. The input template is fsaverage.

		Right Pial			Right WM			Average
Method	data	RMSD↓	ASSD↓	% SIF↓	RMSD↓	ASSD↓	% SIF↓	Dice↑
V2C [10]	ADNI	1.139 ±0.569	0.210 ±0.030	3.174	1.010 ±0.485	0.185 ±0.032	0.727	0.823
Topofit [61]	ADNI	-	-	-	1.326 ±0.406	**0.137 ±0.033**	**0.020**	0.871
V2CC	ADNI	**0.911 ±0.404**	**0.192 ±0.029**	**2.981**	**0.821 ±0.326**	0.186 ±0.035	0.110	**0.920**
V2C [10]	Mindb.	-	0.305 ±0.045	5.372	-	0.196 ±0.023	1.272	0.780
V2CC	Mindb.	-	0.305 ±0.048	4.453	-	0.204 ±0.030	0.157	0.865
V2C [10]	J-ADNI	-	0.262 ±0.046	3.578	-	0.222 ±0.078	1.063	0.803
V2CC	J-ADNI	-	0.262 ±0.048	3.614	-	0.230 ±0.079	0.140	0.913
		Left Pial			Left WM			Average
V2C [10]	ADNI	1.195 ±0.589	0.211 ±0.035	3.059	1.059 ±0.495	0.187 ±0.041	0.859	0.818
Topofit [61]	ADNI	-	-	-	1.300 ±0.362	0.137 ±0.041	0.027	0.875
V2CC	ADNI	0.950 ±0.398	0.196 ±0.034	2.903	0.853 ±0.306	0.186 ±0.042	0.120	0.921
V2C [10]	Mindb.	-	0.314 ±0.062	5.097	-	0.194 ±0.032	1.186	0.783
V2CC	Mindb.	-	0.305 ±0.053	4.356	-	0.199 ±0.031	0.187	0.865
V2C [10]	J-ADNI	-	0.258 ±0.047	3.491	-	0.221 ±0.080	1.077	0.805
V2CC	J-ADNI	-	0.262 ±0.053	3.549	-	0.226 ±0.082	0.148	0.916

that even though FreeSurfer surfaces have been used as ground truth to train our model, V2CC generalizes well and is more robust to subtle image changes. The bottom box in Figure 13.3 visualizes the parcellation result of one example subject and the average parcellation error over the whole test set. Parcellation errors occur mainly in boundary regions for all methods, but these boundary regions are much finer in V2CC. To test the generalization ability of our method, we tested V2CC and V2C on two unseen datasets J-ADNI and Mindboggle. We observe that V2CC achieves better parcellation Dice scores, while the surface reconstruction accuracy is similar for both methods.

13.3.4 Downstream Applications

We hypothesize that our meshes with vertex correspondence to the template can be directly used for downstream applications, such as group comparisons or disease classification, without the need for post-processing steps. We compared subjects with Alzheimer's disease (AD) and healthy controls of the $\text{ADNI}_{\text{large}}$ test set, where we compare cortical thickness measures on a per-vertex level. We present a visualization of p-values in Figure 13.4. Meshes generated with V2CC highlight regions similar to FreeSurfer meshes. The visualization shows significant atrophy throughout the cortex, with more substantial thinning in the left hemisphere, which matches findings from studies on cortical thinning in Alzheimer's disease [138, 152]. We further performed an AD classification study based on thickness measures on the $\text{ADNI}_{\text{large}}$ test set. We computed mean thickness values per parcel (DKT atlas parcellation) for V2CC, FreeSurfer, and the V2C [10] baseline. We show the classification results for AD and controls

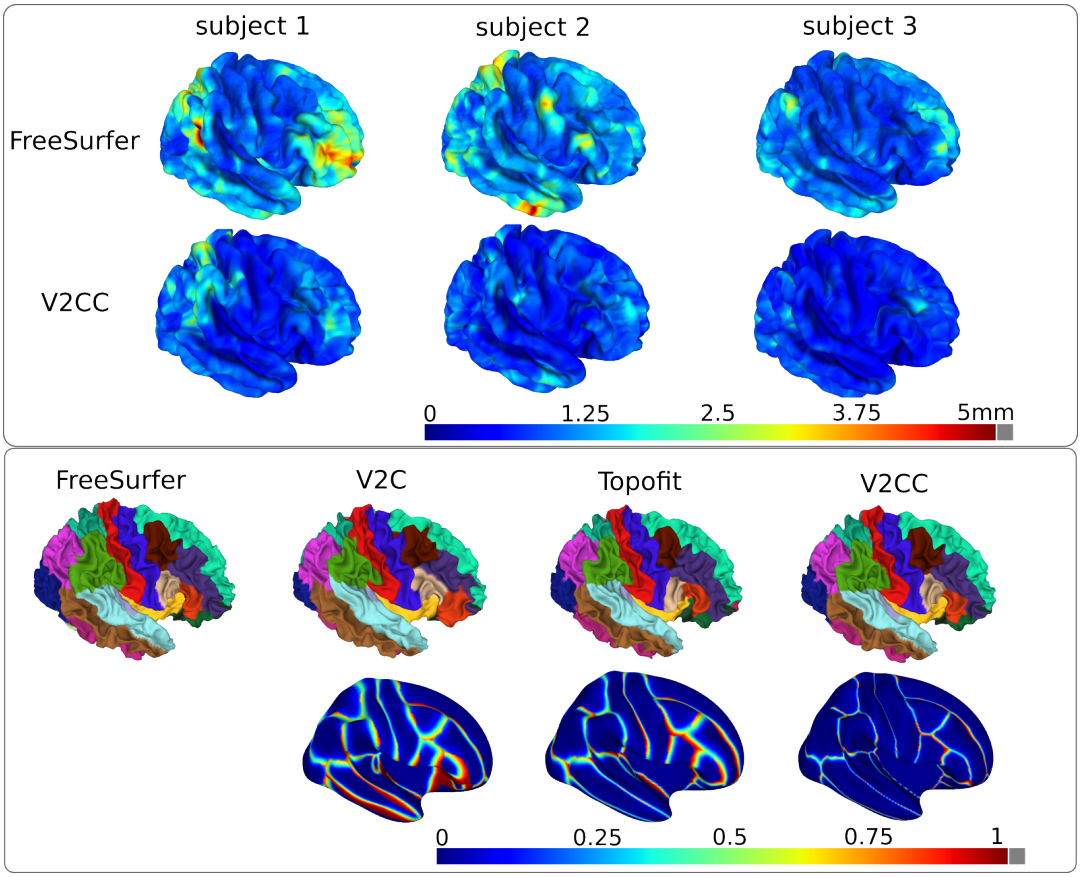

Figure 13.3. Top box: vertex RMSD on the TRT dataset. Bottom box: Top: Parcellation examples on a white surface of the right hemisphere of an example subject from the $\text{ADNI}_{\text{large}}$ test set. Bottom: Fraction of misclassified vertices over the test set, displayed on the smoothed FsAverage template.

using a gradient-boosted regression tree, trained on thickness measurements from the $\text{ADNI}_{\text{large}}$ training set. The classifiers achieved 0.810 balanced accuracy (bacc) for Freesurfer, 0.804 bacc for V2CC, and 0.776 bacc, for V2C on the $\text{ADNI}_{\text{large}}$ test set, demonstrating that V2CC achieves comparable results to FS and outperforms V2C.

13.4 Conclusion

In this work, we proposed V2CC, a novel approach for cortical surface reconstruction that directly provides vertex correspondences. V2CC utilizes a pre-processing step, where ground truth meshes are registered to a template, and directly learns the correspondences by optimizing an $\mathcal{L}_1$ loss instead of the commonly used Chamfer loss. We evaluated V2CC on several datasets, including $\text{ADNI}_{\text{large}}$, TRT, Mindboggle-101, and J-ADNI, and compared it to state-of-the-art methods. Our experimental results show that V2CC achieves comparable performance to previous methods regarding surface reconstruction accuracy. However, V2CC improves intra- and inter-subject correspondence and disease classification based on cortical thickness. We have evaluated our proposed pre-processing step and loss function with V2C as the backbone network. However, the underlying concepts are generic and could be integrated into other methods like Topofit or CFPP.

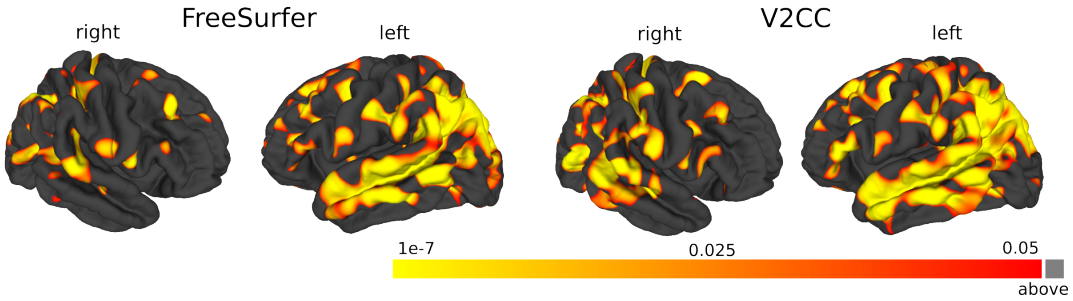

Figure 13.4. Group study of per-vertex cortical thickness measures in patients with Alzheimer's disease and healthy controls on the $\text{ADNI}_{\text{large}}$ test set. Colors indicate regions with significantly lower cortical thickness in AD subjects (t-test, one-sided). Note that our predicted meshes can be directly compared on a per-vertex basis, while FreeSurfer meshes need to be inflated to a sphere and registered.

14 Discussion of Surface-Based Segmentation Methods

14.1 Summary

In this part, I presented three novel methods to address challenges in cortical surface reconstruction and parcellation and improved vertex correspondence.

First, we introduced Vox2Cortex, a method for concurrently reconstructing white matter and pial surfaces from brain MR images. This approach begins with a general brain template and iteratively deforms it using a combination of convolutional and graph convolutional neural networks. We utilized a novel curvature-weighted Chamfer loss function to achieve high reconstruction accuracy in densely folded regions. The experiments demonstrated that this end-to-end trainable architecture generates state-of-the-art cortical surfaces with reduced computation time and allows for deriving precise cortical thickness maps for studying Alzheimer's disease.

Second, we proposed two extensions to cortex reconstruction networks like Vox2Cortex for joint cortex parcellation: a graph classifier-based extension and one based on a region-based reconstruction loss. Both methods are compatible with template deformation networks, resulting in meshes with slightly reduced surface accuracy and precise parcellation. With rapid inference times and high parcellation accuracy, these algorithms are suitable for large-cohort studies and clinical practice in the fine-grained analysis of brain diseases.

Lastly, we introduced Vox2Cortex with Correspondence (V2CC), an approach for cortical surface reconstruction that directly provides vertex correspondences. V2CC leverages a pre-processing step involving ground truth mesh registration to a template and optimizes an L1 loss instead of the conventional Chamfer loss. The evaluation on multiple datasets, including ADNI, TRT, Mindboggle-101, and J-ADNI, demonstrated that V2CC achieves comparable surface reconstruction accuracy to state-of-the-art methods while improving intra- and inter-subject correspondence and disease classification based on cortical thickness. The pre-processing step and loss function can potentially be integrated into other methods like Topofit or CFPP.

14.2 Discussion

Vox2Cortex was among the initially published template deformation approaches. The method may be sensitive to the quality of the input template and could perform better if the template is representative of the subjects in the dataset. In [10], we used a random subject from our

training set to serve as a template, which outperformed spherical or ellipsoid templates that were predominantly used before. However, choosing a random subject is not optimal, as it could be an outlier. We addressed this in our later work by using FreeSurfer's FsAverage template. We have also experimented with group-specific templates, e.g., different templates for males and females and different age groups. In preliminary experiments, we have seen that this can lead to improved performance.

Concurrently to Vox2Cortex, CorticalFlow [84] was introduced. CorticalFlow, another template deformation approach, learns the deformation field using a 3D U-Net instead of a graph convolutional network. The advantage of learning deformation at the voxel level is the reduction of self-intersections, as demonstrated in Chapter 13. However, a disadvantage is the strong dependence of surface reconstruction accuracy on voxel and template resolution. With lower template resolution, such as with the FsAverage6 template, vertex deformation relies heavily on interpolation, leading to less accurate predictions. Furthermore, the official implementation of CorticalFlow trains separate U-Nets for all four surfaces, making the training process time-consuming.

CorticalFlow has been recently extended to CorticalFlow++ (CFPP) [147], which employs a more accurate ODE solver and deforms white surfaces to predict pial surfaces, analogous to the FreeSurfer approach. This results in vertex correspondence between white and pial surfaces. CortexODE [96], another recently proposed method, also deforms white surfaces to pial surfaces. This approach is prevalent in the literature, likely due to its use in FreeSurfer. For example, the argument is that correspondence between white and pial surfaces is essential for utilizing existing surface analysis tools [147]. Our experiments demonstrate that using nearest-neighbor correspondence is equally accurate for cortical thickness estimation. However, a registration processing step is necessary if one wishes to use Vox2Cortex surfaces with surface analysis tools requiring white-to-pial correspondence.

With our latest adaptation, V2CC, registration is no longer required, as white and pial surfaces correspond to FsAverage, which has white-to-pial correspondence. This results in correspondence between our white and pial surfaces. However, minor errors in correspondence still exist, so if white-to-pial correspondence is crucial, the V2CC network could be modified to predict white surfaces first and then predict pial surfaces based on the white surface.

Many methods, such as CorticalFlow or CortexODE, formulate template deformation as an ODE problem, potentially leading to fewer self-intersections. It is feasible to design the Vox2Cortex architecture similarly, and we have attempted this, resulting in further improvements in surface reconstruction accuracy. The manuscript [11] is currently under review and not part of this thesis.

V2CC's performance may be sensitive to the quality of the pre-processing step, including template registration and ground truth mesh alignment. For example, if FreeSurfer's spherical registration leads to errors or reduced surface quality, as it also resamples the mesh and potentially could lead to self-intersections, this would impact the quality of V2CC's predictions. Learning vertex correspondence in an unsupervised way could be a potential solution to this problem.

A general problem for cortical surface reconstruction is the reliance on FreeSurfer as a silver standard, as no manual ground truth surfaces exist. In the future, it would be interesting to develop a method for training and evaluating cortical surface reconstruction techniques without relying on FreeSurfer. Recent research has proposed a cortical thickness estimation benchmark [145], which involves designing a synthetic dataset to assess a method's ability to detect subtle changes in cortical thickness, which could be interesting for evaluating our methods further.

Other methods, such as Topofit [61] or CFPP, could potentially benefit from incorporating V2CC's pre-processing step and loss function to improve vertex correspondence in their frameworks. In general, the task of cortical surface reconstruction from Magnetic Resonance Imaging (MRI) scans has reached a point where surface accuracy is already quite precise on commonly used datasets like ADNI. The focus now shifts to evaluating the robustness of these models on large-scale cohorts like the UK Biobank, whether the models are fair to underrepresented demographics, and how to improve such fairness.

14.3 Future Work

Referring back to the challenges defined in Section 10.5, we have made significant strides in mitigating some of the obstacles confronting deep learning for cortical surface reconstruction. However, certain challenges persist, which can be addressed in future work.

With our proposed methods, Vox2Cortex and V2CC, the inference time has been reduced to mere seconds, dramatically improving the multi-hour runtime required by FreeSurfer. At this point, efforts to increase speed further seem less imperative.

Our approach also successfully delivers a surface accuracy well below the $1mm$ threshold. The accuracy achieved, we believe, meets the need for a highly accurate representation of cortical surfaces.

We've addressed the issue of preserving spherical topology by employing a spherical template like FsAverage, ensuring topological correctness, an essential requirement in cortical surface reconstruction.

However, there's a lingering issue of occasional self-intersections in the generated surfaces that remains to be fully resolved. Future explorations should aim to refine our methods to eliminate self-intersections.

Methods such as V2CC, by providing both inter-subject and intra-subject correspondences, facilitate group and intra-subject comparisons with ease. But we still need to apply our approach to longitudinal datasets or compare them to existing longitudinal processing pipelines such as FreeSurfer's longitudinal stream. This would be an exciting direction for future endeavors.

Lastly, it's important to acknowledge our methods' dependence on FreeSurfer. Future work should involve efforts to devise methods that reduce this reliance on third-party software.

Part IV

Conclusion

15 Conclusion

This thesis has comprehensively examined deep learning techniques in medical image segmentation. The main contributions of this thesis have been divided into two main parts, each focusing on different directions of medical image segmentation. The first direction, presented in Part II explored voxel-based segmentation techniques, where we introduced innovative algorithms that enhanced the segmentation quality and efficiency of brain and abdominal MR and Computed Tomography (CT) images. Our methods demonstrated robust performance in the face of domain shift and data scarcity, critical issues in applying deep learning models in medical imaging. In Part III, we tackled the complexities of cortical surface reconstruction and parcellation through surface-based segmentation methodologies. We made significant improvements in the reconstruction quality of cortical surfaces as well as in learning-based and atlas-based parcellation, facilitating improved neuroimaging analyses. The broader implications of our work extend beyond academia and into clinical neuroscience and neurology, as well as other areas of clinical interest, such as abdominal imaging. Improved segmentation and surface reconstruction techniques can significantly impact our understanding and management of various diseases, including neurodegenerative conditions like Alzheimer's.

While significant strides have been made throughout our research, it's important to acknowledge its limitations. A common challenge in the field pertains to biases in the datasets used for training our models. These biases can impact the generalizability of our results, highlighting the need for future research to consider diverse datasets encompassing varied demographic characteristics and health conditions. Additionally, though our methods showed promise in improving computation times and segmentation quality, their translation into a clinical setting remains a challenging task. Looking forward, I see numerous potential avenues for further research. There is scope for our methods to be examined in longitudinal datasets, offering an exciting direction for future investigations. Further, transitioning from research to a clinical setting is challenging yet crucial. Efforts to optimize and tailor our methods for real-world medical environments will be a key focus. This will include the exploration of bias and fairness in segmentation models and delving into the instances where our methods did not perform as expected. Our work with Hallucination-free Organ Segmentation (HALOS), for example, identified an instance where a well-performing model, nnUNet, hallucinated a non-existing organ. Discoveries like this can lead to model refinement and, ultimately, better clinical translation. Analyzing the performance of our methods on population-scale datasets, like the UK Biobank, can provide valuable insights into their efficacy, adaptability, and limitations and potentially be a first step toward clinical translation.

In conclusion, this thesis has made several contributions to the medical image segmentation field by introducing novel voxel-based and surface-based segmentation methods. As we continue to refine these techniques, we hope to see them contribute to driving forward the field of medical image computing and ultimately enhancing patient care.

A Appendix

Proof for Cuvature-Weighted Chamfer:

In order to provide a concise mathematical explanation for the emphasis on geometric accuracy in high-curvature regions compared to low-curvature regions by our curvature-weighted Chamfer loss, let us consider two ground-truth points, denoted as a and b, with corresponding curvatures $\kappa(a) < \kappa(b)$. Additionally, we have the closest predicted points, represented by u and v (as illustrated in Appendix A). Assuming that the distances between the predictions and ground truth are equal, such that $|u - a| = |v - b|$, we can simplify the treatment of u and v as parameters optimized by gradient descent. Let u' be the updated value of u based on the gradient descent update step, given by $u' = u - \lambda \frac{\partial \mathcal{L}_{\mathrm{C}}(a,u)}{\partial u}$, where $\lambda > 0$ represents the learning rate. Referring to Equation (11.6), we can calculate the gradient of the curvature-weighted Chamfer loss with respect to u as follows:

$$\begin{aligned} \frac{\partial \mathcal{L}_{\mathrm{C}}(a,u)}{\partial u} &= \frac{\partial}{\partial u}\left[\kappa(a)\left(\|a-u\|^2 + \|u-a\|^2\right)\right] \\ &= 4\kappa(a)(u-a). \end{aligned} \tag{A.1}$$

The calculation of $\frac{\partial \mathcal{L}_{\mathrm{C}}(b,v)}{\partial v} = 4\kappa(b)(v-b)$ works analogously. The parameter updates are given by

$$\begin{aligned} u' &= u - \frac{\partial \mathcal{L}_{\mathrm{C}}(a,u)}{\partial u} = u + 4\lambda\kappa(a)(a-u), \\ v' &= v - \frac{\partial \mathcal{L}_{\mathrm{C}}(b,v)}{\partial v} = v + 4\lambda\kappa(b)(b-v). \end{aligned} \tag{A.2}$$

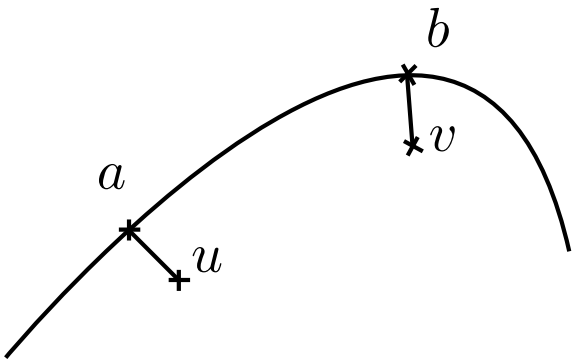

Figure A.1. Given two ground-truth points, a and b, with different curvatures, where $\kappa(a) < \kappa(b)$, and corresponding predicted points, u and v with equal distance to the ground truth points.

Moreover, we have $\|a - u\| = \|b - v\|$ and $\kappa(a) < \kappa(b)$. Consequently, if we assume that we do not "shoot over" the desired outcome, i.e., $0 < 4\lambda\kappa(a) < 4\lambda\kappa(b) < 1$, we can observe that $|v' - b| < |u' - a|$. This means that during one backward pass, point v is pushed closer to b compared to how much point u is pushed towards a. ■

B

Authored and Co-Authored Publications

Authored

1. **A. Rickmann**, F. Bongratz, C. Wachinger. *Vertex Correspondence in Cortical Surface Reconstruction* accepted at International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI), 2023.

2. **A. Rickmann**, M. Xu, T. Wolf, O. Kovalenko, C. Wachinger. *HALOS: Hallucination free Organ Segmentation after Organ Resection Surgery* in International Conference on Information Processing in Medical Imaging. Cham: Springer Nature Switzerland, 2023.

3. **A. Rickmann**, F. Bongratz, S. Pölsterl, I. Sarasua, C. Wachinger. *Joint Reconstruction and Parcellation of Cortical Surfaces* in International Workshop on Machine Learning in Clinical Neuroimaging. Cham: Springer Nature Switzerland, 2022.

4. F. Bongratz, **A. Rickmann**, S. Pölsterl, C. Wachinger. *Vox2Cortex: Fast Explicit Reconstruction of Cortical Surfaces from 3D MRI Scans with Geometric Deep Neural Networks* in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022.

5. **A. Rickmann**, J. Senapati, O. Kovalenko, A. Peters, F. Bamberg, C. Wachinger. *AbdomenNet: Deep Neural Network for Abdominal Organ Segmentation in Epidemiologic Imaging Studies* in BMC Medical Imaging 22.1 (2022): 1-11.

6. F. Gröger, **A. Rickmann**, C. Wachinger. *STRUDEL: Self-Training with Uncertainty Dependent Label Refinement Across Domains* in International Workshop on Machine Learning in Medical Imaging, Cham: Springer Nature Switzerland, 2021.

7. S. Özgün, **A. Rickmann**, A. Guha Roy, C. Wachinger. *Importance Driven Continual Learning for Segmentation Across Domains* in International Workshop on Machine Learning in Medical Imaging, Cham: Springer Nature Switzerland, 2020.

8. **A. Rickmann**, A. Guha Roy, I. Sarasua, C. Wachinger. *Recalibrating 3D ConvNets with Project & Excite* in IEEE transactions on medical imaging, 39.7 (2020): 2461-2471.

9. **A. Rickmann**, A. Guha Roy, I. Sarasua, N. Navab, C. Wachinger. *'Project & Excite' Modules for Segmentation of Volumetric Medical Scans* in International Conference of Medical

Image Computing and Computer Assisted Intervention, Springer International Publishing, 2019.

Co-Authored

1. T. Wolf, F. Bongratz, **A. Rickmann**, S. Pölsterl, C. Wachinger. *Keep the Faith: Faithful Explanations in Convolutional Neural Networks for Case-Based Reasoning* accepted at AAAI Conference on Artificial Intelligence 2024

2. F. Bongratz, **A. Rickmann**, C. Wachinger *Abdominal organ segmentation via deep diffeomorphic mesh deformations* in Scientific Reports 13, 18270 (2023)

3. F. Bongratz, **A. Rickmann**, C. Wachinger *Neural Deformation Fields for Mesh-Based Reconstruction of Cortical Surfaces from MRI* 2023 - manuscript under review

Bibliography

[1] D. G. Andrew Hoopes. *Topofit code*. `https://github.com/ahoopes/topofit`. Accessed: 2023-03-04 (cit. on p. 135).

[2] M. Arjovsky and L. Bottou. "Towards principled methods for training generative adversarial networks". In: *arXiv preprint arXiv:1701.04862* (2017) (cit. on p. 57).

[3] M. Attene. "A lightweight approach to repairing digitized polygon meshes". en. In: *The Visual Computer* 26.11 (Nov. 2010), pp. 1393–1406 (cit. on p. 118).

[4] B. B. Avants, N. Tustison, and G. Song. "Advanced normalization tools (ANTS)". In: *Insight j* 2.365 (2009), pp. 1–35 (cit. on p. 61).

[5] V. Badrinarayanan, A. Kendall, and R. Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017), pp. 2481–2495 (cit. on p. 33).

[6] F. Bamberg, H. Hetterich, S. Rospleszcz, et al. "Subclinical disease burden as assessed by whole-body MRI in subjects with prediabetes, subjects with diabetes, and normal control subjects from the general population: the KORA-MRI study". In: *Diabetes* 66.1 (2017), pp. 158–169 (cit. on p. 74).

[7] F. Bamberg, H.-U. Kauczor, S. Weckbach, et al. "Whole-body MR imaging in the German National Cohort: rationale, design, and technical background". In: *Radiology* 277.1 (2015), pp. 206–220 (cit. on p. 74).

[8] L. Beckett, M. Donohue, C. Wang, P. Aisen, D. Harvey, and N. Saito. "The Alzheimer's Disease Neuroimaging Initiative phase 2: Increasing the length, breadth, and depth of our understanding". In: *Alzheimer's & dementia : the journal of the Alzheimer's Association* 11 (July 2015), pp. 823–31 (cit. on p. 61).

[9] M. F. Bobo, S. Bao, Y. Huo, et al. "Fully convolutional neural networks improve abdominal organ segmentation". In: *Medical Imaging 2018: Image Processing*. Vol. 10574. International Society for Optics and Photonics. 2018, p. 105742V (cit. on p. 69).

[10] F. Bongratz, A.-M. Rickmann, S. Pölsterl, and C. Wachinger. "Vox2Cortex: Fast Explicit Reconstruction of Cortical Surfaces from 3D MRI Scans with Geometric Deep Neural Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 20773–20783 (cit. on pp. 6, 85, 96, 100, 101, 117, 121, 123, 125, 127, 132, 133, 135, 137, 141).

[11] F. Bongratz, A.-M. Rickmann, and C. Wachinger. "Neural Deformation Fields for Mesh-Based Reconstruction of Cortical Surfaces from MRI". In: - (2023) (cit. on p. 142).

[12] M. M. Bronstein, J. Bruna, T. Cohen, and P. Velickovic. "Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges". In: *CoRR* abs/2104.13478 (2021). arXiv: `2104.13478` (cit. on pp. 95, 105).

[13] H. Chen, Q. Dou, L. Yu, J. Qin, and P. A. Heng. "VoxResNet: Deep voxelwise residual networks for brain segmentation from 3D MR images". In: *NeuroImage* 170.April (2018), pp. 446–455 (cit. on pp. 33, 35, 43–45, 47, 48).

[14] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs". In: *IEEE transactions on pattern analysis and machine intelligence* 40.4 (2017), pp. 834–848 (cit. on p. 33).

[15] Y. Chen, D. Ruan, J. Xiao, et al. "Fully automated multi-organ segmentation in abdominal magnetic resonance imaging with deep neural networks". In: *Medical physics* 47.10 (2020), p. 4971 (cit. on pp. 68, 69).

[16] Y. Chen, H. Fan, B. Xu, et al. "Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 3435–3444 (cit. on p. 60).

[17] 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*. Ed. by S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, and W. Wells. Cham: Springer International Publishing, 2016, pp. 424–432 (cit. on pp. 26, 27, 47, 48, 103).

[18] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger. "3D U-Net: learning dense volumetric segmentation from sparse annotation". In: *MICCAI*. Springer. 2016, pp. 424–432 (cit. on pp. 33, 43, 45, 49, 52).

[19] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. "MeshLab: an Open-Source Mesh Processing Tool". In: *Eurographics Italian Chapter Conference*. Ed. by V. Scarano, R. D. Chiara, and U. Erra. The Eurographics Association, 2008 (cit. on p. 126).

[20] P. Coupé, B. Mansencal, M. Clément, et al. "AssemblyNet: A large ensemble of CNNs for 3D whole brain MRI segmentation". In: *NeuroImage* 219 (2020), p. 117026 (cit. on p. 122).

[21] R. S. Cruz, L. Lebrat, P. Bourgeat, C. Fookes, J. Fripp, and O. Salvado. "DeepCSR: A 3D Deep Learning Approach for Cortical Surface Reconstruction". In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2021), pp. 806–815 (cit. on pp. 96, 98, 102, 110, 114, 115, 118, 121, 132).

[22] G. Cucurull, K. Wagstyl, A. Casanova, et al. "Convolutional neural networks for mesh-based parcellation of the cerebral cortex". In: *Medical Imaging with Deep Learning*. 2018 (cit. on pp. 121, 122, 124).

[23] R. Dahnke, R. A. Yotter, and C. Gaser. "Cortical thickness and central surface estimation". In: *Neuroimage* 65 (2013), pp. 336–348 (cit. on p. 131).

[24] A. M. Dale, B. Fischl, and M. I. Sereno. "Cortical surface-based analysis: I. Segmentation and surface reconstruction". In: *Neuroimage* 9.2 (1999), pp. 179–194 (cit. on pp. 91, 101, 106).

[25] L. DE TOLEDO-MORRELL, I. Goncharova, B. Dickerson, R. S. Wilson, and D. A. Bennett. "From healthy aging to early Alzheimer's disease: in vivo detection of entorhinal cortex atrophy". In: *Annals of the New York Academy of Sciences* 911.1 (2000), pp. 240–253 (cit. on p. 88).

[26] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255 (cit. on p. 34).

[27] R. S. Desikan, F. Ségonne, B. Fischl, et al. "An automated labeling system for subdividing the human cerebral cortex on MRI scans into gyral based regions of interest". In: *Neuroimage* 31.3 (2006), pp. 968–980 (cit. on pp. 91, 94, 123, 135).

[28] C. Destrieux, B. Fischl, A. Dale, and E. Halgren. “Automatic parcellation of human cortical gyri and sulci using standard anatomical nomenclature”. In: *NeuroImage* 53.1 (2010), pp. 1–15 (cit. on pp. 91, 92, 94, 123).

[29] J. Dolz, K. Gopinath, J. Yuan, H. Lombaert, C. Desrosiers, and I. B. Ayed. “HyperDense-Net: A hyper-densely connected CNN for multi-modal image segmentation”. In: *IEEE transactions on medical imaging* 38.5 (2018), pp. 1116–1126 (cit. on pp. 33, 35).

[30] J. Dubois, M. Benders, A. Cachia, et al. “Mapping the early cortical folding process in the preterm newborn brain”. In: *Cerebral cortex* 18.6 (2008), pp. 1444–1454 (cit. on p. 88).

[31] B. N. Dugger and D. W. Dickson. “Pathology of neurodegenerative diseases”. In: *Cold Spring Harbor perspectives in biology* 9.7 (2017), a028035 (cit. on p. 88).

[32] J. S. Duncan and N. Ayache. “Medical image analysis: Progress over two decades and the challenges ahead”. In: *IEEE transactions on pattern analysis and machine intelligence* 22.1 (2000), pp. 85–106 (cit. on pp. 12, 13).

[33] H. Edelsbrunner and J. L. Harer. *Computational topology: an introduction*. American Mathematical Society, 2022 (cit. on p. 89).

[34] K. M. Eschenburg, T. J. Grabowski, and D. R. Haynor. “Learning Cortical Parcellations Using Graph Neural Networks”. In: *Frontiers in neuroscience* 15 (2021) (cit. on pp. 122, 124).

[35] A.-M. R. Fabian Bongratz. *Vox2Cortex code*. `https://github.com/ai-med/Vox2Cortex`. Accessed: 2023-03-04 (cit. on p. 135).

[36] H. Fan, H. Su, and L. Guibas. “A Point Set Generation Network for 3D Object Reconstruction from a Single Image”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2463–2471 (cit. on p. 107).

[37] F. Fazekas, J. B. Chawluk, A. Alavi, H. I. Hurtig, and R. A. Zimmerman. “MR signal abnormalities at 1.5 T in Alzheimer’s dementia and normal aging”. In: *American Journal of Neuroradiology* 8.3 (1987), pp. 421–426 (cit. on p. 55).

[38] B Fischl, M. Sereno, R. Tootell, and A. Dale. “High-resolution inter-subject averaging and a coordinate system for the cortical surface.” In: *Human Brain Mapping* 8.4 (1999), pp. 272–284 (cit. on pp. 91, 124, 132).

[39] B. Fischl. “FreeSurfer”. In: *Neuroimage* 62.2 (2012), pp. 774–781 (cit. on pp. 42, 85, 90, 91, 98, 99, 131).

[40] B. Fischl, D. H. Salat, E. Busa, et al. “Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain”. In: *Neuron* 33.3 (2002), pp. 341–355 (cit. on p. 91).

[41] B. Fischl, M. I. Sereno, and A. M. Dale. “Cortical surface-based analysis: II: inflation, flattening, and a surface-based coordinate system”. In: *Neuroimage* 9.2 (1999), pp. 195–207 (cit. on pp. 91, 101, 132).

[42] B. Fischl, A. Van Der Kouwe, C. Destrieux, et al. “Automatically parcellating the human cerebral cortex”. In: *Cerebral cortex* 14.1 (2004), pp. 11–22 (cit. on pp. 91, 94).

[43] Y. Gal and Z. Ghahramani. “Dropout as a bayesian approximation: Representing model uncertainty in deep learning”. In: *international conference on machine learning*. PMLR. 2016, pp. 1050–1059 (cit. on pp. 30, 59).

[44] M. Garland and P. S. Heckbert. “Surface Simplification Using Quadric Error Metrics”. In: *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’97. USA: ACM Press/Addison-Wesley Publishing Co., 1997, 209–216 (cit. on p. 98).

[45] E. Gibson, F. Giganti, Y. Hu, et al. “Automatic multi-organ segmentation on abdominal CT with dense v-networks”. In: *IEEE transactions on medical imaging* 37.8 (2018), pp. 1822–1834 (cit. on pp. 68, 69).

[46] G. Gkioxari, J. Johnson, and J. Malik. “Mesh R-CNN”. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 9784–9794 (cit. on pp. 107, 109, 125).

[47] K. Gopinath, C. Desrosiers, and H. Lombaert. “Graph convolutions on spectral embeddings for cortical surface parcellation”. In: *Medical image analysis* 54 (2019), pp. 297–305 (cit. on pp. 121, 123).

[48] K. Gopinath, C. Desrosiers, and H. Lombaert. “SegRecon: Learning Joint Brain Surface Reconstruction and Segmentation from Images”. In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*. Ed. by M. de Bruijne, P. C. Cattin, S. Cotin, et al. Cham: Springer International Publishing, 2021, pp. 650–659 (cit. on pp. 102, 121, 123).

[49] L. Griffanti, M. Jenkinson, S. Suri, et al. “Classification and characterization of periventricular and deep white matter hyperintensities on MRI: a study in older adults”. In: *Neuroimage* 170 (2018), pp. 174–181 (cit. on p. 55).

[50] F. Gröger, A.-M. Rickmann, and C. Wachinger. “STRUDEL: Self-training with Uncertainty Dependent Label Refinement Across Domains”. In: *Machine Learning in Medical Imaging*. Ed. by C. Lian, X. Cao, I. Rekik, X. Xu, and P. Yan. Cham: Springer International Publishing, 2021, pp. 306–316 (cit. on p. 5).

[51] T. Groueix, M. Fisher, V. G. Kim, B. Russell, and M. Aubry. “AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation”. In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018 (cit. on pp. 103, 113).

[52] H. Guan and M. Liu. “Domain adaptation for medical image analysis: a survey”. In: *IEEE Transactions on Biomedical Engineering* 69.3 (2021), pp. 1173–1185 (cit. on p. 68).

[53] M. Guan, V. Gulshan, A. Dai, and G. Hinton. “Who said what: Modeling individual labelers improves classification”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1. 2018 (cit. on p. 58).

[54] K. Gupta and M. Chandraker. “Neural Mesh Flow: 3D Manifold Mesh Generation via Diffeomorphic Flows”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 1747–1758 (cit. on p. 107).

[55] M. Hartig, D. Truran-Sacrey, S. Raptentsetsang, et al. *UCSF FreeSurfer Methods*. Tech. rep. Alzheimer’s Disease Neuroimaging Initiative, 2014 (cit. on pp. 97, 117).

[56] A. Hatamizadeh, Y. Tang, V. Nath, et al. “Unetr: Transformers for 3d medical image segmentation”. In: *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2022, pp. 574–584 (cit. on p. 34).

[57] K. He, X. Zhang, S. Ren, and J. Sun. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 770–778 (cit. on pp. 103, 104).

[58] K. He, X. Zhang, S. Ren, and J. Sun. “Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification”. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 1026–1034 (cit. on p. 28).

[59] L. Henschel, S. Conjeti, S. Estrada, K. Diers, B. Fischl, and M. Reuter. “FastSurfer - A fast and accurate deep learning based neuroimaging pipeline”. In: *NeuroImage* 219 (2020), p. 117012 (cit. on pp. 29, 102, 122, 126, 127, 136).

[60] M. d. C. V. Hernández, V. González-Castro, D. T. Ghandour, et al. “On the computational assessment of white matter hyperintensity progression: difficulties in method selection and bias field correction performance on images with significant white matter pathology”. In: *Neuroradiology* 58.5 (2016), pp. 475–485 (cit. on p. 61).

[61] A. Hoopes, J. E. Iglesias, B. Fischl, D. Greve, and A. V. Dalca. "TopoFit: Rapid Reconstruction of Topologically-Correct Cortical Surfaces". In: *Medical Imaging with Deep Learning*. 2021 (cit. on pp. 132, 135, 137, 143).

[62] J. Hu, L. Shen, and G. Sun. "Squeeze-and-Excitation Networks". In: *CVPR*. 2018 (cit. on pp. 34, 36, 37, 42, 44, 48, 49, 52, 73).

[63] J. Hu, L. Shen, and G. Sun. "Squeeze-and-excitation networks". In: *CVPR*. 2018 (cit. on p. 70).

[64] Y. Huo, Z. Xu, K. Aboud, et al. "Spatially localized atlas network tiles enables 3D whole brain segmentation from limited data". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2018, pp. 698–705 (cit. on pp. 34, 35).

[65] Y. Huo, Z. Xu, S. Bao, A. Assad, R. G. Abramson, and B. A. Landman. "Adversarial synthesis learning enables segmentation without target modality ground truth". In: *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*. IEEE. 2018, pp. 1217–1220 (cit. on p. 57).

[66] Y. Huo, Z. Xu, Y. Xiong, et al. "3D whole brain segmentation using spatially localized atlas network tiles". In: *NeuroImage* 194 (2019), pp. 105–119 (cit. on p. 122).

[67] S. Ioffe and C. Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by F. Bach and D. Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, 2015, pp. 448–456 (cit. on p. 28).

[68] F. Isensee, P. F. Jaeger, S. A. A. Kohl, J. Petersen, and K. Maier-Hein. "nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation." In: *Nature methods* (2020) (cit. on pp. 27, 28, 70, 76, 77, 103, 114).

[69] F. Isensee, P. F. Jaeger, S. A. Kohl, J. Petersen, and K. H. Maier-Hein. "nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation". In: *Nature methods* 18.2 (2021), pp. 203–211 (cit. on pp. 27, 28, 68, 69).

[70] C. R. Jack, M. A. Bernstein, N. C. Fox, et al. "The Alzheimer's disease neuroimaging initiative (ADNI): MRI methods". In: *Journal of magnetic resonance imaging* 27.4 (2008), pp. 685–691 (cit. on p. 43).

[71] K. Kamnitsas, C. Baumgartner, C. Ledig, et al. "Unsupervised domain adaptation in brain lesion segmentation with adversarial networks". In: *Information Processing in Medical Imaging: 25th International Conference, IPMI 2017, Boone, NC, USA, June 25-30, 2017, Proceedings 25*. Springer. 2017, pp. 597–609 (cit. on p. 57).

[72] K. Kamnitsas, C. Ledig, V. F. Newcombe, et al. "Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation". In: *Medical image analysis* 36 (2017), pp. 61–78 (cit. on p. 35).

[73] T. Kart, M. Fischer, S. Winzeck, et al. "Automated imaging-based abdominal organ segmentation and quality control in 20,000 participants of the UK Biobank and German National Cohort Studies". In: *Scientific Reports* 12.1 (2022), pp. 1–11 (cit. on p. 74).

[74] M. Kass, A. Witkin, and D. Terzopoulos. "Snakes: Active contour models". In: *International journal of computer vision* 1.4 (1988), pp. 321–331 (cit. on p. 13).

[75] D. N. Kennedy, C. Haselgrove, S. M. Hodge, P. S. Rane, N. Makris, and J. A. Frazier. "CANDIShare: A Resource for Pediatric Neuroimaging Data". In: *Neuroinformatics* 10.3 (2012), pp. 319–322 (cit. on p. 43).

[76] D. P. Kingma and J. Ba. "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Y. Bengio and Y. LeCun. 2015 (cit. on p. 111).

[77] A. Klein and J. Tourville. "101 Labeled Brain Images and a Consistent Human Cortical Labeling Protocol". In: *Frontiers in Neuroscience* 6 (2012) (cit. on pp. 92, 98, 135).

[78] A. Klein and J. Tourville. “101 Labeled Brain Images and a Consistent Human Cortical Labeling Protocol”. In: *Frontiers in Neuroscience* 6 (2012) (cit. on pp. 94, 123, 135).

[79] F. Kong and S. C. Shadden. *Whole Heart Mesh Generation For Image-Based Computational Simulations By Learning Free-From Deformations*. 2021. arXiv: 2107.10839 [eess.IV] (cit. on pp. 102, 103, 105, 106, 112, 113).

[80] F. Kong, N. Wilson, and S. C. Shadden. *A Deep-Learning Approach For Direct Whole-Heart Mesh Reconstruction*. 2021. arXiv: 2102.07899 [eess.IV] (cit. on pp. 102–107, 110, 112).

[81] H. J. Kuijf, J. M. Biesbroek, J. De Bresser, et al. “Standardized assessment of automatic segmentation of white matter hyperintensities and results of the WMH segmentation challenge”. In: *IEEE transactions on medical imaging* 38.11 (2019), pp. 2556–2568 (cit. on pp. 56–62).

[82] B Landman and S Warfield. “MICCAI 2012 workshop on multi-atlas labeling”. In: *MICCAI*. 2012 (cit. on pp. 34, 42).

[83] B. Landman and S. Warfield. “MICCAI 2012 workshop on multi-atlas labeling”. In: *MICCAI Grand Challenge and Workshop on Multi-Atlas Labeling, CreateSpace Independent Publishing Platform, Nice, France*. 2012 (cit. on pp. 98, 106, 111).

[84] L. Lebrat, R. Santa Cruz, F. de Gournay, et al. “CorticalFlow: A Diffeomorphic Mesh Transformer Network for Cortical Surface Reconstruction”. In: *Advances in Neural Information Processing Systems* 34 (2021) (cit. on pp. 121, 123, 125, 127, 132, 142).

[85] T. Lei, R. Wang, Y. Wan, X. Du, H. Meng, and A. K. Nandi. “Medical Image Segmentation Using Deep Learning: A Survey.” In: (2020) (cit. on pp. 12, 26).

[86] J. P. Lerch, J. C. Pruessner, A. P. Zijdenbos, H. Hampel, S. J. Teipel, and A. C. Evans. “Focal decline of cortical thickness in Alzheimer’s disease identified by computational neuroanatomy.” In: *Cerebral cortex* 15 7 (2005), pp. 995–1001 (cit. on p. 132).

[87] T. Lewiner, H. Lopes, A. W. Vieira, and G. Tavares. “Efficient implementation of marching cubes’ cases with topological guarantees”. In: *Journal of graphics tools* 8.2 (2003), pp. 1–15 (cit. on pp. 92, 101, 102, 114).

[88] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. “Focal loss for dense object detection”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988 (cit. on p. 16).

[89] G. Litjens, T. Kooi, B. E. Bejnordi, et al. “A survey on deep learning in medical image analysis”. In: *Medical Image Analysis* 42 (2017), pp. 60–88 (cit. on pp. 3, 12, 26).

[90] T. J. Littlejohns, J. Holliday, L. M. Gibson, et al. “The UK Biobank imaging enhancement of 100,000 participants: rationale, data collection, management and future directions”. In: *Nature Communications* 11.1 (2020), pp. 1–12 (cit. on pp. 68, 74).

[91] L. Liu, J. M. Wolterink, C. Brune, and R. N. Veldhuis. “Anatomy-aided deep learning for medical image segmentation: a review”. In: *Physics in Medicine & Biology* 66.11 (2021), 11TR01 (cit. on p. 68).

[92] Z. Liu, L. Tong, L. Chen, et al. “Deep learning based brain tumor segmentation: a survey”. In: *Complex & Intelligent Systems* (2022), pp. 1–26 (cit. on p. 68).

[93] J. Long, E. Shelhamer, and T. Darrell. “Fully convolutional networks for semantic segmentation”. In: *CVPR*. 2015, pp. 3431–3440 (cit. on pp. 26, 33).

[94] W. E. Lorensen and H. E. Cline. “Marching Cubes: A High Resolution 3D Surface Construction Algorithm”. In: *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’87. New York, NY, USA: Association for Computing Machinery, 1987, 163–169 (cit. on p. 92).

[95] I. Loshchilov and F. Hutter. “Decoupled Weight Decay Regularization”. In: *International Conference on Learning Representations*. 2019 (cit. on p. 125).

[96] Q. Ma, L. Li, E. C. Robinson, B. Kainz, D. Rueckert, and A. Alansary. “CortexODE: Learning Cortical Surface Reconstruction by Neural ODEs”. In: *IEEE Transactions on Medical Imaging* (2022) (cit. on pp. 132, 142).

[97] Q. Ma, E. C. Robinson, B. Kainz, D. Rueckert, and A. Alansary. “PialNN: A Fast Deep Learning Framework for Cortical Pial Surface Reconstruction”. en. In: *Machine Learning in Clinical Neuroimaging*. Ed. by A. Abdulkadir, S. M. Kia, M. Habes, et al. Vol. 13001. Cham: Springer International Publishing, 2021, pp. 73–81 (cit. on pp. 102, 118, 121).

[98] J. R. Maclaren, Z. Han, S. B. Vos, N. J. Fischbein, and R. Bammer. “Reliability of brain volume measurements: A test-retest dataset”. In: *Scientific Data* 1 (2014) (cit. on pp. 97, 114, 135).

[99] L. Maier-Hein, B. Menze, et al. “Metrics reloaded: Pitfalls and recommendations for image analysis validation”. In: *arXiv. org* 2206.01653 (2022) (cit. on p. 30).

[100] D. S. Marcus, A. F. Fotenos, J. G. Csernansky, J. C. Morris, and R. L. Buckner. “Open access series of imaging studies: longitudinal MRI data in nondemented and demented older adults”. In: *Journal of cognitive neuroscience* 22.12 (2010), pp. 2677–2684 (cit. on p. 42).

[101] D. S. Marcus, T. H. Wang, J. Parker, J. G. Csernansky, J. C. Morris, and R. L. Buckner. “Open Access Series of Imaging Studies (OASIS): Cross-sectional MRI Data in Young, Middle Aged, Nondemented, and Demented Older Adults”. In: *Journal of Cognitive Neuroscience* 19.9 (Sept. 2007), pp. 1498–1507 (cit. on p. 97).

[102] J. C. Mazziotta, A. W. Toga, A. Evans, P. Fox, J. Lancaster, et al. “A probabilistic atlas of the human brain: theory and rationale for its development”. In: *Neuroimage* 2.2 (1995), pp. 89–101 (cit. on p. 98).

[103] T. McInerney and D. Terzopoulos. “Deformable models in medical image analysis: a survey”. In: *Medical image analysis* 1.2 (1996), pp. 91–108 (cit. on p. 13).

[104] S. Mehta, E. Mercan, J. Bartlett, D. Weaver, J. G. Elmore, and L. Shapiro. “Y-Net: joint segmentation and classification for diagnosis of breast biopsy images”. In: *MICCAI*. Springer. 2018 (cit. on p. 70).

[105] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. “Occupancy Networks: Learning 3D Reconstruction in Function Space”. In: *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019 (cit. on p. 102).

[106] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. “Discrete Differential-Geometry Operators for Triangulated 2-Manifolds”. In: *Visualization and Mathematics III*. Ed. by H.-C. Hege and K. Polthier. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 35–57 (cit. on p. 109).

[107] M. I. Miller, A. B. Massie, J. Ratnanather, K. N. Botteron, and J. G. Csernansky. “Bayesian Construction of Geometrically Based Cortical Thickness Metrics”. In: *NeuroImage* 12.6 (2000), pp. 676–687 (cit. on p. 115).

[108] F. Milletari, S.-A. Ahmadi, C. Kroll, et al. “Hough-CNN: deep learning for segmentation of deep brain regions in MRI and ultrasound”. In: *Computer Vision and Image Understanding* 164 (2017), pp. 92–102 (cit. on p. 33).

[109] F. Milletari, N. Navab, and S.-A. Ahmadi. “V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation”. In: *2016 Fourth International Conference on 3D Vision (3DV)*. 2016, pp. 565–571 (cit. on pp. 16, 26–29, 33, 43).

[110] P. Mlynarski, H. Delingette, A. Criminisi, and N. Ayache. “Deep learning with mixed supervision for brain tumor segmentation”. In: *Journal of Medical Imaging* 6.3 (2019), p. 034002 (cit. on pp. 70, 76–78).

[111] M. Modat, D. M. Cash, P. Daga, G. P. Winston, J. S. Duncan, and S. Ourselin. “Global image registration using a symmetric block-matching approach”. In: *Journal of medical imaging* 1.2 (2014), pp. 024003–024003 (cit. on p. 98).

[112] J. H. Morrison and P. R. Hof. “Life and death of neurons in the aging brain”. In: *Science* 278.5337 (1997), pp. 412–419 (cit. on p. 88).

[113] T. Nair, D. Precup, D. L. Arnold, and T. Arbel. “Exploring Uncertainty Measures in Deep Networks for Multiple Sclerosis Lesion Detection and Segmentation”. In: *Medical Image Computing and Computer Assisted Intervention–MICCAI 2018: 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part I*. Springer. 2018, pp. 655–663 (cit. on p. 56).

[114] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa. “Laplacian mesh optimization”. In: *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia 2006, Kuala Lumpur, Malaysia, November 29 - December 2, 2006*. Ed. by Y. T. Lee, S. M. H. Shamsuddin, D. Gutierrez, and N. M. Suaib. ACM, 2006, pp. 381–389 (cit. on pp. 109, 110).

[115] D. Nie, Y. Gao, L. Wang, and D. Shen. “ASDNet: attention based semi-supervised deep networks for medical image segmentation”. In: *International conference on medical image computing and computer-assisted intervention*. Springer. 2018, pp. 370–378 (cit. on p. 57).

[116] S. Nikolov, S. Blackwell, A. Zverovitch, et al. “Deep learning to achieve clinically applicable segmentation of head and neck anatomy for radiotherapy”. In: *arXiv preprint arXiv:1809.04430* (2018) (cit. on pp. 31, 45).

[117] O. Oktay, E. Ferrante, K. Kamnitsas, et al. “Anatomically constrained neural networks (ACNNs): application to cardiac image enhancement and segmentation”. In: *IEEE transactions on medical imaging* 37.2 (2017), pp. 384–395 (cit. on p. 68).

[118] D. A. Ortendahl and J. W. Carlson. “Segmentation of magnetic resonance images using fuzzy clustering”. In: *Information processing in medical imaging*. Springer. 1988, pp. 91–106 (cit. on p. 13).

[119] S. Osher and J. A. Sethian. “Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations”. In: *Journal of computational physics* 79.1 (1988), pp. 12–49 (cit. on p. 13).

[120] N. Otsu. “A threshold selection method from gray-level histograms”. In: *IEEE transactions on systems, man, and cybernetics* 9.1 (1979), pp. 62–66 (cit. on p. 13).

[121] S. Ourselin, A. Roche, G. Subsol, X. Pennec, and N. Ayache. “Reconstructing a 3D structure from serial histological sections”. In: *Image and vision computing* 19.1-2 (2001), pp. 25–31 (cit. on p. 98).

[122] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. “Deepsdf: Learning continuous signed distance functions for shape representation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 165–174 (cit. on p. 102).

[123] A. Paszke, S. Gross, F. Massa, et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035 (cit. on p. 126).

[124] S. Pereira, A. Pinto, J. Amorim, A. Ribeiro, V. Alves, and C. A. Silva. “Adaptive feature recombination and recalibration for semantic segmentation with Fully Convolutional Networks”. In: *IEEE transactions on medical imaging* (2019) (cit. on pp. 36, 47).

[125] P. Perona and J. Malik. “Scale-space and edge detection using anisotropic diffusion”. In: *IEEE Transactions on pattern analysis and machine intelligence* 12.7 (1990), pp. 629–639 (cit. on p. 12).

[126] D. L. Pham, C. Xu, and J. L. Prince. “Current methods in medical image segmentation”. In: *Annual review of biomedical engineering* 2.1 (2000), pp. 315–337 (cit. on p. 12).

[127] N. Prins and P. Scheltens. “White matter hyperintensities, cognitive impairment and dementia: An update”. In: *Nature reviews. Neurology* 11 (Feb. 2015) (cit. on p. 55).

[128] S. Rabanser, O. Shchur, and S. Günnemann. “Introduction to tensor decompositions and their applications in machine learning”. In: *arXiv preprint arXiv:1711.10781* (2017) (cit. on p. 34).

[129] A. Radford, L. Metz, and S. Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *arXiv preprint arXiv:1511.06434* (2015) (cit. on p. 57).

[130] N. Ravi, J. Reizenstein, D. Novotny, et al. “Accelerating 3D Deep Learning with PyTorch3D”. In: *arXiv:2007.08501* (2020) (cit. on pp. 95, 105, 126).

[131] L. Raz, J. Knoefel, and K. Bhaskar. “The neuropathology and cerebrovascular mechanisms of dementia”. In: *Journal of cerebral blood flow and metabolism: official journal of the International Society of Cerebral Blood Flow and Metabolism* 36 (July 2015) (cit. on p. 55).

[132] A.-M. Rickmann, F. Bongratz, S. Pölsterl, I. Sarasua, and C. Wachinger. “Joint Reconstruction and Parcellation of Cortical Surfaces”. In: *Machine Learning in Clinical Neuroimaging: 5th International Workshop, MLCN 2022, Held in Conjunction with MICCAI 2022, Singapore, September 18, 2022, Proceedings*. Springer. 2022, pp. 3–12 (cit. on pp. 6, 85, 121).

[133] A.-M. Rickmann, F. Bongratz, and C. Wachinger. “Vertex Correspondence in Cortical Surface Reconstruction”. In: *Medical Image Computing and Computer Assisted Interventions*. Springer. 2023 (cit. on p. 6).

[134] A.-M. Rickmann, A. G. Roy, I. Sarasua, N. Navab, and C. Wachinger. “‘Project & excite’modules for segmentation of volumetric medical scans”. In: *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part II 22*. Springer. 2019, pp. 39–47 (cit. on p. 34).

[135] A.-M. Rickmann, A. G. Roy, I. Sarasua, and C. Wachinger. “Recalibrating 3D convnets with project & excite”. In: *IEEE transactions on medical imaging* 39.7 (2020), pp. 2461–2471 (cit. on pp. 5, 33, 34).

[136] A.-M. Rickmann, J. Senapati, O. Kovalenko, A. Peters, F. Bamberg, and C. Wachinger. “AbdomenNet: deep neural network for abdominal organ segmentation in epidemiologic imaging studies”. In: *BMC Medical Imaging* 22.1 (2022), pp. 1–11 (cit. on p. 74).

[137] A.-M. Rickmann, M. Xu, T. N. Wolf, O. Kovalenko, and C. Wachinger. “HALOS: Hallucination-Free Organ Segmentation After Organ Resection Surgery”. In: *Information Processing in Medical Imaging*. Ed. by A. Frangi, M. de Bruijne, D. Wassermann, and N. Navab. Cham: Springer Nature Switzerland, 2023, pp. 667–678 (cit. on p. 5).

[138] J. M. Roe, D. Vidal-Piñeiro, Ø. Sørensen, et al. “Asymmetric thinning of the cerebral cortex across the adult lifespan is accelerated in Alzheimer’s disease”. In: *Nature communications* 12.1 (2021), p. 721 (cit. on pp. 132, 137).

[139] O. Ronneberger, P. Fischer, and T. Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *MICCAI*. 2015 (cit. on pp. 26, 27, 29, 103).

[140] O. Ronneberger, P. Fischer, and T. Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *MICCAI*. Springer. 2015, pp. 234–241 (cit. on pp. 13, 33, 60).

[141] H. R. Roth, H. Oda, Y. Hayashi, et al. “Hierarchical 3D fully convolutional networks for multi-organ segmentation”. In: *arXiv preprint arXiv:1704.06382* (2017) (cit. on pp. 68, 69).

[142] A. G. Roy, S. Conjeti, N. Navab, C. Wachinger, and A. D. N. Initiative. “QuickNAT: A fully convolutional network for quick and accurate segmentation of neuroanatomy”. In: *NeuroImage* 186 (2019), pp. 713–727 (cit. on p. 68).

[143] A. G. Roy, S. Conjeti, N. Navab, and C. Wachinger. “QuickNAT: A fully convolutional network for quick and accurate segmentation of neuroanatomy”. In: *NeuroImage* 186 (2019), pp. 713–727 (cit. on pp. 29, 33, 45, 102).

[144] A. G. Roy, N. Navab, and C. Wachinger. “Recalibrating Fully Convolutional Networks With Spatial and Channel “Squeeze and Excitation” Blocks”. In: *IEEE TMI* 38.2 (2019), pp. 540–549 (cit. on pp. 34, 36–39, 48, 49, 52, 61).

[145] F. Rusak, R. Santa Cruz, L. Lebrat, et al. “Quantifiable brain atrophy synthesis for benchmarking of cortical thickness estimation methods”. In: *Medical Image Analysis* 82 (2022), p. 102576 (cit. on p. 143).

[146] R. Santa Cruz, L. Lebrat, D. Fu, et al. *CorticalFlow++*. `https://bitbucket.csiro.au/projects/CRCPMAX/repos/corticalflow/browse`. Accessed: 2023-03-04 (cit. on p. 135).

[147] R. Santa Cruz, L. Lebrat, D. Fu, et al. “CorticalFlow++: Boosting Cortical Surface Reconstruction Accuracy, Regularity, and Interoperability”. In: *Medical Image Computing and Computer Assisted Intervention–MICCAI 2022: 25th International Conference, Singapore, September 18–22, 2022, Proceedings, Part V*. Springer. 2022, pp. 496–505 (cit. on pp. 132, 135, 142).

[148] P. Schmidt. “Bayesian inference for structured additive regression models for large-scale problems with applications to medical imaging”. PhD thesis. Ludwig-Maximilians-Universität München, 2017 (cit. on p. 56).

[149] F. Shamshad, S. Khan, S. W. Zamir, et al. “Transformers in medical imaging: A survey”. In: *Medical Image Analysis* (2023), p. 102802 (cit. on pp. 14, 82).

[150] M. E. Shaw, P. S. Sachdev, K. J. Anstey, and N. Cherbuin. “Age-related cortical thinning in cognitively healthy individuals in their 60s: the PATH Through Life study”. In: *Neurobiology of Aging* 39 (2016), pp. 202–209 (cit. on p. 132).

[151] I. Shin, S. Woo, F. Pan, and I. S. Kweon. “Two-Phase Pseudo Label Densification for Self-training Based Domain Adaptation”. In: *Computer Vision – ECCV 2020*. Cham: Springer International Publishing, 2020, pp. 532–548 (cit. on p. 57).

[152] V. Singh, H. Chertkow, J. P. Lerch, A. C. Evans, A. E. Dorr, and N. J. Kabani. “Spatial patterns of cortical thinning in mild cognitive impairment and Alzheimer’s disease”. In: *Brain* 129.11 (2006), pp. 2885–2893 (cit. on pp. 132, 137).

[153] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein. “Implicit neural representations with periodic activation functions”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 7462–7473 (cit. on p. 102).

[154] E. Smith, S. Fujimoto, A. Romero, and D. Meger. “GEOMetrics: Exploiting Geometric Structure for Graph-Encoded Objects”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by K. Chaudhuri and R. Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. Long Beach, California, USA: PMLR, 2019, pp. 5866–5876 (cit. on p. 109).

[155] L. N. Smith. “Cyclical Learning Rates for Training Neural Networks”. In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2017, pp. 464–472 (cit. on p. 125).

[156] S. M. Smith, M. Jenkinson, M. W. Woolrich, et al. “Advances in functional and structural MR image analysis and implementation as FSL”. In: *Neuroimage* 23 (2004), S208–S219 (cit. on p. 101).

[157] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 1929–1958 (cit. on p. 30).

[158] M. Suzuki, M. G. Linguraru, and K. Okada. “Multi-organ segmentation with missing organs in abdominal CT images”. In: *MICCAI*. Springer. 2012 (cit. on pp. 68, 69).

[159] N. Tajbakhsh, L. Jeyaseelan, Q. Li, J. N. Chiang, Z. Wu, and X. Ding. “Embracing imperfect datasets: A review of deep learning solutions for medical image segmentation”. In: *Medical Image Analysis* 63 (2020), p. 101693 (cit. on p. 57).

[160] S. Tilborghs, J. Bertels, D. Robben, D. Vandermeulen, and F. Maes. “The Dice Loss in the Context of Missing or Empty Labels: Introducing and”. In: *MICCAI*. Springer. 2022 (cit. on pp. 69, 76, 77).

[161] O. Jimenez-del Toro, H. Müller, M. Krenn, et al. “Cloud-based evaluation of anatomical structure segmentation and landmark detection algorithms: VISCERAL anatomy benchmarks”. In: *IEEE transactions on medical imaging* 35.11 (2016), pp. 2459–2475 (cit. on pp. 34, 43).

[162] M. A. Tubi, F. W. Feingold, D. Kothapalli, et al. “White matter hyperintensities and their relationship to cognition: Effects of segmentation algorithm”. In: *NeuroImage* 206 (2020), p. 116327 (cit. on p. 61).

[163] N. J. Tustison, B. B. Avants, P. A. Cook, et al. “N4ITK: improved N3 bias correction”. In: *IEEE transactions on medical imaging* 29.6 (2010), pp. 1310–1320 (cit. on pp. 61, 74).

[164] D. Ulyanov, A. Vedaldi, and V. Lempitsky. “Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis”. In: *CVPR*. 2017, pp. 6924–6932 (cit. on p. 44).

[165] Q. Vanderbecq, E. Xu, S. Ströer, et al. “Comparison and validation of seven white matter hyperintensities segmentation software in elderly patients”. In: *NeuroImage: Clinical* 27 (July 2020), p. 102357 (cit. on pp. 56–58, 62, 63).

[166] J. Vollmer, R. Mencl, and H. Muller. “Improved Laplacian Smoothing of Noisy Surface Meshes”. en. In: *Computer Graphics Forum* 18.3 (Sept. 1999), pp. 131–138 (cit. on p. 106).

[167] C. Wachinger, M. Reuter, and T. Klein. “DeepNAT: Deep convolutional neural network for segmenting neuroanatomy”. In: *NeuroImage* 170 (2018), pp. 434–445 (cit. on pp. 33, 35).

[168] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. “Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images”. In: *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI*. Vol. 11215. Lecture Notes in Computer Science. Springer, 2018, pp. 55–71 (cit. on pp. 103, 105–107, 110, 113).

[169] N. Wang, Y. Zhang, Z. Li, et al. “Pixel2Mesh: 3D mesh model generation via image guided deformation”. In: *IEEE transactions on pattern analysis and machine intelligence* (2020) (cit. on p. 102).

[170] W. Wang, D. Ceylan, R. Mech, and U. Neumann. “3DN: 3D Deformation Network”. In: *CVPR*. 2019 (cit. on pp. 103, 113).

[171] Y. Wang, Y. Zhou, W. Shen, S. Park, E. K. Fishman, and A. L. Yuille. “Abdominal multi-organ segmentation with organ-attention networks and statistical fusion”. In: *Medical image analysis* 55 (2019), pp. 88–102 (cit. on p. 69).

[172] Y. Wang, J. Peng, and Z. Zhang. “Uncertainty-aware pseudo label refinery for domain adaptive semantic segmentation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 9092–9101 (cit. on p. 82).

[173] C. Wen, Y. Zhang, Z. Li, and Y. Fu. “Pixel2Mesh++: Multi-View 3D Mesh Generation via Deformation”. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (2019), pp. 1042–1051 (cit. on p. 102).

[174] U. Wickramasinghe, E. Remelli, G. Knott, and P. Fua. “Voxel2Mesh: 3D Mesh Model Generation from Volumetric Data”. In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*. Ed. by A. L. Martel, P. Abolmaesumi, D. Stoyanov, et al. Cham: Springer International Publishing, 2020, pp. 299–308 (cit. on pp. 14, 99, 102, 103, 105–107, 110, 112, 113).

[175] T. N. Wolf, S. Pölsterl, C. Wachinger, A. D. N. Initiative, et al. “DAFT: A universal module to interweave tabular data and 3D images in CNNs”. In: *NeuroImage* 260 (2022), p. 119505 (cit. on pp. 69, 70, 73).

[176] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon. “Cbam: Convolutional block attention module”. In: *ECCV*. 2018, pp. 3–19 (cit. on pp. 36, 37, 39, 48, 49, 52).

[177] Y.-H. Wu, S.-H. Gao, J. Mei, et al. “Jcs: An explainable covid-19 diagnosis system by joint classification and segmentation”. In: *IEEE Transactions on Image Processing* 30 (2021), pp. 3113–3126 (cit. on p. 70).

[178] Y. Xia, F. Liu, D. Yang, et al. “3d semi-supervised learning with uncertainty-aware multi-view co-training”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2020, pp. 3646–3655 (cit. on p. 57).

[179] Q. Xu, W. Wang, D. Ceylan, R. Mech, and U. Neumann. “DISN: Deep Implicit Surface Network for High-quality Single-view 3D Reconstruction”. In: *NeurIPS*. 2019 (cit. on p. 102).

[180] L. Yu, S. Wang, X. Li, C.-W. Fu, and P.-A. Heng. “Uncertainty-aware self-ensembling model for semi-supervised 3D left atrium segmentation”. In: *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer. 2019, pp. 605–613 (cit. on p. 57).

[181] G. Zeng, X. Yang, J. Li, L. Yu, P.-A. Heng, and G. Zheng. “3D U-net with Multi-level Deep Supervision: Fully Automatic Segmentation of Proximal Femur in 3D MR Images”. In: *Machine Learning in Medical Imaging*. Ed. by Q. Wang, Y. Shi, H.-I. Suk, and K. Suzuki. Cham: Springer International Publishing, 2017, pp. 274–282 (cit. on p. 103).

[182] Z. Zhang, Q. Liu, and Y. Wang. “Road Extraction by Deep Residual U-Net”. In: *IEEE Geoscience and Remote Sensing Letters* 15.5 (2018), pp. 749–753 (cit. on p. 103).

[183] F. Zhao, Z. Wu, L. Wang, et al. “Unsupervised Learning for Spherical Surface Registration”. In: *Machine Learning in Medical Imaging*. Ed. by M. Liu, P. Yan, C. Lian, and X. Cao. Cham: Springer International Publishing, 2020, pp. 373–383 (cit. on p. 110).

[184] F. Zhao, S. Xia, Z. Wu, et al. “Spherical U-Net on cortical surfaces: methods and applications”. In: *Information Processing in Medical Imaging: 26th International Conference, IPMI 2019, Hong Kong, China, June 2–7, 2019, Proceedings 26*. Springer. 2019, pp. 855–866 (cit. on p. 14).

[185] H. Zheng, S. M. Motch Perrine, M. K. Pitirri, et al. “Cartilage Segmentation in High-Resolution 3D Micro-CT Images via Uncertainty-Guided Self-training with Very Sparse Annotation”. In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*. Cham: Springer, 2020, pp. 802–812 (cit. on p. 57).

[186] Y. Zhou, Z. Li, S. Bai, et al. “Prior-aware neural network for partially-supervised multi-organ segmentation”. In: *ICCV*. 2019 (cit. on p. 68).

[187] W. Zhu, Y. Huang, L. Zeng, et al. “AnatomyNet: Deep learning for fast and fully automated whole-volume segmentation of head and neck anatomy”. In: *Medical physics* 46.2 (2019), pp. 576–589 (cit. on pp. 36, 37, 48, 49, 52).

[188] B. Zoph, G. Ghiasi, T.-Y. Lin, et al. “Rethinking Pre-training and Self-training”. In: *Advances in Neural Information Processing Systems* 33 (2020) (cit. on pp. 57, 58).

[189] Y. Zou, Z. Yu, B. V. Kumar, and J. Wang. “Unsupervised Domain Adaptation for Semantic Segmentation via Class-Balanced Self-Training”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 289–305 (cit. on p. 57).

[190] Y. Zou, Z. Yu, X. Liu, B. V. Kumar, and J. Wang. “Confidence Regularized Self-Training”. In: *The IEEE International Conference on Computer Vision (ICCV)*. 2019 (cit. on p. 57).

List of Terms

ASSD Average Symmetric Surface Distance. 112–114, 116, 126, 127, 135, 137, 174, 175

CNN Convolutional Neural Network. 13–15, 25, 26, 30, 34, 37, 56, 73, 95, 102, 103, 105, 107, 112, 113, 133, 134

CSF cerebrospinal fluid. 91

CT Computed Tomography. 5, 10–12, 17, 29, 33, 43, 51, 69, 147

DAFT Dynamic Affine Feature Map Transform. 70, 73

DK Desikan-Killiany. 94

DKT Desikan-Killiany-Tourville. 94, 123, 131, 135, 137

F-CNN Fully-Convolutional Neural Network. 25–29, 33–36, 42, 43, 46, 47, 52, 81, 103, 122

FLAIR Fluid-Attenuated Inversion Recovery. 10, 55, 61, 65, 168

GM gray matter. 91

GNN Graph Neural Network. 14, 15, 95, 103–105, 112, 113, 124, 127, 170

HALOS Hallucination-free Organ Segmentation. 5, 24, 78, 80–82, 147

HD Hausdorff Distance. 31

HD-90 90th-percentile Hausdorff Distance. 112–114, 116, 126, 127, 174, 175

ICP Iterative Closest-Point algorithm. 115

IoU Intersection over Union. 31

LPA Lesion Prediction Algorithm. 56, 58, 62–65

MLP Multi-Layer Perceptron. 14, 102

MRI Magnetic Resonance Imaging. 3–6, 10–12, 17, 29, 33, 42, 61, 69, 74, 85, 88–91, 93–99, 102, 103, 118, 121–123, 133, 135, 143, 167, 169, 171

PE Project & Excite. 34, 43, 45–52, 81

ReLU Rectified Linear Unit. 28, 95, 103–105

RMSD Root Mean Square Deviation. 135–138, 171, 175

SDF Signed Distance Function. 13, 93, 96, 123

STRUDEL Self-training with Uncertainty Guided Label refinement. 24, 56, 60, 62–65, 81, 82, 168

SVD Small Vessel Disease. 55

T1w T1 weighted MRI scams, displaying fat as bright and water as dark. 10, 88, 89, 91, 97, 98, 135, 169

T2w T2 weighted MRI scams, displaying fat as dark and water as bright. 10

TRT Test-Retest dataset. 97, 114, 115, 135, 138, 171

UDA Unsupervised Domain Adaptation. 56, 57, 65

V2C Vox2Cortex. 121, 126, 127, 132–134, 136–138, 171

V2CC Vox2Cortex with Correspondence. 133, 134, 136–138, 141–143, 171

WM white matter. 91, 100, 102, 115, 169

WMH White matter hyperintensity. 55–62

List of Figures

List of Tables