Automatic Analysis of Cortical Areas in Whole Brain Histological Sections using Convolutional Neural Networks

Inaugural-Dissertation

zur Erlangung des Doktorgrades
der Mathematisch-Naturwissenschaftlichen Fakultät
der Heinrich-Heine-Universität Düsseldorf

vorgelegt von

Hannah Spitzer
aus Berlin

Düsseldorf, Februar 2019

aus dem Institut für Machine Learning
der Heinrich-Heine-Universität Düsseldorf

Gedruckt mit Genehmigung der
Mathematisch-Naturwissenschaftlichen Fakultät der
Heinrich-Heine-Universität Düsseldorf

Berichterstatter:

1. Prof. Dr. Stefan HARMELING

2. Prof. Dr. med. Katrin AMUNTS

Tag der mündlichen Prüfung:
10.02.2020

Abstract

The segregation of the human brain in cytoarchitectonic areas is an important prerequisite for the allocation of functional imaging, physiological, connectivity, molecular and genetic data to structurally well-defined entities of the human brain. Cytoarchitecture describes the spatial distribution of cell bodies and their shape and size, and is most appropriately studied at microscopic resolution based on cell-body stained histological sections. To determine boundaries between cytoarchitectonic areas, an observer-independent method that uses image analysis and multivariate statistical tools to capture changes in the distribution of cell bodies is already established. Nowadays, new technologies for high-throughput microscopy allow rapid digitization of histological sections, which increases the need for a fully automatic brain area segmentation method. This task is extremely challenging due to the high inter-individual variability in cortical folding, sectioning artifacts, limited labeled training data, and the need for large input sizes for automatic methods.

This work shows that convolutional neural networks, a special class of deep artificial neural networks, are suitable for automatic brain area segmentation. It introduces a semantic segmentation model that combines texture input given by high-resolution extracts of the histological sections with prior knowledge given by an existing probabilistic brain area atlas, the JuBrain atlas. This atlas prior helps the model to localize the texture input in the brain and allows it to make topologically correct brain area predictions. To overcome the limited amount of brain area annotations, the model can be pre-trained on a modified task for which training data is easier to obtain. Pre-training the model on a self-supervised task based on predicting the spatial distance between image patches extracted from sections of the same brain significantly increases the segmentation performance and enables the prediction of several brain areas in previously unseen brains.

The self-supervised model learns a compact internal feature representation of the input using the inherent structure of the brain, without having explicit access to the concept of brain areas. Extensive evaluations indicate that these features encode cytoarchitectonic properties. This is remarkable result which allows the data-driven analysis of the structure of the entire brain. Although the presented model is not yet robust enough for automatic annotation of all areas in complete human brains, it is already leveraged for practical use by training specialized multi-scale models to propagate brain area labels from annotated sections to spatially close sections. This workflow has the potential to speed up current brain mapping projects by reducing the workload of the neuroscientists and produces previously unattainable high-resolution 3D views of single brain areas.

Zusammenfassung

Die Unterteilung des menschlichen Gehirns in zytoarchitektonische Areale ist eine wichtige Voraussetzung für die Zuordnung von funktionalen Bildgebungs-, Konnektivitäts-, physiologischen, molekularen und genetischen Daten zu strukturell wohldefinierten Bereichen im Gehirn. Zytoarchitektur beschreibt die räumliche Verteilung von Zellkörpern sowie deren Form und Größe und wird am Besten mit mikroskopischer Auflösung auf zellkörpergefärbten histologischen Schnitten untersucht. Um die Grenzen zwischen zytoarchitektonischen Arealen zu bestimmen, wurde bereits eine beobachterunabhängige Methode etabliert, die mithilfe von Bildanalyse und multivariaten statistischen Analysen Änderungen in der Verteilung von Zellkörpern erfasst. Heutzutage ermöglichen neue Technologien für die Hochdurchsatzmikroskopie eine schnellere Digitalisierung histologischer Schnitte, was die Notwendigkeit einer vollautomatischen Methode zur Segmentierung von zytoarchitektonischen Arealen erhöht. Diese Aufgabe ist aufgrund der hohen interindividuellen Variabilität der kortikalen Faltung, der Artefakte die während des Schneidens auftreten, der begrenzten Trainingsdaten und der Notwendigkeit großer Bildausschnitte zum Training äußerst anspruchsvoll.

Diese Dissertation zeigt, dass Convolutional Neural Networks, eine spezielle Klasse von künstlichen neuronalen Netzen, für die automatische Segmentierung von zytoarchitektonischen Arealen geeignet sind. Es wird ein Modell zur Segmentierung von Arealen vorgestellt, welches die Textur hochaufgelößter Bildausschnitte der histologischen Schnitte mit Vorwissen aus einem probabilistischen Gehirnatlas (JuBrain Atlas) kombiniert. Dieser sogenannte Atlas Prior hilft dem Modell, den Bildausschnitt im Gehirn zu lokalisieren, und ermöglicht es ihm, topologisch korrekte Vorhersagen für die Gehirnareale zu treffen. Da nur eine berenzte Anzahl von Annotationen für Gehirnareale vorhanden ist, kann das Modell auf einer modifizierten Aufgabe vortrainiert werden, für die Trainingsdaten leichter zu erhalten sind. Das Vortrainieren des Modells auf einer selbstüberwachten Aufgabe basierend auf der Vorhersage des räumlichen Abstands zwischen zwei Bildern, die aus histologischen Schnitten desselben Gehirnvolumens extrahiert wurden, erhöht die Qualität der Segmentierungen erheblich und ermöglicht die Vorhersage mehrerer Gehirnareale in zuvor nicht verwendeten Gehirnen.

Das selbstüberwachte Modell lernt anhand der inhärenten Struktur des Gehirns eine kompakte interne Merkmalsdarstellung der Eingabebilder, ohne explizit das Konzept der Gehirnareale zu benötigen. Umfangreiche Auswertungen zeigen, dass diese Merkmale zytoarchitektonische Eigenschaften enkodieren. Dies ist ein bemerkenswertes Ergebnis welches uns die datengetriebene Analyse der Struktur des gesamten

Gehirns erlaubt. Obwohl das vorgestellte Modell noch nicht robust genug ist, um alle Areale in menschlichen Gehirnen automatisch zu segmentieren, wird es bereits für den praktischen Einsatz genutzt: Spezialisierte Modelle werden trainiert um Annotierungen von Arealen auf räumlich nahe Schnitte des gleichen Gehirns zu übertragen. Dieses Verfahren hat das Potenzial, aktuelle Annotierungsprojekte zu beschleunigen, indem die Arbeitsbelastung der Neurowissenschaftler reduziert wird, und erzeugt bisher unerreichte hochauflösende 3D-Ansichten einzelner Gehirnareale.

Acknowledgements

This interdisciplinary doctoral thesis was developed in the *Big Data Analytics* group of Dr. Timo Dickscheid at the *Institute of Neuroscience and Medicine* (INM-1), Forschungszentrum Jülich (FZJ), and submitted to the *Heinrich-Heine-University Düsseldorf* (HHU) to obtain a doctorate in computer science. It has received funding from the Helmholtz Association through the Helmholtz Portfolio Theme Supercomputing and Modeling for the Human Brain, from the European Unions Horizon 2020 Research and Innovation Programme under Grant Agreement No. 7202070 (HBP SGA1) and Grant Agreement No. 785907 (HBP SGA2). Computing time was granted through JARA-HPC on the supercomputer JURECA at Forschungszentrum Jülich. Besides these institutions, many people contributed to the success of this project.

First, I would like to thank Dr. Timo Dickscheid (INM-1, FZJ). He supported this endeavor from the very beginning, always encouraged me to develop my ideas further, and did everything he could to give me the time and freedom that I needed to work on this project. He made sure to always have time to discuss successes and failures and provided insightful ideas for new directions. I especially appreciate the time he took to proof-read drafts of this document and provide suggestions to make it even better.

My sincere thanks goes to my thesis advisor, Prof. Stefan Harmeling (Institute of Machine Learning, HHU) for his support in the last years. He provided a fresh point of view to our discussions and made valuable suggestions regarding machine learning methods and the evaluation of the self-supervised features in Section 6.4. His comments on several drafts of papers and this thesis helped to make my writing clearer and more concise.

I am very grateful to my other thesis advisor, Prof. Katrin Amunts (INM-1, FZJ), who introduced me to the fascinating world of microscopic brain structure. She was very supportive of the entire project and took time out of her busy schedule to discuss the neuroscientific aspects of this thesis. I appreciate the various opportunities that she gave me to travel to conferences and to our collaboration partners at Montreal Neurological Institute (MNI), McGill University in Montréal.

It was through the cooperation with the MNI, that I met Konrad Wagstyl (University of Cambridge) with whom I collaborated on using profiles to identify brain areas

(Chapter 4). We worked closely together for about two months which were spread out in small chunks of time over three years. I am very grateful for the chance to work together and still amazed at how much you can get done in a few days. Besides this collaboration, Konrad Wagstyl also assisted with using the registration files of the BigBrain to calculate gray/white matter masks (Section 2.3.3) and provided the initial scripts for visualizing scalars on (inflated) brain surfaces (used in Section 6.4).

At this point I would like to thank Prof. Alan Evans and the lab members of the ACELab in the MNI, McGill University, for welcoming me to their lab for several shorter stays and one month-long stay. In particular, I am obliged to Claude Lepage (MNI, McGill University) who provided the BigBrain reconstruction and assisted me in using it. Thanks is also due to Hartmut Mohlberg (INM-1, FZJ) for sharing the manual labels of brain areas acquired throughout the years for the JuBrain atlas, which allowed me to create training datasets for the automatic brain area segmentation model presented in this thesis.

My gratitude goes to my masters student Christian Schiffer (INM-1, FZJ) who assisted me in several projects related to brain area segmentation and discussed new deep learning techniques with me. He worked on the label propagation (Chapter 7) and he improved the gray/white matter segmentation used in several parts of this thesis (Section 2.3.4). Christian Schiffer also played an integral role in speeding up the deep learning code and parallelizing it to run on several GPUs on the supercomputer. I would like to thank the team of the Jülich Supercomputing Centre (JSC), FZJ, for providing such an amazing infrastructure and support that makes the training of such large deep learning models possible. In particular, I am grateful for the support of Dr. Andreas Herten in using the JURON supercomputer (*JURON* 2018) and would like to thank Dr. Lena Oden for implementing efficient caching solutions that made training even faster.

I would like to thank Kai Kiwitz (Medical Faculty, HHU) for answering many of my neuroscience and statistics related questions. His detailed analyzes of the feature map activations shown in Figure 5.9, interpretation of the clusters from Figure 6.13, and helpful comments on drafts of this document were very much appreciated. I am also grateful for Dr. Sebastian Bludau's (INM-1, FZJ) patient explanations of histological brain areas and how to identify them manually. I would like to thank both Sarah Haas and Christian Schiffer (INM-1, FZJ) for proof-reading and providing valuable comments on the final draft of this document.

Furthermore, I would like to thank the regulars of our lunch table, the running club, and the cinema evenings for providing professional and moral support, and all members of the INM-1 for providing such a pleasant working atmosphere at the institute. Last but not least my gratitude goes to my family for their support and encouragement during my doctoral studies. Thank you.

Contents

1 Introduction

Understanding the organization of healthy human brains is the key towards understanding how neurodegenerative diseases like Parkinson's and Alzheimer's disease affect the brain. Brain organization can be studied at different levels: starting from low-resolution magnetic resonance imaging (MRI) and functional magnetic resonance imaging (fMRI) studies carried out in-vivo and at a population scale, towards high-resolution single- or few-brain studies using cell-body stained histological post-mortem sections. Histological analyses of cytoarchitecture allow the parcellation of the cortex of the brain into cytoarchitectonic areas based on the distribution, shape, and size of neuronal cell bodies (Amunts and Zilles, 2015). A reference atlas containing these cytoarchitectonic areas is essential for the precise allocation of functional imaging, physiological, connectivity, molecular and genetic data to structurally well-defined entities of the human brain (Amunts, Schleicher, et al., 2007). Current methods for the analysis of cytoarchitecture are based on image analysis and multivariate statistical tools (Schleicher, Amunts, et al., 1999). However, they do not scale with new high-throughput microscopy techniques that allow rapid digitization of histological sections at 1 µm resolution.

Deep learning has enjoyed a surge in popularity in recent years due to its ability to reach and surpass human performance in different areas. Noteworthy examples of the power of deep learning are the better-than-human performance of convolutional neural networks (CNNs) on image classification (He et al., 2016) and skin cancer recognition (Esteva et al., 2017) and the recent win a deep neural network scored in the game Go over a human champion (Silver et al., 2016). Due to its ability to learn to recognize complex patterns in image data, deep learning is well suited to automatically recognize and classify cortical areas. Recent research in unsupervised and self-supervised pre-training (Doersch, Gupta, et al., 2015; Noroozi and Favaro, 2016) and transfer learning (Yosinski et al., 2014) with deep neural networks gives the possibility to train deep learning models on a small training dataset.

The aim of this thesis is to show that deep learning can be used to recognize cytoarchitectonic areas in high-resolution histological cell-body stained sections. Section 1.1 highlights the challenges and potential benefits associated with this aim. The concrete objectives covered by this thesis are defined in Section 1.2. Finally, Section 1.3 summarizes the neural network architectures developed for this thesis and gives a short outline of the following chapters.

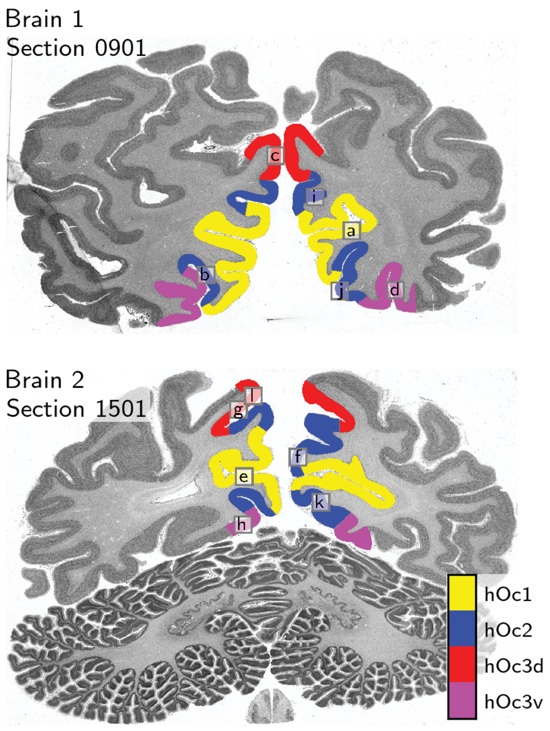
1.1 Problem statement and challenges

The aim of this thesis is to automatically parcellate high-resolution histological brain sections into cytoarchitectonic areas using deep learning methods. Cytoarchitectonic areas are defined by the laminar distribution of neurons throughout the cortex, the columnar structure of the cortex, and the presence or absence of cell clusters (Amunts and Zilles, 2015). They will also be called *(cortical) brain areas* in the following. Figure 1.1 shows the structure of different cortical areas. Section 2.1 gives a more detailed description of histological sections and definition of a cytoarchitectonic brain area. There are several challenges associated with automatic recognition of cytoarchitectonic areas (see Figure 1.1b for examples):

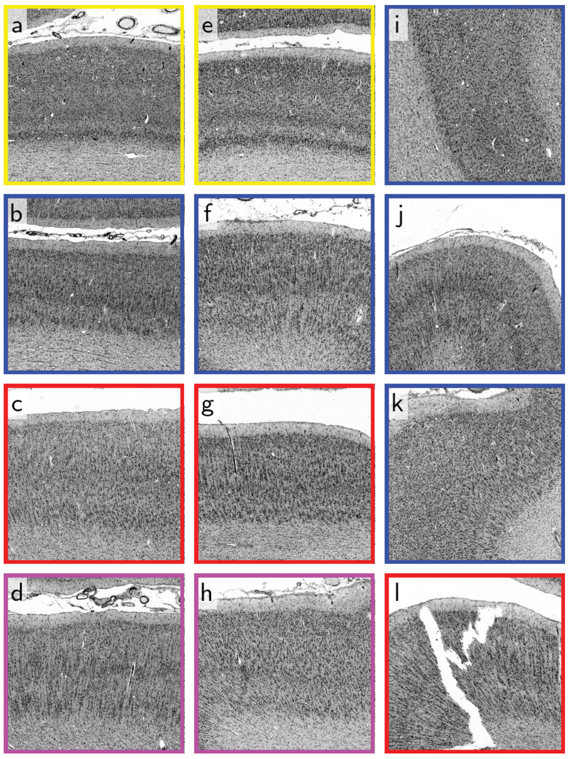
- **Localization of brain areas**. Brain areas are distinguished by subtle differences in the lamination pattern of the cortex. The human brain is highly variable between individuals with the main source of differences being the cortical folding pattern (Mangin, Riviere, et al., 2004). Some brain areas are always located at the same position relative to an anatomical landmark like a specific fold. For example, the primary visual cortex is located in the calcarine sulcus and the separation between motor and somatosensory cortex is in the central sulcus (Amunts and Zilles, 2015). However, in general, brain areas are not correlated to anatomical landmarks and vary in size between different individuals (Amunts, Lepage, et al., 2013; Fischl, Rajendran, et al., 2007; Zilles, Schleicher, et al., 1997). Automatic methods should thus not focus on the folding of the cortex but on the cell densities inside the cortex to recognize brain areas.

- **Fixed topology of brain areas**. Brain areas, unlike objects in natural images, follow a topology. In every brain, each brain area always has the same brain areas as neighbors. Although the localization of the brain area relative to anatomical landmarks and its relative size varies between individuals, its neighbors are fixed. Automatic methods should produce results that respect this topology.

- **Oblique cuts, curvature, and artifacts**. The organization of the cortex in laminae parallel to the surface of the brain is best visible when the sectioning plane is perpendicular to the surface of the brain. Due to the highly folded surface, every section contains parts of obliquely (non-perpendicularly) sectioned cortex which does not show a clear lamination pattern. The sectioning with the microtome additionally introduces sectioning artifacts (Pichat et al., 2018). Furthermore, the cortical folds (i.e., sulci and gyri) distort the relative thicknesses of the cortical laminae (Waehnert et al., 2014). Last, staining inhomogeneities can occur on the same section or between sections. Owing to these



(a) Cell-body stained histological sections from two different brains showing areas `hOc1`, `hOc2`, `hOc3d`, and `hOc3v` of the visual system.



(b) Details showing cytoarchitectonic properties of different areas. They are rotated such that the pial surface faces upwards. The color of the border corresponds to the area that the detail was extracted from and the precise location is marked on the histological section (Figure 1.1a) by a letter.

Figure 1.1: Brain area recognition is a challenging task. Brain areas, e.g., areas `hOc1` (a, e), `hOc2` (b, f), `hOc3d` (c, g) and `hOc3v` (d, h), have different lamination patterns. For a layperson differentiating between brain areas is difficult (e.g., (b) and (g) look similar, but are different brain areas). The cortical laminae are obscured by curvature (j), oblique sectioning (i, k) and sometimes also sectioning artifacts (l). Areas follow a certain topology; e.g., `hOc1` is surrounded by `hOc2` in both brain 1 and brain 2. However, the large inter-individual variability between brains results in different local sulci and gyri in the two sections.

effects, the lamination pattern is obscured in many parts of the data, resulting in additional challenges for automatic methods.

- **Limited labeled training data**. Brain area parcellation is a challenging task that requires neuroanatomists. Naturally, not much labeled training data is available, and it is expensive to generate. Thus, training deep learning models, which usually require lots of labeled training data, is challenging.

- **Large input sizes**. Recognizing cytoarchitectonic patterns requires on the one hand high-resolution images to recognize the shape and size of neurons, and on the other hand a large field of view to recognize relevant cell clusters and consistent patterns. These requirements result in large model architectures and a lot of data that needs to be processed for each input thus leading to computational challenges.

This thesis introduces deep learning based methods for automatic brain segmentation that successfully deal with such difficulties. Making headway in automated brain area parcellation has a direct impact on neuroscience. First, the models introduced in this thesis have the potential to support and speed up the manual brain area mapping process, enabling the delineation of new areas and brain volumes in significantly less time. Second, an automatic model for analysis of high-resolution histological sections can be used to extract compact feature representations of the high-resolution input. These features can be automatically calculated at every location in the cortex and correlated with other signals that are available in the same space (e.g., connectivity data, local curvature, cortical thickness, etc.). This is not possible when manually extracting features and allows to validate hypotheses about brain structure and gives insights into which features are objectively useful for cortical analyzes. High-resolution brain area segmentation provides unique challenges for the application of machine learning methods as seen above. This thesis explores several ways for dealing with limited training data that may be of use for other applications in medical imaging settings.

1.2 Objectives

In order to show the applicability of deep learning methods for the task of brain area parcellation, several approaches are explored in this thesis. These approaches range from fully-supervised segmentation of brain areas in several brain volumes over propagation of one area label to spatially close sections to self-supervised feature learning from sections of one brain. In detail, the following objectives are defined:

- **Investigate the feasibility of using cortical profiles for automatic segmentation.** The currently used observer-independent brain area border confirmation method (Schleicher, Amunts, et al., 1999) uses gray-level index (GLI) profiles that are extracted from the high-resolution images. To evaluate the usefulness of these features in a machine learning setting, a 1D CNN will be trained for the classification of profiles in different brain areas.

- **Develop a model leveraging high-resolution texture input and prior knowledge.** The main advantage of deep learning models is their ability to learn relevant feature representations from raw data. Thus, an end-to-end model for brain area segmentation will be developed and compared to the previous profile feature model to see if the features learned by the CNN are different to the features encoded in the profiles. Brain areas follow a certain topology. Thus, the effect of combining high-resolution texture information with prior information about the topology of areas in the CNN model will be investigated.

- **Build a workflow to propagate labels from one section to spatially close sections to support the manual mapping.** In the current manual mapping process, every 60th brain section is analyzed. A profitable application for automatic brain area segmentation methods is to fill the gaps between two annotated sections without requiring a registration between the sections. The effect of limiting the brain area segmentation task to the label propagation task will be analyzed.

- **Design a self-supervised learning task to leverage unlabeled sections.** Due to the time-consuming manual brain area labeling process, training data for supervised brain area segmentation is limited. However, a large amount of unlabeled sections is available. A self-supervised learning task can be used to fit models on the unlabeled sections, allowing them to learn features from the cortex in an unsupervised manner. This auxiliary task can be used to pre-train supervised brain area segmentation models.

1.3 Outline

This thesis describes and compares different neural network models. The Glossary on page 145 provides a central list of all neural network architectures (e.g., `NeuralNetworkArchitecture`) and their trained instances (the models, e.g., `nn-model`). Additionally, it describes the sets of histological sections (collections) that were used to sample training datasets, and the training datasets $\mathcal{X}$ themselves. Finally, the Glossary lists the abbreviations and mathematical notation introduced in this thesis.

Chapter 2 introduces the necessary neuroscientific and deep learning background. First, an overview over the human brain is given, and the current approach for brain area delineation is introduced. Second, relevant deep learning principles and neural network architectures are introduced. Third, the datasets that are used in this thesis are presented. Chapter 3 discusses related work on brain area parcellation, analysis of histological sections, and training neural networks with limited training data. Chapters 4 to 7 introduce and discuss four different approaches to automatic analysis of brain areas on high-resolution histological sections. Figure 1.2 provides a visual summary of all CNN architectures, their respective inputs and outputs, and the different training tasks that were developed for these approaches.

In Chapter 4 the *profile classification task* is examined. For this task, profile features are extracted from high-resolution histological brain sections and a 1D CNN (`ProfileNetwork`) is trained to classify the inputs into different cortical areas. The advantages and drawbacks of this model are discussed and the input features are analyzed with respect to their predictiveness for different brain areas. The *patch segmentation task* is covered by Chapter 5 with an end-to-end CNN model (`BaseNetwork` and `AtlasAwareNetwork`) which combines high-resolution texture inputs with prior knowledge to produce high-resolution segmentations of the input. This model is compared with the 1D CNN from Chapter 4. The influence of the prior knowledge and the learned feature maps are analyzed.

Chapter 6 introduces the self-supervised *distance and location regression task* which leverages the 3D relationship between two sections of the same brain. This allows to exploit unlabeled sections for brain area segmentation by pre-training `SiameseNetwork` models on this auxiliary task, thus partly eliminating the disadvantages of limited labeled training data. The self-supervised task allows to learn compact feature representations from the high-resolution input which are analyzed in this chapter. Chapter 7 limits the general patch segmentation task to the *deep label propagation task* which has the aim of propagating one area label to spatially close sections in the same brain volume. For this task, a `MultiScaleNetwork` model combining high-resolution texture information with low-resolution texture information providing additional context is developed. With these models, a label propagation workflow is designed to densely predict one brain area given sparse manual annotations. These precise predictions can then be used to provide high-resolution area reconstructions for the multilevel human brain atlas which is being developed in the context of the Human Brain Project (HBP) (*HBP Human Brain Atlas* 2018). Finally, Chapter 8 summarizes the main results and conclusions and highlights venues of future work arising from the presented findings.

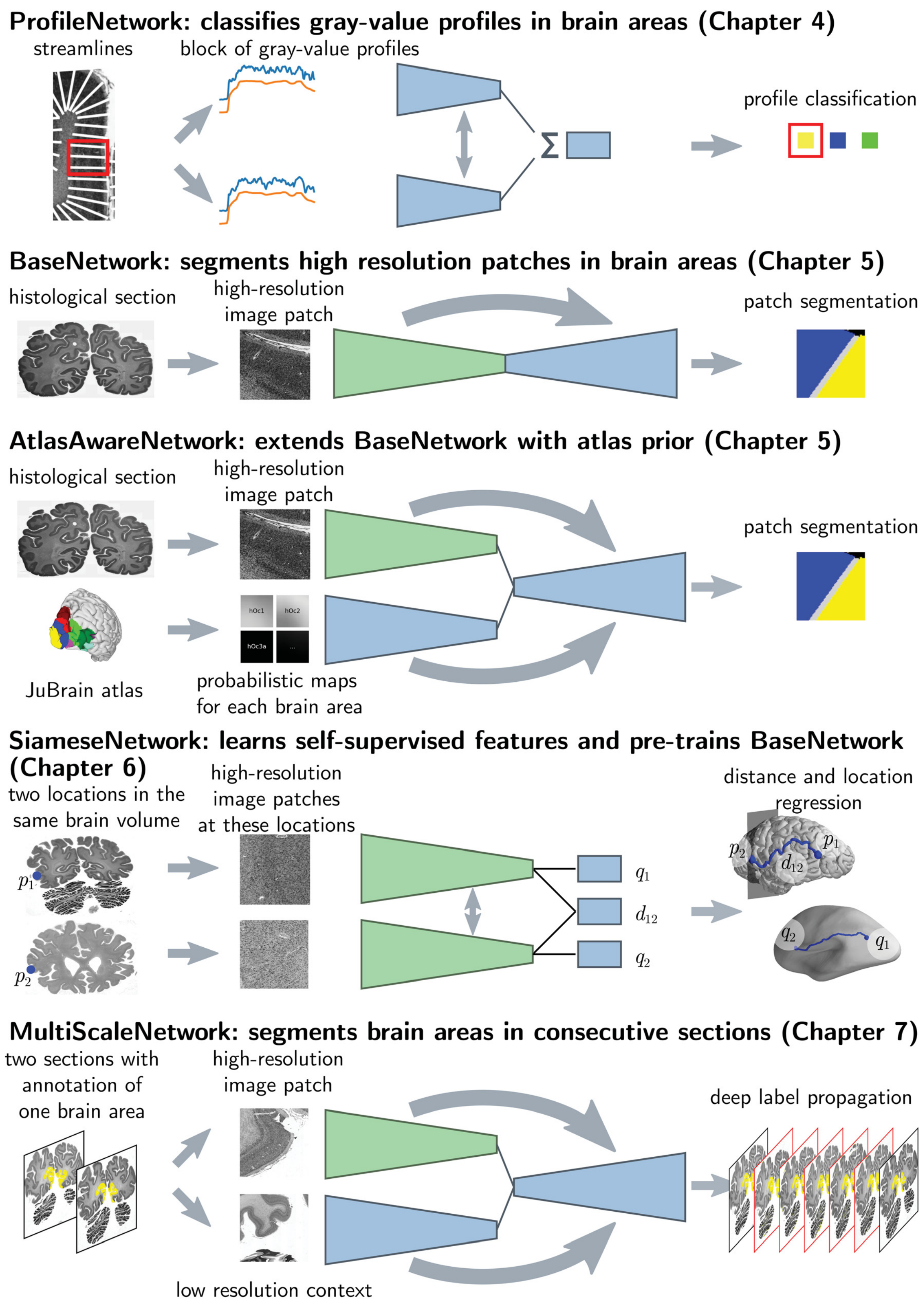

ProfileNetwork: classifies gray-value profiles in brain areas (Chapter 4)
streamlines
block of gray-value profiles
Σ
profile classification
BaseNetwork: segments high resolution patches in brain areas (Chapter 5)
histological section
high-resolution image patch
patch segmentation
AtlasAwareNetwork: extends BaseNetwork with atlas prior (Chapter 5)
histological section
high-resolution image patch
patch segmentation
hOc1
hOc2
hOc3a
...
JuBrain atlas
probabilistic maps for each brain area
SiameseNetwork: learns self-supervised features and pre-trains BaseNetwork (Chapter 6)
two locations in the same brain volume
high-resolution image patches at these locations
distance and location regression
p_1
p_2
q_1
d_{12}
q_2
MultiScaleNetwork: segments brain areas in consecutive sections (Chapter 7)
two sections with annotation of one brain area
high-resolution image patch
deep label propagation
low resolution context

Figure 1.2: Visualization of neural network architectures introduced in this thesis. Several networks partly share the same architecture. These parts are marked in green.

2 Background

This thesis combines understanding the structure of the human brain with deep learning research. This chapter briefly introduces both of these research areas, focusing on aspects that are relevant for this thesis. For a more complete overview of the human brain and its function, see Zilles and Amunts (2012). Goodfellow et al. (2016) provide a current and in-depth overview over deep learning techniques. Since artificial neural networks were initially inspired by biological neural networks, there is some shared terminology between deep learning and neuroanatomy. In this thesis, the term *neuron* will be exclusively used for biological neurons, and the equivalent structures in a deep neural network will be referred to as *nodes*. The cortical laminae will always be called *laminae*; the term *layer* will be exclusively used for a layer in an artificial neural network. In addition, the term *neural network* will always refer to an *artificial neural network*, not a biological neural network.

In the following, Section 2.1 gives an overview over the structure of the human brain, different imaging modalities, and principles of brain mapping. Then, Section 2.2 introduces convolutional neural networks (CNNs) and semantic segmentation architectures, and how to train and evaluate them. Finally, Section 2.3 covers the datasets collected from histological brain sections that will be used in the following chapters to train models for brain area segmentation.

2.1 The human brain

The brain is our most complex organ, consisting of 86×10^9 neurons (Azevedo et al., 2009), each one connected to up to a thousand other neurons via synapses. It controls most functions of our body and is dedicated to perception, the processing of the stimuli, and the formulation of appropriate responses. In the following, standard anatomical nomenclature is used. Figure 2.1 provides a coordinate system with the most important terms.

The human brain consists of the cerebrum, the cerebellum, and the brainstem. Out of the three, the cerebrum constitutes the biggest part of the central nervous system. The cerebrum can be separated into two hemispheres that are roughly mirror symmetric, although several hemispheric differences exist. Each hemisphere can be further subdivided into the parietal, temporal, occipital, and frontal lobes

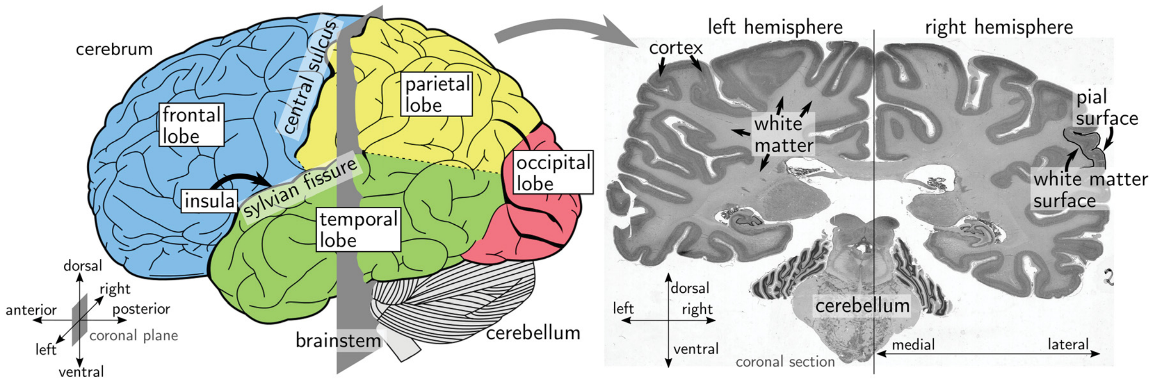


Figure 2.1: Overview of the human brain.[1] Left: Schematic drawing of the brain with the four lobes of the cerebrum. The insula is hidden inside the sylvian fissure. Right: Coronal cell-body stained histological section of the brain through parietal and temporal lobes and the cerebellum showing cortex and white matter.

and the insula based on gross anatomical landmarks (see Figure 2.1). The outer surface of the brain contains a 2 mm to 4 mm thin band (Zilles and Amunts, 2012) of interconnected neurons called the *cortex* or *gray matter*. The inside of the brain is comprised of myelinated fiber tracts and glia cells (*white matter*) and subcortical regions containing neurons (*subcortical nuclei*). Processing of information takes place in the cortex and in the subcortical nuclei, while the fiber tracts connect cortical and subcortical regions and transport sensory signals to the brain and motor signals to the muscles.

The surface of the brain is highly folded (*gyrification*). Outward folds are called *gyri* and inward folds *sulci*. Two-thirds of the cortex is buried in the sulci (Zilles and Amunts, 2012). The insular lobe, for example, is not visible from the outside, but buried deep in the lateral fissure which separates the temporal lobe from the parietal and frontal lobes. There is a set of primary gyri and sulci that are present in all humans. However, the brain is highly variable, and folding patterns and sizes of different brain regions vary between different individuals (Amunts, Schleicher, et al., 2002; Zilles and Palomero-Gallagher, 2015). The outer surface of the brain is called *pial surface*. The border between gray matter and white matter is called *white matter surface*.

Cytoarchitecture describes the cellular composition of the cortex and its variation throughout the cortex. Using cytoarchitecture, the cortex can be divided in *isocortex* and *allocortex*. The isocortex covers nearly 96% of the cortical surface and is characterized by six laminae of different cellular composition. In contrast, the allocortex has a more variable laminar pattern (Zilles and Amunts, 2012). Based on differences in the cellular composition of the cortical laminae, different *cytoarchitectonic*

[1]Schematic brain image retrieved from https://medicalxpress.com, under CC0 license.

areas (also termed *(structural) brain areas* in the following) can be distinguished in the cortex (e.g., Amunts, Schleicher, et al., 2007; Amunts and Zilles, 2015; Schleicher, Palomero-Gallagher, et al., 2005). These brain areas are the focus of the following sections. Section 2.1.1 describes the imaging modality that is needed to identify different brain areas. Then Section 2.1.2, defines cytoarchitectonic brain areas, shows examples, and discusses the relationship between structure and function. The currently used observer-independent method for brain area identification is outlined in Section 2.1.3. This is followed by a discussion of image registration techniques (Section 2.1.4) that are used amongst others to create the JuBrain atlas containing probabilistic maps of cytoarchitectonic areas in a common reference space (Section 2.1.5).

2.1.1 Imaging modalities

The brain can be observed at different scales and with different objectives. We differentiate between in-vivo and post-mortem imaging modalities. Whole brain volumes can be imaged in-vivo at a resolution of several millimeters up to 250 µm per voxel using magnetic resonance imaging (MRI). MRI uses the magnetic properties of hydrogen to get contrast between different parts of the brain tissue. A structural MRI allows to distinguish between cortex and white matter. This modality is often used to detect tumors or other lesions in the brain or to research structural changes that the brain undergoes during aging. Functional magnetic resonance imaging (fMRI) measures the blood oxygen level in different regions. Since brain regions need more oxygen when activated, this is an indirect measure of functional activation. Thus, fMRI can be used to identify and characterize functional brain networks.

With high-field MRI, the laminar structure of the cortex could be visualized in several cortical regions (Duyn, 2012). However, moving beyond this resolution, and visualizing single neurons and their connectivity is not possible with current MRI techniques. To enable a high-resolution view of the cytoarchitecture, i.e., the morphology of cells and their distribution in the brain, histological sections have to be acquired. For this modality, post-mortem brains are embedded in paraffin or frozen (cryosectioning) and sectioned into 20 µm thin sections using a microtome. The resulting tissue sections are put on a glass slide, and stained for cell-bodies using silver staining (Merker, 1983). See Amunts, Malikovic, et al. (2000) for a detailed description of the tissue preparation and staining protocol. The resulting sections can then be digitized and subsequently analyzed at 1 µm resolution using a light microscope. A post-mortem histological human brain section can have a size of up to 8×10 cm. At 1 µm resolution this results in sizes of up to $80\,000 \times 100\,000$ pixels and file sizes of 8 GB at 8 bit precision for one digital scan. This resolution allows us to distinguish the individual neurons in the cortex. Figure 2.2 shows an entire histological section

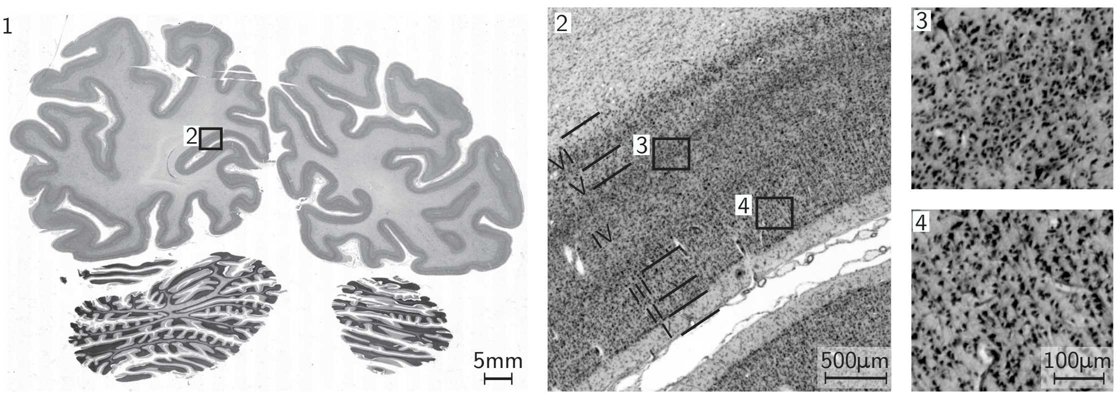


Figure 2.2: Cell-body stained histological section of the human brain at different magnifications. 1. Whole brain section. 2. Detail showing cortex and the cortical laminae (roman numerals I-VI). 3. Extract of granular lamina IV showing small cells. 4. Extract of lamina III showing pyramidal cells with the characteristic triangular shape.

and extracts of the cortex at different magnifications.

Although the individual images are only two dimensional, they can be registered together to reconstruct a 3D volume of the original brain (see Section 2.1.4 for details regarding registration). The BigBrain is a prominent example of such a reconstructed 3D volume (Amunts, Lepage, et al., 2013). It is the first brain volume that was reconstructed from over 7400 cell-body stained histological sections at an isotropic resolution of 20 µm per voxel. It is part of the Human Brain Project's (HBP) multilevel human brain atlas (*HBP Human Brain Atlas* 2018).[2]

Histological sections have the disadvantage that during the sectioning the tissue will be inevitably deformed and artifacts such as rips and tears will be introduced. Although it is possible to reconstruct a 3D volume from the individual sections (e.g., Amunts, Lepage, et al., 2013), this is a time-consuming process. Recently, Magnain et al. (2014) showed that blocks of tissue that were imaged with optical coherence tomography (OCT) contained similar information to Nissl stained histological sections[3]. OCT is an imaging technique that measures the backscattered light of the tissue sample which changes depending on the refraction index of the tissue. This allows a non-invasive analysis of blocks with a width of several centimeters. In the future, this technique may provide an alternative to the gold-standard cell-body stained histological sections.

[2]The BigBrain can be explored using an interactive web viewer at `http://bigbrain.humanbrainproject.org`, and is available for download at `http://bigbrain.loris.ca`.
[3]Similar to silver staining, Nissl staining is used to visualize neuronal cell bodies.

Table 2.1: Characteristics of the six isocortical laminae (Trepel, 2004, p. 219)

Lamina	Characteristics
I (molecular layer)	no pyramidal cells, low density
II (external granular layer)	granular, small pyramidal cells, high density
III (external pyramidal layer)	large pyramidal cells, average density
IV (internal granular layer)	granular, small pyramidal cells and other cells, high density
V (internal pyramidal layer)	very large pyramidal cells, average density
VI (multiform layer)	small pyramidal cells and other cells, moderate density

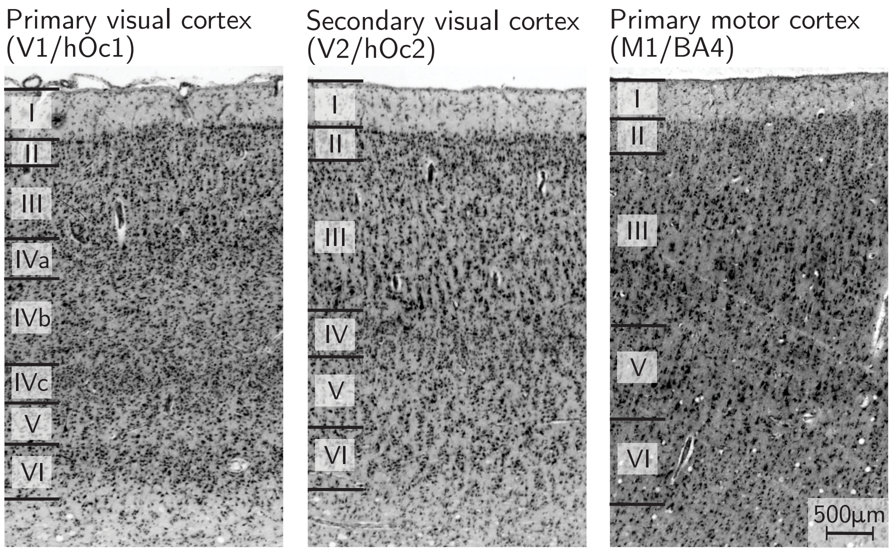

Primary visual cortex (V1/hOc1)
I
II
III
IVa
IVb
IVc
V
VI
Secondary visual cortex (V2/hOc2)
I
II
III
IV
V
VI
Primary motor cortex (M1/BA4)
I
II
III
V
VI
500µm

Figure 2.3: Comparison of cortex from primary and secondary visual areas and primary motor cortex. The size and composition of the laminae is different for each area.

2.1.2 Cytoarchitecture

Cytoarchitecture describes the distribution, type, and size of neuronal cell bodies in the cortex. It is studied using the cell-body stained histological sections described in Section 2.1.1. The isocortex consists of six laminae that contain different cell types and densities, columnar structures and repetitive, modular-like structures (Amunts and Zilles, 2015). The six laminae of the isocortex are commonly denoted by roman numerals starting at the pial surface going inward to the gray/white matter border. Their different characteristics are summarized in Table 2.1. Figure 2.2 shows two extracts of cortex from different laminae (Subfigures 3 and 4). Their different composition is clearly visible.

Cytoarchitectonic areas The laminar structure and the cellular composition of the laminae of the isocortex are different at different locations. *Cytoarchitectonic areas* (also termed *(cortical) brain areas* in the following) with distinct laminar characteristics and thicknesses can be identified. Between different cytoarchitectonic areas, the characteristics and thicknesses of the cortical laminae varies (see Figure 2.3). For example, the primary visual cortex (V1) contains a very thick lamina IV that can even be subdivided in several sub-laminae. It can be clearly separated from the secondary visual cortex (V2) which has a thinner lamina IV without sub-layering. In contrast, the motor and pre-motor cortex do not contain a visible lamina IV and have a fuzzy border between lamina VI and white matter.

In the beginning of the 20th century several researchers studied differences in cytoarchitecture and published maps of the human cerebral cortex (e.g., Brodmann, 1909; von Economo and Koskinas, 1925; C. Vogt and 0. Vogt, 1919). These maps influence neuroscientific research even today. However, these maps were defined based on a small sample of brains and with highly objective criteria, limiting the reproducibility and generalizability of the maps (Zilles and Amunts, 2010). Modern brain mapping considers brain areas as structurally and functionally well-defined entities (Amunts and Zilles, 2015). An observer-independent method was developed to accurately place a border between two cytoarchitectonic areas based on statistical criteria (Schleicher, Amunts, et al., 1999; Schleicher, Palomero-Gallagher, et al., 2005). This method is described in detail in Section 2.1.3.

Structure and function The structurally defined brain areas can be correlated to functionally defined areas (e.g., Amunts and Zilles, 2015; Eickhoff, Paus, et al., 2007; C. Vogt and 0. Vogt, 1919). Based on this insight, brain areas can be divided in primary, secondary, tertiary, and higher associative brain areas (Trepel, 2004, p. 218). The primary sensory brain areas receive sensory information (e.g., visual, auditory, somatosensory, and gustatory). For example, the primary visual cortex is located in the occipital lobe in the calcarine sulcus and the primary somatosensory area is located on the posterior wall of the central sulcus. The primary sensory areas are surrounded by secondary and tertiary areas that receive information from the primary sensory areas. In contrast to the sensory areas, the primary motor area has direct connections to the spinal cord and generates voluntary movements. It is located opposite the primary somatosensory area on the anterior wall of the central sulcus. Besides these brain areas that are (mostly) involved in the processing of specific sensory or motor information, higher associative areas exist (e.g., in the frontal cortex). These areas integrate signals from various sensory inputs, and are involved with motion planning, emotion, and memory. Primary sensory areas have a distinctive laminar pattern, whereas higher associative areas have a less clear lamination. This makes the primary areas easy to recognize and well-suited as a first test case for automatic brain area recognition methods.

Structural gradients Besides structural division in distinct cortical areas, several large-scale gradients can be found in the cortex. Recently Huntenburg et al. (2017) have proposed that areas are arranged on a sensorimotor (sensory and motor areas) to transmodal (higher associative areas) gradient. As already mentioned, the degree of laminar differentiation is related to the position of the area within a sensory structural hierarchy (Barbas, 1986). Similarly, Collins et al. (2010) found primary sensory regions to have higher neuronal density than other cortical areas. MRI studies have found changes of intracortical myelin (Glasser et al., 2016) and cortical thickness (Wagstyl, Ronan, et al., 2015) along these gradients.

2.1.3 Observer-independent method for brain area mapping

The observer-independent method of Schleicher, Amunts, et al. (1999) allows to define the position of an areal border based on statistical criteria. Cytoarchitectonic areas are distinguished by their laminar structure and the cellular composition of the laminae (Section 2.1.2). This laminar structure of the cortex can be captured by the gray-value intensities (*profiles*) of traverses of the cortex from the pial surface to the gray/white matter border (*streamlines*). For the calculation of the streamlines precise delineations of the inner and outer boundary are provided manually. The streamlines are calculated by following the gradient of a potential field between the inner and outer cortical boundary calculated with Laplace's equation (Jones et al., 2000).

The profiles are extracted by sampling the gray-level index (GLI) image along the cortical traverses. The GLI is calculated by segmenting the cells on a 1 µm histological image and then calculating the volume fraction of cell bodies in windows of 16×16 pixel resulting in 16 µm resolution images. A higher GLI signifies a larger fraction of cell bodies. Thus, the shape of the profile quantifies the changes in the volume faction of cell bodies from the pial surface to the gray/white matter border and provides valuable information about cytoarchitecture.

With the observer-independent method, cytoarchitectonic borders are identified by significant changes between the shapes of neighboring blocks of profiles. For each profile, 10 features are calculated based on the central moments and their derivatives. Then, two blocks of neighboring profiles are compared by calculating the Mahanalobis distance (Mahalanobis, 1936) between the mean feature vectors of each block. The Mahanalobis distance is a generalized Euclidean distance that takes into account the covariance of the features. Those locations that have significant differences in Mahanalobis distances over several block sizes are accepted as borders between cortical areas. The significance was inferentially tested with a T-test with Bonferroni correc-

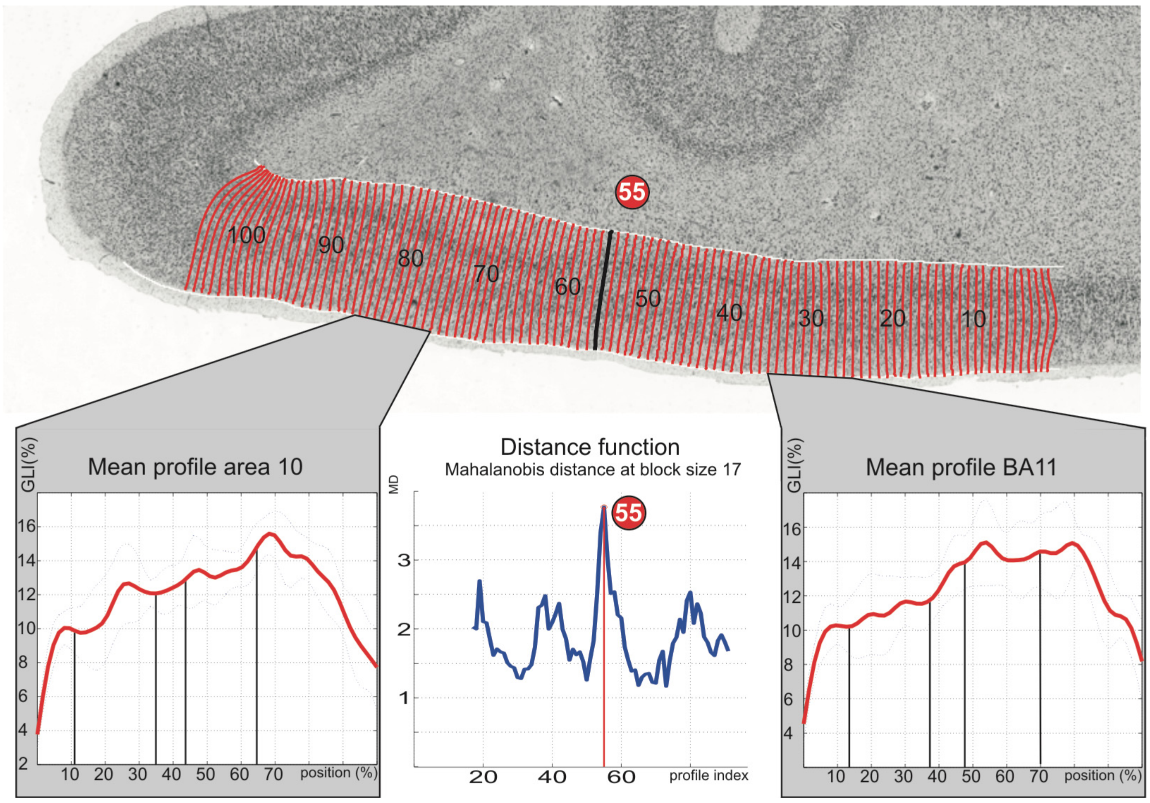


Figure 2.4: Application of the observer independent method of Schleicher, Amunts, et al. (1999).[4] Cortical traverses (streamlines) are colored in red on the cortex. Length-normalized mean GLI profiles from neighboring cytoarchitectonic areas are shown on the bottom right and bottom left. The borders between the cortical laminae are visible by black lines. The Mahanalobis distance between neighboring blocks of profiles is plotted for different locations on the cortex at the bottom middle. The distance is maximal at position 55, indicating a border between two cytoarchitectonic areas.

tion. Figure 2.4 visualizes mean profiles from two different areas and their resulting Mahanalobis distance. The advantage of this method is that it allows for an objective definition of a border. In Chapter 4 we will build upon this method to train a model that automatically classifies a given profile in different cytoarchitectonic areas.

2.1.4 Image registration

Image registration is the process of transforming and deforming a source image S in such a way that it looks like a target image T. The source and target images can be 2D sections or 3D volumes. This enables for example the alignment of adjacent histological sections or of two MRI volumes from different subjects. Zitova and Flusser

[4]Figure reproduced from Bludau (2011) by permission of Sebastian Bludau (INM-1, FZJ).

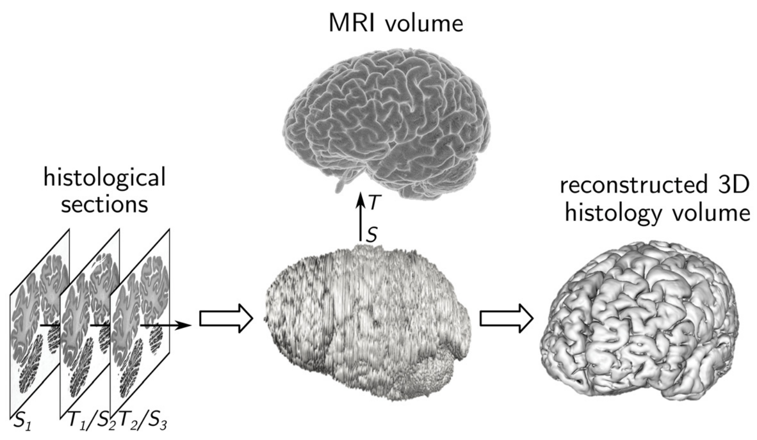


Figure 2.5: 3D histology reconstruction. Histological sections are registered section-by-section. The resulting volume is registered to a reference MRI volume. S is the source image and T is the target image for a given registration.

(2003) provide a comprehensive general overview over image registration methods and Pichat et al. (2018) summarize approaches relevant to histology reconstruction. The objective of the registration is to minimize an energy function $E(S, T)$ which usually consists of two terms. The first one is the matching criterion, i.e., a measure of similarity between source S and target T, and the second one is a regularization term which constrains the space of possible transformations of source S (Pichat et al., 2018).

Broadly, registration algorithms can be categorized in two categories based on the type of transformations: landmark-based and intensity-based registration. For landmark-based registration first points of interest are extracted from both S and T. Then, correspondences between the landmarks are found and a transformation is calculated. Thus, landmark-based registration is based only on a subset of features from the images, and is therefore fast. Intensity-based registration, in contrast, uses the voxel intensities of the images to estimate the similarity between S and T and is thus slower but more accurate than landmark-based methods. The registration procedures mentioned in this thesis (e.g., the creation of the BigBrain and the JuBrain atlas) use the more accurate intensity-based registration.

Image registration is used to reconstruct a 3D volume from the individual histological sections which are deformed during the cutting of the brain. This is usually achieved by registering adjacent pairs of sections to each other. With only this section-to-section registration, it is not guaranteed that the resulting volume resembles the original brain volume as curves tend to be straightened out by this process. Therefore, in a second step, the registered sections are usually registered to a reference brain volume which can for example be an MRI that was taken prior to slicing the brain. Figure 2.5 schematically shows this process. Once a 3D histological volume exists, this can also be registered to other brain volumes from other subjects. This is done in Section 2.1.5 for the creation of the probabilistic maps in the JuBrain atlas.

(a) Area `hOc1` (V1) **(b)** Area `hOc2` (V2)

Figure 2.6: Example probabilistic maps from the JuBrain atlas.[5] Colors range from red (high probability) to blue (low probability).

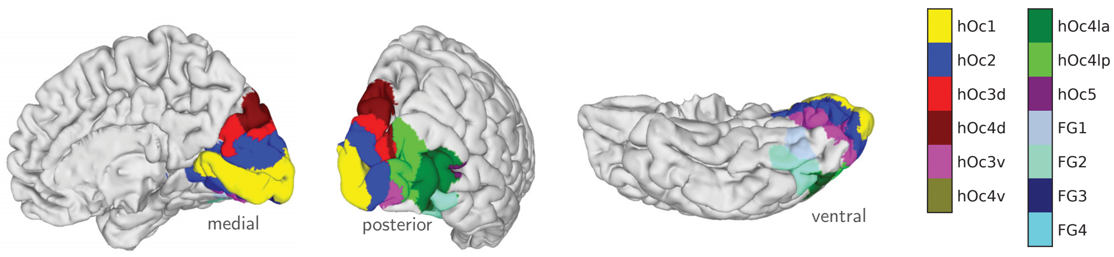
2.1.5 JuBrain: Probabilistic atlas of brain areas

One of the ongoing efforts at the Institute of Neuroscience and Medicine (INM-1) is the assembly of the *JuBrain* atlas, a novel probabilistic atlas of cytoarchitectonic areas (Amunts, Schleicher, et al., 2007; Amunts and Zilles, 2015; Zilles and Amunts, 2010). For each area contained in this atlas, annotations of the brain area were made on 10 different brains using the observer-independent method of Schleicher, Amunts, et al. (1999). These annotations are registered to a common reference space, the Colin 27 space (Holmes et al., 1998), and averaged to form probabilistic maps that contain for each voxel and brain area the probability that this brain area is present. This atlas can be used to allocate the results of functional studies to specific structural brain areas (Eickhoff, Paus, et al., 2007) and it can be used to interpret the results of connectivity studies (S. Caspers et al., 2013). Thus, the JuBrain atlas is an important tool for establishing further structure/function relationships (Eickhoff, Heim, et al., 2006). Figure 2.6 shows probabilistic areas of the JuBrain atlas in the common reference space.

Structural areas of the visual system The JuBrain atlas contains annotations of several areas of the visual system. These areas are shortly listed here, as they will be used as labels for the supervised classification and segmentation models in Chapters 4, 5 and 7. The visual system is located in the occipital lobe.

Visual inputs from the retina, which have been preprocessed in subcortical areas, reach the cerebral cortex at the calcarine sulcus. This region is known as the primary visual cortex, or V1. Its structural equivalent is area `hOc1` (Amunts, Malikovic, et al., 2000) which can be easily identified by the stria of Gennari, a light band in lamina IV which is formed by the optic radiation from the thalamus (see Figure 2.3). Surrounding the primary visual cortex is the secondary visual cortex, V2, which

[5]Images taken from the JuBrain Cytoarchitectonic Atlas Viewer (*JuBrain Atlas Viewer* 2014–2019).



(a) Visual areas in the JuBrain atlas **(b)** Colormap

Figure 2.7: Areas of the visual system in the JuBrain atlas shown as a maximum probability map. This visualization allows to show several (potentially overlapping) probabilistic maps at once by showing at each voxel the area with the highest probability. Not shown are areas `hOc4v`, `FG3`, and `FG4`.

starts to process the visual stimuli (Trepel, 2004, p. 236). The structural equivalent to this area is area `hOc2` (Amunts, Malikovic, et al., 2000). Dorsally from `hOc2` structural areas `hOc3d`, `hOc4d` (Kujovic et al., 2013) were defined. Ventrally from `hOc2` structural areas `hOc3v` and `hOc4v` (Rottschy et al., 2007), `FG1`, `FG2` (J. Caspers et al., 2013), `FG3`, and `FG4` (Lorenz et al., 2015) were found. At the lateral side of the occipital lobe, structural areas `hOc4lp`, `hOc4la` (Malikovic, Amunts, Schleicher, Mohlberg, Kujovic, et al., 2016), and `hOc5` (Malikovic, Amunts, Schleicher, Mohlberg, Eickhoff, et al., 2006) were delineated.

Figure 2.7 shows these areas on the JuBrain atlas. The colormap for this figure, defined in Figure 2.7b, will be used in the remainder of the thesis for coloring structural areas of the visual system. From these areas, the primary visual area `hOc1` is easiest to identify. As mentioned in Section 2.1.2, it can be used as a proof-of-concept test. To evaluate the capabilities of automatic segmentation models to recognize arbitrary brain areas, the performance on other areas than `hOc1` needs to be considered.

2.2 Deep learning for image analysis

Deep neural networks have a long history, starting from simple one layer perceptrons (Rosenblatt, 1961) and evolving to fully connected multi-layer perceptrons (Hinton, 1986). The first hierarchical convolutional network architecture, the Neocognitron, was developed by Fukushima and Miyake (1982). After initial periods of active research in these architectures, they were mostly ignored until the 2000s due to computational problems and learning instabilities. The renaissance of CNNs and their wide-spread adaption in the computer vision and machine learning community was due to a spectacular success by Krizhevsky et al. (2012) on the ImageNet competition (Deng et al., 2009). They trained a deep CNN on a 1000 class classification task using the 14 million natural images in the ImageNet database and were able to beat competing methods by a large margin. The success of this model was in large parts due to three factors (LeCun et al., 2015): 1. The efficient use of GPUs for training the network, 2. the availability of huge amounts of data to train the model, and 3. new training and regularization techniques (Nair and Hinton, 2010; Srivastava et al., 2014). This success revolutionized the field of computer vision and machine learning. Today, deep neural networks dominate in almost all recognition, detection, and segmentation tasks. Notable successes of the last years include the better-than-human classification of skin cancer (Esteva et al., 2017), AlphaGo which beat the human Go champion (Silver et al., 2016), and advances in neural machine translation (Johnson et al., 2017).

The power of deep neural networks stems from their ability to learn hierarchical feature representations from data. Unlike classical machine learning, which learns classifiers based on engineered features, deep learning allows to learn features directly from the raw input data, and to derive higher level features from lower level features. Although a multi-layer perceptron with one hidden layer can theoretically represent any continuous function (Universal Approximation Theorem; Hornik, 1991), it is not guaranteed that such a function will be found in practice. Deep models with many layers have the advantage that they require fewer parameters, are well-regularized, and have better generalization properties (Bengio, 2009).

Section 2.2.1 presents common building blocks of CNNs. Following this, Section 2.2.2 introduces typical CNN architectures, while Section 2.2.3 focuses on segmentation architectures. Then, Section 2.2.4 details the training procedure for artificial neural networks, Section 2.2.5 introduces evaluation metrics, and Section 2.2.6 explains the concept of transfer learning. In the following, lowercase characters denote scalars or vectors, and uppercase characters denote tensors with rank two or higher (e.g., matrices, volumes).

2.2.1 Building blocks of artificial neural networks

A neural network is, in very general terms, a non-linear function $f : \mathcal{D}_0 \to \mathcal{D}_1; f(X;\theta)$ which is parametrized by θ. In the case of image processing, the input domain $\mathcal{D}_0$ is typically three-dimensional with $\mathcal{D}_0 = \mathbb{R}^{h \times w \times c}$, $h, w, c \in \mathbb{N}$ for images with height h, width w, and c channels (e.g., $c = 3$ for RGB images). The semantic segmentation network introduced in Chapter 5 produces segmentations of the input image $X \in \mathcal{D}_0$ in k classes, with the output segmentation $Z = f(X;\theta) \in \mathcal{D}_1 = \mathbb{R}^{h' \times w' \times k}$, with $h' \leq h$ and $w' \leq w$ due to the loss of border pixels during *valid* convolutions (see Equation (2.4)). The elements $Z^{(i,j,m)}$ of Z can be interpreted as the probability that the pixel $X^{(i,j)}$ at row i and column j of the input X belongs to class m. In contrast, the regression network from Chapter 6 produces a scalar value $z \in \mathcal{D}_1 = \mathbb{R}$ for an input image $X \in \mathcal{D}_0 = \mathbb{R}^{h \times w \times c}$.

Artificial neural networks can be visualized as directed weighted graphs. Typically, the nodes in a network are organized in consecutive layers $l_i(X;\theta)$ where the outputs of layer $i-1$ (*activations*) serve as inputs to layer i, resulting in a feed-forward network. A two-layer neural network is thus defined as $f(X;\theta) = l_2(l_1(X;\theta_1);\theta_2)$ with $\theta = \theta_1 \cup \theta_2$. This layer-wise definition allows the neural network to learn increasingly complex features that are calculated based on the features learned in the previous layers. Neural networks are not restricted to a strict linear sequence of layers. For example the U-Net architecture introduced in Section 2.2.3 has skip-connections between different layers, and recurrent neural networks (RNNs) have feedback connections, creating a loop over a set of layers. There are different types of layers that can be employed in a neural network. In the following the layers that area most important in the context of this work are introduced.

Fully connected layer A fully connected layer (fcl) connects every input node to every output node. Fully connected layers were used in the multilayer perceptron, an early neural network architecture (Hinton, 1986). Mathematically, a fully connected layer is defined by the function $l_{\text{fcl}} : \mathbb{R}^n \to \mathbb{R}^m$ which calculates a linear weighted combination of input values $x \in \mathbb{R}^n$ followed by an element-wise non-linear *activation function* φ:[6]

$$l_{\text{fcl}}(x;\theta) = \varphi(Ax + b)\,, \tag{2.1}$$

with $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, and $\theta = (A, b)$.

Convolutional layer In a fully connected layer, the value of each output node depends on every input node. However, when learning features on images, low-level features like lines and edges can be found using only small parts of the input image, and may occur at any position in the input. Additionally, the local relationship between pixels is relevant for recognizing structures in the image. A convolutional layer (conv) locally connects input and output nodes and uses weight sharing to learn the same local features at every position in the input and is thus well-suited for image processing. It can be described in terms of a kernel A that is convolved over the input image X and a bias term b which is added after the convolution to each output value. We call the output of a convolution *feature map*.

For inputs $X \in \mathbb{R}^{h \times w \times c}$ with size $h \times w$ and c channels, a kernel $A \in \mathbb{R}^{m \times n \times c}$ with $m \leq h$ and $n \leq w$ is used. Usually, the kernel dimensions m and n are odd. Let $p = (m-1)/2$ and $q = (n-1)/2$ be the center of kernel A. The *convolution* of input

[6]Note that here the input x is a one-dimensional vector and is thus per our notation convention denoted with a lowercase character. In the remainder of this section, the input to a layer is usually a matrix or a tensor with rank four and is thus denoted by an uppercase character X.

X with kernel A is defined by:

$$Z(i-p, j-q) = (X * A)(i, j) = \sum_{r=-p}^{p} \sum_{t=-q}^{q} \sum_{k=1}^{c} X(i+r, j+t, k) \cdot A(p-r, q-t, k)\,, \quad (2.2)$$

with the convolution operation denoted by an asterisk $*$, and $i = p, ..., h-p$, $j = q, ..., w-q$. The above formula describes a *valid* convolution. It is only defined for pixels inside the input image, thus resulting in a slightly smaller feature map $Z \in \mathbb{R}^{(h-2p)\times(w-2q)}$ due to the loss of border pixels where a valid convolution was not possible. For image segmentation this is the preferred way of doing convolutions, since padding the input image might introduce incorrect predictions at the image border. In Equation (2.2), the kernel is applied to every pixel in the input image. We can extend this definition to the more general *strided convolution* with stride $s \in \mathbb{N}$. In this general definition of a convolution, the kernel is applied to every s-th pixel in the input image:

$$Z(\frac{i-p}{s}, \frac{j-q}{s}) = (X *_s A)(i, j)\,, \quad (2.3)$$

with the convolution defined as in Equation (2.2) and $i = p + 0s, p + 1s, ..., h - p$, $j = q + 0s, q + 1s, ..., w - q$. A stride $s > 1$ has the effect that it reduces the size of the output z by factor $1/s$ which can be advantageous when processing very large images.

In Equations (2.2) and (2.3) the convolution is applied to a 2D c-channel image resulting in a 2D feature map, because the convolution calculates a weighted sum over all c input channels. A convolutional layer typically combines d convolutions, resulting in d feature maps. Analogous to the fully connected layer in Equation (2.1), a convolutional layer is defined by doing a convolution on the input image, adding a bias term b and applying a non-linear activation function φ. The 2D convolutional layer is defined by the function $l_{\text{conv}} : \mathbb{R}^{h\times w\times c} \to \mathbb{R}^{((h-m+1)/s)\times((w-n+1)/s)\times d}$. The e-th feature map is defined by:

$$l_{\text{conv}}(X; \theta)^{(:,:,e)} = \varphi(X *_s A^{(e)} + b^{(e)})\,, \quad (2.4)$$

with $e = 1, ..., d$, kernel $A \in \mathbb{R}^{d\times m\times n\times c}$, bias $b \in \mathbb{R}^d$, stride $s \in \mathbb{N}$, and parameters $\theta = (A, b)$. In the following, we will call the above defined convolutional layer an $m \times n$ conv layer with d kernels and stride s.

Pooling layer In a typical CNN architecture convolutional layers are alternated with subsampling layers. These layers reduce the resolution of the feature maps to allow the next convolutional layer to learn more general high-level features that depend on a larger part of the original input image and are to some extend translation invariant (Goodfellow et al., 2016, p. 336).

The subsampling layer used in this thesis is the max pooling layer (max pool) which takes the maximum of each input feature map in non-overlapping rectangular neighborhoods. A pooling layer with a window size $s \in \mathbb{N}$ is a function $l_{\text{pool}} : \mathbb{R}^{h \times w \times c} \to \mathbb{R}^{(h/s) \times (w/s) \times c}$. The result at position (i, j) in feature map k is defined as

$$l_{\text{pool}}(X)^{(i,j,k)} = \max\{X(p, q, k) \,|\, p = is, ..., is + s\,,\ q = js, ..., js + s\}\,, \tag{2.5}$$

with $i = 1, ..., h/s$ and $j = 1, ..., w/s$. In the following we will call the above defined pooling layer a $s \times s$ max pool layer.

Upsampling layer A CNN with pooling layers can learn complex, translation invariant features, that encompass a large part of the image. However, pooling layers reduce the output resolution which is disadvantageous for segmentation problems. Upsampling layers can be used to increase the resolution of the output. The upsampling layer used in this thesis for the semantic segmentation architecture in Chapter 5 consists of an upsampling of the feature map by factor 2 followed by a 2×2 conv layer (Ronneberger et al., 2015). For input feature maps $X \in \mathbb{R}^{h \times w \times c}$ let $X_{\text{interp}}(i, j, k)$ be their interpolated indexing function using e.g., nearest neighbor interpolation. The upsampling function $\text{UP} : \mathbb{R}^{h \times w \times c} \to \mathbb{R}^{2h \times 2w \times c}$ is defined for the output at position (i, j) as

$$\text{UP}(X)^{(i,j)} = X_{\text{interp}}(\frac{i}{2}, \frac{j}{2})\,, \tag{2.6}$$

with $i = 1, ..., 2h$ and $j = 1, ..., 2w$. Using Equation (2.6), the upsampling layer $l_{\text{up}} : \mathbb{R}^{h \times w \times c} \to \mathbb{R}^{(2h-1) \times (2w-1) \times d}$ is defined for the e-th feature map as follows:

$$l_{\text{up}}(X; \theta)^{(e)} = \varphi(\text{UP}(X) * A^{(e)} + b^{(e)})\,, \tag{2.7}$$

with $e = 1, ..., d$, $A \in \mathbb{R}^{d \times 2 \times 2 \times c}$, $b \in \mathbb{R}^d$, and $\theta = (A, b)$. We will call this layer a 2×2 up-conv layer. Due to the d convolutions after the upsampling operation, it produces d output feature maps.

Batch normalization Ioffe and Szegedy (2015) introduced batch normalization (batch norm) to enable faster and more stable training of CNNs. It normalizes the outputs of a convolutional layer by normalizing each feature map individually. Let $\mathcal{X}_{\mathcal{B}} = \{X_1, ...X_m\}$ be the activations of the current batch that should be normalized, with elements $X \in \mathbb{R}^{h \times w \times d}$ and batch size m. During training, statistics for normalization are calculated on $\mathcal{X}_\mathcal{B}$ for each feature map:

$$\mu_\mathcal{B} = \frac{1}{mhw} \sum_{n=1}^{m} \sum_{i=1}^{h} \sum_{j=1}^{w} X_n^{(i,j,:)} \quad \in \mathbb{R}^d \quad \text{and} \tag{2.8}$$

$$\sigma_\mathcal{B}^2 = \frac{1}{mhw} \sum_{n=1}^{m} \sum_{i=1}^{h} \sum_{j=1}^{w} (X_n^{(i,j,:)} - \mu_\mathcal{B})^2 \quad \in \mathbb{R}^d \,. \tag{2.9}$$

Using only this normalization after a convolutional layer would mean that the layer cannot represent the unit transform anymore, because inputs are always transformed to have mean zero and variance one. In order to not loose representation capabilities, a pair of trainable parameters $\gamma_\text{BN} \in \mathbb{R}^d$ and $\beta_\text{BN} \in \mathbb{R}^d$ is introduced. These parameters allow scaling and shifting of the normalized values. The *batch normalizing transform* is defined for the e-th feature map of X with $e = 1, ..., d$ as

$$\text{BN}(X; \gamma_\text{BN}, \beta_\text{BN})^{(e)} = \gamma_\text{BN}^{(e)} \left(\frac{X^{(:,:,e)} - \mu_\mathcal{B}^{(e)}}{\sqrt{\sigma_\mathcal{B}^{2(e)} + \epsilon}} \right) + \beta_\text{BN}^{(e)} \,. \tag{2.10}$$

A convolutional layer with batch normalization is defined analogous to Equation (2.4) as

$$l_{\text{bn} \circ \text{conv}}(X; \theta)^{(e)} = \varphi(\text{BN}(X *_s A^{(e)}; \gamma_\text{BN}, \beta_\text{BN})) \,, \tag{2.11}$$

with $e = 1, ..., d$, input $X \in \mathbb{R}^{h \times w \times c}$, kernel $A \in \mathbb{R}^{d \times m \times n \times c}$, stride $s \in \mathbb{N}$, and parameters $\theta = (A, \gamma_\text{BN}, \beta_\text{BN})$. Note that the bias term b is omitted in this definition, because it would not have any effect due to the normalization. The parameter β_BN takes over the role as a bias in the convolutional layer with batch norm.

During testing, instead of using per-batch statistics for μ and σ^2, population statistics that are accumulated during training are used to normalize each batch. Using batch normalization after each convolutional layer of a CNN allows the use of higher learning rates and regularizes the model (Ioffe and Szegedy, 2015).

Activation functions Non-linear activations functions $\varphi : \mathcal{D} \to \mathcal{D}$ introduce non-linear properties to the neural network. Traditionally, saturating *sigmoid activation functions* were used in an element-wise manner, mapping each element x of $X \in \mathcal{D}$ to

$$\varphi(x) = \frac{1}{1 + \exp(-x)} \,. \tag{2.12}$$

However, convergence using saturating activation functions is slow and might lead to vanishing gradients during training (Glorot et al., 2011). Therefore, a non-saturating activation function, the *rectified linear unit (ReLU)*, is often used in CNNs:

$$\varphi(x) = \max\{0, x\}\,. \tag{2.13}$$

Softmax activation functions are applied to the output $x \in \mathbb{R}^k$ of a classifier to represent the probability distribution over k possible classes, with $\sum_{i=1}^{k} \varphi(x)^{(i)} = 1$. The softmax function for the i-th element of input x is given by

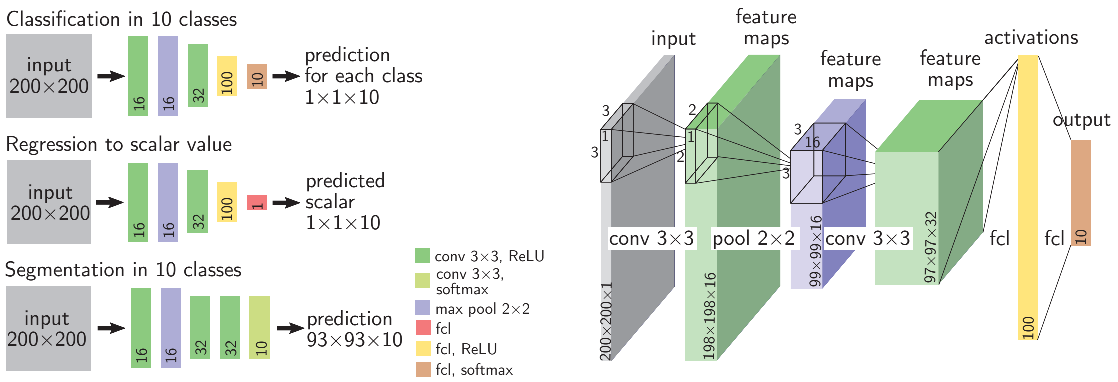
$$\varphi(x)^{(i)} = \frac{\exp(x^{(i)})}{\sum_{j=1}^{k} \exp(x^{(j)})}\,. \tag{2.14}$$

2.2.2 Convolutional neural network architectures

The most popular CNN architectures for image classification, regression and segmentation consist of the basic building blocks presented in Section 2.2.1. In the following, the characteristic structure of neural network architectures for these tasks is described (Figure 2.8). A common scheme for designing CNN architectures for the k-class *image classification* task is to use several blocks containing one or more convolutional layers (including a non-linear activation function and batch normalization) and a pooling layer which reduces the spatial resolution of the internal representations and allows to learn more complex features in the following layers. This convolutional part may be followed by one or more fully connected layers. The last layer contains a softmax activation function, resulting in a classification vector $z \in \mathbb{R}^k$, with $\sum_{i=1}^{k} z^{(i)} = 1$, where each entry $z^{(i)}$ can be interpreted as the probability that input X belongs to class i. For *regression tasks* a similar architecture is employed. In this case, the final fully connected layer outputs a scalar $z \in \mathbb{R}$. To allow the model to predict any scalar value, usually no activation function is used in the output layer.

Image segmentation can be transformed to a classification problem by classifying each pixel in the input image separately based on an image patch centered around the pixel. However, for the prediction of two neighboring pixels the input image patches overlap, resulting in overlapping internal filter responses. This results in inefficient, multiple computations of the same filter response. Thus, most segmentation architectures do not contain fully connected layers. This allows them to predict arbitrarily sized images and to share computations for the prediction of neighboring pixels. The output of a segmentation network is a segmented image $Z \in \mathbb{R}^{h' \times w' \times k}$ whose size depends on the size of the input. Image segmentation architectures are described in more detail in Section 2.2.3.

Deviating from the one-input-one-output scheme, it is possible to have neural network architectures with several input branches that are concatenated at some point in the network and with several output branches that branch off at some point in the network. During the design of neural network architectures, special attention needs



(a) Compact representation of different architectures. Each layer is drawn as a colored rectangle. The color of the rectangle determines its type, and the number in the rectangle is the number of output channels that this layer produces. Classification, regression, and segmentation networks can have the same first layers, but differ in later layers.

(b) Detailed representation of the classification architecture showing the internal feature maps. Exemplary convolutional kernels are shown in the upper corner of the feature maps. The model contains a block with a convolutional layer (16 kernels of size 3×3) and a 2×2 pooling layer, followed by a convolutional layer (32 kernels of size 3×3) and two fully connected layers with 100 and 10 nodes each.

Figure 2.8: Example CNN architectures for classification, regression, and segmentation.

to be paid that the output dimensions of one layer fit to the next layer. For example the input to a 2×2 max-pool layer needs to be of even size because the pooling summarizes blocks of 2×2 pixels. The architectures and their input sizes presented in this thesis were selected such that all internal layer outputs provide a valid input for the following layer.

2.2.3 Semantic segmentation architectures

Receptive field Segmentation architectures take an input image $X \in \mathbb{R}^{h \times w}$ and produce an output segmentation $Z \in \mathbb{R}^{h' \times w' \times k}$ whose size depends on the size of the input. It is important to note, that although arbitrarily large inputs are possible, only a fixed size local window of input pixels will contribute to any output pixel. This *receptive field* depends on the number and size of convolutional and pooling layers. Starting with a receptive field of 1, a valid convolutional layer (no padding) with a kernel of size $m \times m$ adds $m - 1$ pixels to the receptive field. A pooling layer with window size s multiplies the receptive field by factor s. The receptive field of a network is at the same time the minimal size of images that can be processed by the model. When constructing a semantic segmentation network it is important to consider the receptive field, because a too small receptive field might prevent that relevant features are learned and a too large receptive field restricts possible inputs for the model to very large images and increases the number of parameters.

Fully convolutional networks Long et al. (2015) proposed one of the early segmentation architectures, the fully convolutional networks (FCNs). A FCN uses, as the name suggests, only convolutional layers and pooling layers, no fully connected layers. The use of pooling layers results in a lower-resolution output segmentation. To have an output at the same resolution as the input, Long et al. (2015) use up-convolutions and skip connections from intermediate layers to the output layer resulting in a high-resolution output segmentation that contains local structures and global semantic information.

In order to produce even more detailed segmentations that are based on a large field of view, several improvements of the original FCN have been developed. Yu and Koltun (2015) propose to use dilated convolutions that increase the field of view without increasing the number of parameters and without decreasing the resolution. Similarly, L.-C. Chen et al. (2018) use atrous convolutions (“with holes") that effectively increase the size of the receptive field without adding more parameters to the model. The drawback of these two architectures is that all internal feature maps will have the size of the input, because no pooling layers are used. This increases the memory requirements of the models during training. For the large inputs that are required for histological area segmentation (Section 1.1) this is not feasible, as the model would not fit on the GPU anymore.

U-Net Ronneberger et al. (2015) proposed the *U-Net architecture* as a segmentation architecture that has a large receptive field and outputs precise segmentations at the same time. Since this architecture employs a downsampling path with pooling layers and an upsampling path with up-convolution layers, the memory requirements of the U-Net are less prohibitive. The U-Net is a popular architecture for medical segmentation tasks and is used in this thesis as the basis for the CNN model in Chapters 5 and 6.

The U-Net consists of a downsampling and upsampling path which are interconnected by copying activations from the downsampling to the upsampling path. Figure 2.9 shows the U-Net architecture as depicted in Ronneberger et al., 2015. Here, the “U" after which the architecture was named is clearly visible, with the downsampling path on the left side of the “U" and the upsampling path on the right. The drawings of the U-Net-based architectures in this thesis contain flattened representations of the “U" (e.g., see Figure 5.1).

The downsampling path consists of convolutional and pooling layers and calculates a low-resolution feature representation of the input image containing context

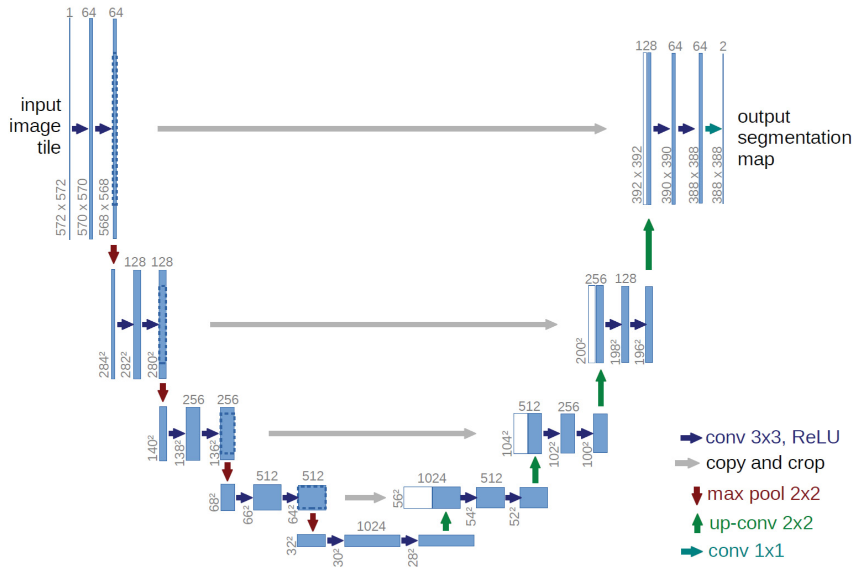


Figure 2.9: U-Net architecture for semantic segmentation (Ronneberger et al., 2015).[7] Contracting/downsampling path on the left, expanding/upsampling path on the right.

information. The layers are grouped in four blocks, with each block containing two convolutional layers with kernel size 3×3 and ReLU activation functions, and one max pooling layer with window size 2×2. Starting with 16 channels, the number of channels is doubled in each block. At the bottom of the "U", a block of two 3×3 convolutional layers is located which is followed by the upsampling path.

The upsampling path consists of convolutional and up-sampling layers (up-conv) that increase the resolution of the output and allow context propagation from the downsampling path to the output. Again, the layers are grouped in four blocks which mirror the blocks from the downsampling path. Each block contains one up-conv layer and two convolutional layers with kernel size 3×3 and ReLU activation functions. Each up-conv layer halves the number of channels.

Activations are copied after the last conv layer of each block in the downsampling path to the upsampling path and concatenated to the activations that result from the up-conv layer at the beginning of each block in the upsampling path. These skip connections allow the network to use high-resolution location information from the downsampling path to produce precise output segmentations.

[7] Reprinted from Ronneberger et al. (2015), Copyright (2015), with permission by Springer Nature.

2.2.4 Training deep neural networks

After an artificial neural network $f(X;\theta)$ has been defined, we want to find the optimal parameter configuration θ^* that allows the model to make correct predictions for every possible input. This optimization is done based on a training set consisting of n inputs $\mathcal{X} = \{X_1, ..., X_n\}$ and their corresponding groundtruth values $\mathcal{Y} = \{Y_1, ..., Y_n\}$. The optimal parameters θ^* are those that minimize a loss function L comparing the model predictions $f(\mathcal{X};\theta)$ with the true values $\mathcal{Y}$:

$$\theta^* = \mathrm{argmin}_\theta \, L(\mathcal{X}, \mathcal{Y}; \theta) \,. \tag{2.15}$$

Stochastic gradient descent In practice, the analytical minimization of Equation (2.15) is computationally intractable. Instead, an iterative numerical optimization procedure is used. Mini-batch stochastic gradient descent (SGD) is one of the standard optimization algorithms. Starting from (randomly or otherwise initialized) parameters θ^0, this algorithm iteratively updates the parameters based on a small sample (batch) $\mathcal{X}_\mathcal{B} \subseteq \mathcal{X}$ of the training data set with labels $\mathcal{Y}_\mathcal{B}$ by taking steps in the direction of the negative gradient $-\nabla_\theta L$ of the loss function with respect to parameters θ:

$$\begin{aligned} \Delta\theta^{i+1} &= -\eta \nabla_\theta L(\mathcal{X}_\mathcal{B}, \mathcal{Y}_\mathcal{B}; \theta^i) \\ \theta^{i+1} &= \theta^i + \Delta\theta^{i+1} \,. \end{aligned} \tag{2.16}$$

The size of the steps is controlled by the learning rate hyper-parameter $\eta \in \mathbb{R}^+$. One complete pass through all training data is called an *epoch*. In practice, several (usually between 10 and 1000) epochs are needed to train a CNN model.

Nesterov accelerated gradient In order to avoid excessive oscillations of the gradient due to a small batch size, SGD can be adapted to use momentum by letting the weight updates of the previous timesteps influence the current weight update. The optimization algorithm used in this thesis, Nesterov accelerated gradient (NAG) (Nesterov, 1983), can be seen as SGD with momentum (Sutskever et al., 2013). In Equation (2.16) the update of θ is the gradient of the loss function evaluated with the current parameters. In NAG the weight update starts from the previous weight update and is adjusted depending on the gradient of the loss function evaluated with the *updated* parameters $\theta^i + \Delta\theta^i$:

$$\Delta\theta^{i+1} = -\eta \nabla_\theta L(\mathcal{X}_\mathcal{B}, \mathcal{Y}_\mathcal{B}; \theta^i + \mu\Delta\theta^i) + \mu\Delta\theta^i \,. \tag{2.17}$$

The hyper-parameter $\mu \in \mathbb{R}_{>0}$ controls the influence of the momentum on the weight update. A NAG update can be described as taking a SGD step, and then correcting the update based on the gradient of the loss function at the new estimated position.

This allows the algorithm to change the updates in a responsive way, and to behave more stable for higher learning rates (Sutskever et al., 2013).

Back-propagation In order to calculate the update of θ, the gradient of the loss function with respect to θ needs to be calculated. The back-propagation algorithm (Rumelhart et al., 1986) allows the calculation of gradients in deep neural networks. It alternates a *forward pass* where an input is propagated through the network to calculate an output value, and a *backward pass* where the resulting loss is propagated back through the network. It uses the chain rule of calculus to iteratively compute the gradients for each layer.

Cross-entropy loss The cross-entropy loss is commonly used for classification and segmentation tasks. It calculates the pixel-wise difference between groundtruth $Y \in \mathcal{Y}$ and predicted output $f(X;\theta)$ for inputs $X \in \mathcal{X}$. If classes are imbalanced or if one class should get special importance, the contribution of each class to the loss can be weighted with a weight $\omega_j \in \mathbb{R}$ for each class j. The cross-entropy loss is defined as

$$L(\mathcal{X},\mathcal{Y};\theta) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k}\omega_j Y_i^{(j)} \log(f(X_i;\theta)^{(j)}) \tag{2.18}$$

for a k-class classification problem with $n = |\mathcal{X}|$ evaluation samples X_i, and groundtruth values Y_i given in one-hot encoding. This loss is used in the profile classification and area segmentation models from Chapters 4 and 5.

L1 loss Regression models output a scalar value y which can be compared with the desired groundtruth value using for example an L1 loss:

$$L(\mathcal{X},\mathcal{Y};\theta) = \frac{1}{n}\sum_{i=1}^{n}\|f(X_i;\theta) - y_i\|_1 . \tag{2.19}$$

Using the L1 norm instead of the L2 norm has the effect that large errors are not overly penalized. This can be advantageous in some cases (see Section 6.1).

Overfitting and weight decay If a model correctly classifies every training sample, but does not generalize to unseen validation samples, this model is *overfitting* to the training data. This happens when the training dataset is small and the model has enough parameters to fit them to the noise in the training data instead of learning generalizable features. There are several regularization techniques to combat overfitting. One popular technique is called weight decay which adds an L2 regularization of the parameters θ to the loss function:

$$\hat{L}(\mathcal{X}, \mathcal{Y}; \theta) = L(\mathcal{X}, \mathcal{Y}; \theta) + \lambda \|\theta\|_2^2 \,, \tag{2.20}$$

with $\lambda \in \mathbb{R}^+$. In general terms, the weight decay prevents that the weights of the neural network keep growing in order to fit perfectly to the training data. In practice, this reduces overfitting (Krogh and Hertz, 1992).

Hyper-parameter tuning Hyper-parameters are properties of the model or of the optimization algorithm that are fixed before training. Hyper-parameters for neural networks include model parameters like the number and type of layers and the number and size of convolutional kernels, as well as training parameters like learning rate η, momentum μ, and weight decay factor λ. Unfortunately there is no one-fits-all approach that can be used to determine the hyper-parameters. Instead, hyper-parameters need to be determined for each task and training dataset. This is a non-trivial task, especially since a single hyper-parameter cannot be considered independently of the other hyper-parameters.

A common strategy for choosing hyper-parameters, which is also used in this thesis, is to start from values found in literature (e.g., a neural network architecture that was previously used on a similar task) and to empirically optimize the hyper-parameters from these initial values on a validation set, using either manual, random, or grid search. However, evaluating many different configurations of hyper-parameters is time-consuming, since for each hyper-parameter configuration a neural network needs to be trained. A promising new approach is to use Bayesian optimization to select the hyper-parameters (Snoek et al., 2012). The idea is build a probabilistic model of the objective function and to use this model to choose promising hyper-parameters that should be evaluated next. This informed choice of the next hyper-parameter configuration can in practice reduce the amount of different hyper-parameter configurations that need to be evaluated compared to grid search and random search.

2.2.5 Evaluating the training process in deep neural networks

In order to evaluate the performance of a trained k-class semantic segmentation model $f(X;\theta) \in \mathbb{R}^{h' \times w' \times k}$ several metrics can be calculated. Let $\mathcal{X} = \{X_1, ..., X_n \,|\, X_i \in \mathbb{R}^{h \times w \times c}\}$ be a dataset of n images and $\mathcal{Y} = \{Y_1, ..., Y_n \,|\, Y_i \in \{0,1\}^{h' \times w' \times k}\}$ their corresponding groundtruth labels for the k classes in one-hot encoding. Define $\mathcal{G}_k$ as the set of all indices of pixels labeled as class k, $\mathcal{G}_k = \{(n,i,j) \,|\, Y_n^{(i,j,k)} = 1 \,,\, Y_n \in \mathcal{Y} \,,\, i = 1, ..., h \,,\, j = 1, ..., w\}$, and $\mathcal{P}_k$ as the set of all indices of pixels predicted as class k, $\mathcal{P}_k = \{(n,i,j) \,|\, \operatorname{argmax} f(X_n;\theta)^{(i,j)} = k \,,\, X_n \in \mathcal{X} \,,\, i = 1, ..., h \,,\, j = 1, ..., w\}$, and $s = n \cdot h \cdot w$ as the number of all evaluated pixels.

Accuracy The accuracy is the most commonly used score in classification and segmentation settings and is defined as the ratio of the number of correctly classified pixels to the number of all pixels:

$$\text{acc} = \frac{\sum_{i=1}^{k} |\mathcal{G}_i \cap \mathcal{P}_i|}{s} \,. \tag{2.21}$$

The accuracy has the drawback that it only focuses on correctly classified pixels, and not on falsely classified ones. Consider for example a test dataset $\mathcal{X}$ with ten samples, nine of which are of class a and one of class b. A model that predicts class a for every sample intuitively does not perform well, but gets an accuracy of $(9 + 0)/10 = 0.9$. In this thesis the accuracy is only used to control progress during training on a left out validation set.

Dice score For assessing the final performance of a model, the Dice score is calculated. The Dice score (or Sorensen-Dice index, F1 score) is defined as the harmonic mean between precision and recall. Define the precision for class i as the ratio of the number of pixels correctly predicted as class i to the number of all pixels predicted as class i: $\text{precision}_i = |\mathcal{G}_i \cap \mathcal{P}_i|/|\mathcal{P}_i|$. Define the recall for class i as the ratio of the number of pixels correctly predicted as class i to the number of all pixels labeled as class i: $\text{recall}_i = |\mathcal{G}_i \cap \mathcal{P}_i|/|\mathcal{G}_i|$. With this we can define the Dice score for class i as:

$$\text{Dice}_i = 2\frac{\text{precision}_i \cdot \text{recall}_i}{\text{precision}_i + \text{recall}_i} \,. \tag{2.22}$$

As the overall aim of the segmentation is to correctly predict as many pixels as possible, a weighted mean of the per-class Dice scores is reported in the evaluations:

$$\text{Dice} = \frac{\sum_{i=1}^{k} |\mathcal{G}_i| \cdot \text{Dice}_i}{s} \,. \tag{2.23}$$

This metric considers both precision and recall and thus outputs a more realistic score for the above toy example with $\text{Dice}_a = 2(0.9 \cdot 1)/(0.9 + 1) = 0.95$ and $\text{Dice}_b = 0$ resulting in a final weighted Dice score of $(9 \cdot 0.95 + 1 \cdot 0)/10 = 0.85$.

Weighted misclassification error All errors influence the Dice score equally. However, in the context of brain area segmentation, the labels follow a certain topology. The evaluation score should consider this topology by weighting different errors differently. When two models have the same Dice score, the model that makes “more plausible" errors is preferable. In the context of brain area segmentation, errors are more plausible, when they confuse topologically close labels as opposed to topologically distant labels.

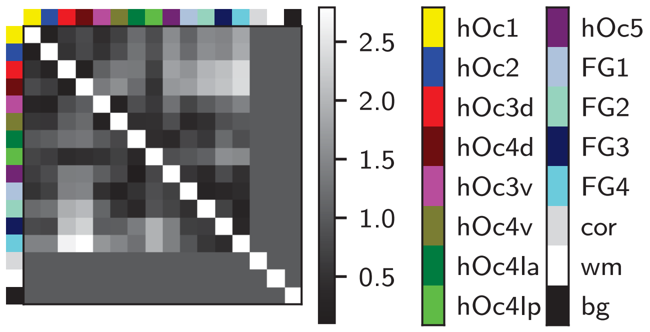


Figure 2.10: Normalized distances ω_{ij} between brain areas. Calculated using JuBrain atlas maximum probability maps.

The distance of brain areas can be calculated using the JuBrain atlas (Section 2.1.5): For every area pair (i, j), calculate the shortest distances of every point in area i to area j. Let d_{ij} be the mean of these distances. Define "severity" of an error as the normalized distance between the true area i and the predicted area j:

$$\omega_{ij} = \frac{d_{ij}}{\frac{1}{k^2}\sum_{i=1}^{k}\sum_{j=1}^{k} d_{ij}}\,. \tag{2.24}$$

For classes k not present in the JuBrain atlas (white matter, background and unknown cortex), set the weight $\omega_{k:} = 1$. Figure 2.10 shows the normalized weight matrix which represents the topology of areas in the brain.

We can now weigh different errors by their "severities" with the weighted misclassification error:

$$\text{err} = \sum_{\substack{1\le i,j\le k,\\ i\ne j}} \omega_{ij}|\mathcal{G}_i \cap \mathcal{P}_j|\,. \tag{2.25}$$

This error can be used to determine how well a model has understood the topology of areas.

Confusion matrix A confusion matrix is a helpful tool to assess the per-class performance of a multi-class classification model. Figure 2.11 shows an example confusion matrix. Each row represents the instances (pixels) in a groundtruth class, and each column represents the instances in a predicted class. Thus, the cell in row i and column j contains the number of examples from class i that were predicted as class j. Entries on the diagonal of the confusion matrix represent correct predictions. The confusion matrices shown in this thesis visualize the number of instances on a per-class normalized gray scale.

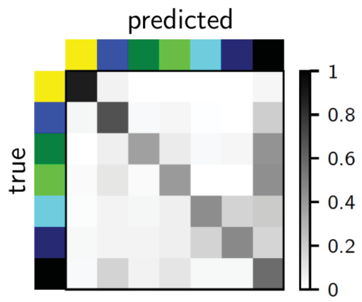


Figure 2.11: Confusion matrix. The entry in row i and column j represents the normalized amount of examples from class i classified as class j. This allows to assess the types of errors that a model makes.

Visualization and saliency After training and evaluating a model on a test set, we might want to look inside the model to understand how it comes to its decision. There are several techniques for this type of analysis. In the following feature map visualization and saliency calculation are used. Let $\mathrm{FM}_i(X) = l_i(l_{i-1}(...l_1(X)...))$ be the feature maps of the i-th layer for the input image $X \in \mathbb{R}^{h\times w\times c}$. Visualizing the internal representations $\mathrm{FM}_i(X)$ of the model for a given input X shows the features that the model computes at each layer and how they are combined to form the output prediction. Analyzing the feature maps gives us an idea of how the model works and what features each layer focuses on.

Saliency techniques go one step further and ask: Which parts of the input were most relevant for making the prediction? In Chapter 4 Grad-CAM (Selvaraju et al., 2017) is used to answer this question. Grad-CAM provides coarse localization maps S for k-class classification models showing the relevant parts of the input X for classifying it into class k. It is especially useful to visualize saliency for models that contain fully connected layers, which do not preserve the localization of features. To calculate localization maps S, first the gradient of the score for class k of the prediction $z \in \mathbb{R}^k$, $z^{(k)}$, is calculated with respect to the feature maps $\mathrm{FM}_l(X) \in \mathbb{R}^{h'\times w'\times d}$ of the last convolutional layer l. Using this gradient, global average-pooled importance weights α_m are obtained for each individual feature map $\mathrm{FM}_l(X)^{(:,:,m)}$:

$$\alpha_m = \frac{1}{h'\cdot w'}\sum_{i=1}^{h'}\sum_{j=1}^{w'}\frac{\partial}{\partial \mathrm{FM}_l(X)^{(i,j,m)}}z^{(k)}\,. \tag{2.26}$$

Grad-CAM is then calculated as a weighted combination of the feature maps $\mathrm{FM}_l(X)$ by their importance weights α:

$$S = \max\left\{\sum_{m=1}^{d}\alpha_m \mathrm{FM}_l(X)^{(:,:,m)}\,,\,0\right\}\,. \tag{2.27}$$

With this definition, the saliency map S shows which parts of the input contain pixels that should be increased in order to increase the prediction score $z^{(k)}$ for class

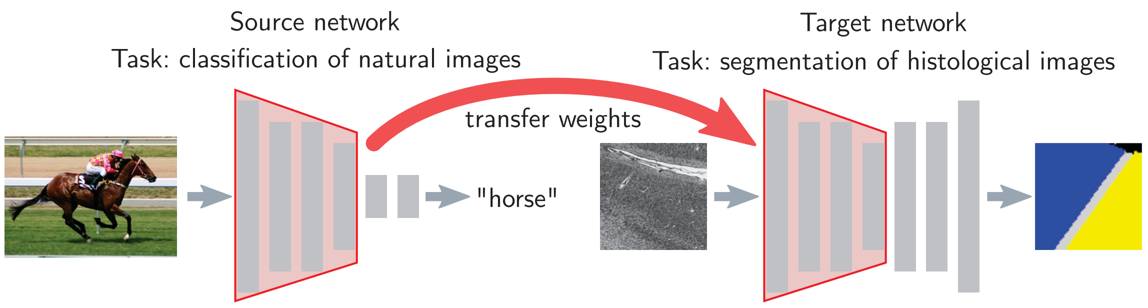

Source network
Task: classification of natural images
transfer weights
"horse"
Target network
Task: segmentation of histological images

Figure 2.12: Transfer learning works by initializing a target network with weights of a source network which was trained on an auxiliary task. The transferred weights can be adapted during training (fine-tuning) or be frozen.

k. Thus, S can give an indication of the parts of the input that contain important concepts used by the model to classify the input in class k.

2.2.6 Transfer learning

During training, the parameters θ of a neural network are iteratively optimized starting from initial parameters θ^0. Usually, the weights of the fully connected and convolutional layers are set to small random values and the biases are set to constant values (Glorot et al., 2011). However, training deep networks from scratch can be a difficult task. First, the models require a large amount of labeled training data to find the optimal parameters θ^*. Second, training the models is computationally expensive, resulting in long training times. Third, there are often overfitting and convergence issues that require repetitive adjustments in architecture and hyper-parameters.

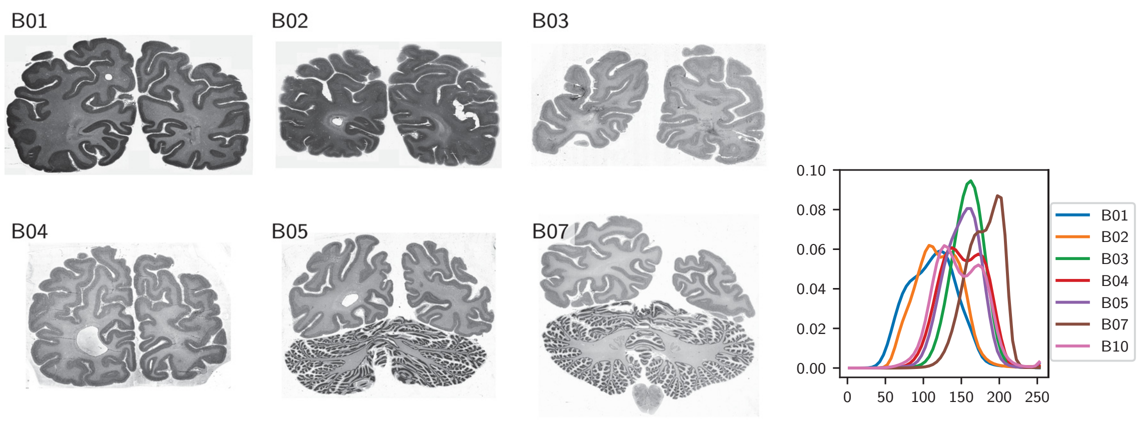
To avoid these issues, models can be initialized with parameters obtained by training an auxiliary task on a different domain where training data is abundant. This *transfer learning* scheme is visualized in Figure 2.12 and is justified as follows: In the first layers, a CNN learns general features that are independent of the actual task and dataset. In later layers these general features transition to task- and domain-specific features (Yosinski et al., 2014). Using transfer learning, it is possible to leverage the early, general features for other tasks. To do this, the weights of the first n layers of a source network are copied to the first n layers of the target network. The parameters of the remaining layers of the target network are randomly initialized as described above and trained on the target task. During training, the copied layers can either be adapted (*fine-tuned*) together with the randomly initialized layers or the weights can be frozen. In the latter case the base network is used as a feature generator for the target task. Yosinski et al. (2014) show that the performance of a network with frozen features can drop due to splitting between co-adapted neural network nodes. Thus, generally fine-tuning the entire target network should be preferred. Nevertheless, Sharif Razavian et al. (2014) have shown that transfer learning with frozen features is a good baseline for classification, recognition, and retrieval tasks.

2.3 Datasets

Training machine learning models requires a dataset of inputs and (optionally) ground-truth labels. For the task of automatic brain area segmentation datasets $\mathcal{X}$ containing patches extracted from cell-body stained histological sections are used (see Section 2.1.1, and Figure 2.2). The datasets are extracted from sections of the brain collection of the INM-1. This brain collection contains cell-body stained histological sections of several human brains. The brains were assigned an unique identifier “BX" for brain number x. The brains used in this thesis were cut in 20 µm thin coronal sections which were numbered starting from the occipital pole (the “back" of the brain). This means that the section number i has a distance of $0.02i$ mm to the occipital pole. The histological sections were scanned in a light microscope at 1 µm resolution.

In order to define a dataset, first a set of high-resolution histological sections (comprising one or several brain volumes), a *collection*, is specified. These collections are split in train, validation, and test sections. Then, the train, validation and test datasets are extracted from the sections of the respective split. The train dataset $\mathcal{X}^{\text{tr}}$ is used to train the models, the validation dataset $\mathcal{X}^{\text{val}}$ is used to tune the hyper-parameters (e.g., the architecture and the learning rate), and the test dataset $\mathcal{X}^{\text{te}}$ is used to compare the different models. In the following chapters, supervised and unsupervised models are trained and evaluated on several datasets extracted from different collections. This section introduces these collections and data associated with them. Each histological section in the collections was visually assessed and sections with major artifacts or staining irregularities were sorted out. The train, validation and test datasets sampled from these collections are introduced in the respective experiment chapters and summarized in the Glossary on page 145.

The sections used to train the area segmentation models are introduced in Section 2.3.1. The projected probabilistic maps from the JuBrain atlas that are used as additional input for these models are described in Section 2.3.2. Section 2.3.3 introduces the large collection that is used to train the unsupervised models and contains 3727 sections of one human brain volume. Based on this collection, a gray/white matter segmentation was calculated for the other collections (Section 2.3.4).



(a) Section 1141 in different brains **(b)** Gray value histograms

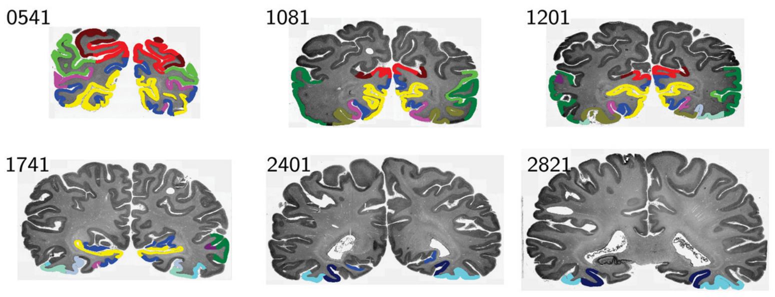
Figure 2.13: Inter-individual variability of brains in the `AtlasBrainsCollection`. Brains differ in visual appearance and overall gray value histograms. Histograms were calculated from all pixels inside the tissue for a specific brain and normalized by the total number of pixels in this brain for better comparison between the brains.

2.3.1 Labeled dataset for brain area segmentation

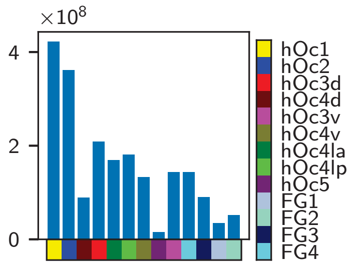
The brain area segmentation models are trained on the `AtlasBrainsCollection` and `B01Collection` to segment areas of the visual system (Section 2.1.5). The `AtlasBrainsCollection` consists of 450 sections from the visual system of brains B01, B02, B03, B04, B05, B07, and B10 from the brain collection of the INM-1. Consecutive sections have a distance of 1.2 mm (up to every 60th physical section is in the dataset). The `B01Collection` is a subset of this collection which only contains sections from brain B01 (62 sections in total).

Although the histological preparation of all brains was the same, they are quite varied in terms of appearance due to inter-subject variability and staining differences. Figure 2.13 shows example sections from each brain and average gray-value histograms of all sections in each brain. Since brain area segmentation is an immensely difficult task with large variations in each brain area due to oblique cuts, curvature, and artifacts, the `B01Collection`, which does not contain additional variation due to inter-individual differences, is used to train the supervised models in Chapters 4 and 5. In Section 6.3 the performance of the models on the more difficult `AtlasBrainsCollection` is shown.

For each section in the datasets partial manual delineations of the 13 structural areas of the visual system (introduced in Section 2.1.5) are available. The areas were manually annotated by expert neuroanatomists and borders were confirmed by the observer-independent method of Schleicher, Amunts, et al. (1999). Since the annotations are partial, not every part of the cortex on the sections is labeled,



(a) Exemplary sections with manual labels



(b) Label frequency

Figure 2.14: Labels of the `B01Collection`. Partial manual brain area annotations confirmed by the observer-independent method. Areas differ in size.

and not every occurrence of a specific visual area is labeled. Figure 2.14a shows examples of the available annotations on sections from the `B01Collection`. Due to the different size of the brain areas, the annotated pixels per area are not evenly distributed. Areas `hOc1` and `hOc2` are most frequent, and area `hOc5` is the smallest. Figure 2.14b shows the relative frequencies of each area in the `B01Collection`. The label distribution between the `B01Collection` and the `AtlasBrainsCollection` does not change significantly.

Train/val/test split The collections are split in train, validation, and test sections by alternating 4 training sections, 1 validation section, 4 training sections, and 1 test section. In this manner 80% of the annotated sections are used for training, and 20% for validation and testing. For the `AtlasBrainsCollection` this results in 303 train, 75 validation, and 72 test sections. For the `B01Collection` 42 sections are in train, 10 sections in validation, and 10 sections in test.

2.3.2 Projection of JuBrain probabilistic maps to the labeled dataset

Chapter 5 combines high-resolution texture inputs with prior knowledge of which areas are likely to occur in this input. For this information, the JuBrain atlas is used. This atlas contains probabilistic maps for the location of each area in a template brain volume (Section 2.1.5). Using standard image registration techniques (Section 2.1.4), the JuBrain atlas (Section 2.1.5) is registered to each section of the `B01Collection` and the `AtlasBrainsCollection`. To get probabilistic maps on individual histological sections, each section of the `B01Collection` and the `AtlasBrainsCollection` is registered to the template brain volume (see Section 2.1.4 for information about image registration). Since we are only interested in rough estimates of the atlas data on the individual section, an affine registration (i.e., shifting, rotating, scaling, and

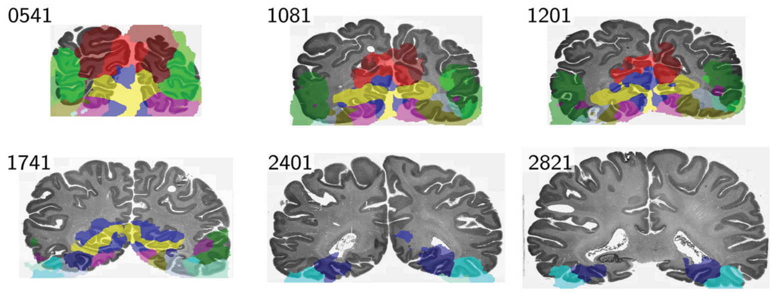


Figure 2.15: Projection of JuBrain atlas to several sections of the B01Collection. The maximum probability map thresholded at probabilities > 0.4 is shown. Compared to manual delineations (Figure 2.14a), the information provided by the atlas is very coarse.

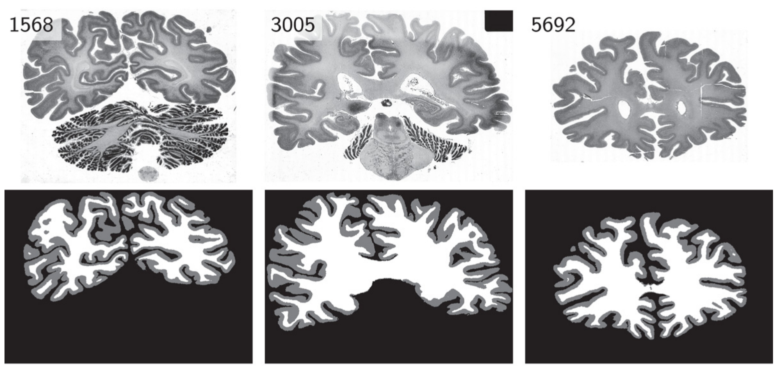
shearing of the atlas data) was calculated, which can be performed with reasonable speed. Using this registration, the JuBrain probabilistic maps are transformed to the individual sections. The transformation results in probabilistic maps for each of the 13 areas annotated in the visual system (Section 2.1.5) at 8 µm resolution. For every location, the probabilistic map of area `a` contains the prior probability that this location on the high-resolution section shows area `area` given the averaged maps of the JuBrain atlas. Figure 2.15 shows the transformed JuBrain atlas on several sections from the B01Collection. The probabilistic maps are used in Chapter 5 as additional input to the segmentation network.

2.3.3 Unlabeled dataset for unsupervised feature learning

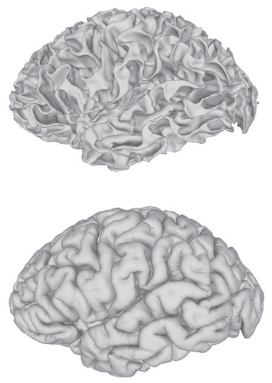
The BigBrainCollection contains 3727 high-resolution sections from the BigBrain (Amunts, Lepage, et al., 2013).[8] Consecutive sections in this collection have a distance of 20 µm. Brain area labels were not included in this collection, however a 3D reconstruction at 20 µm resolution is available.[9] For the 3D reconstructed volume, 200 µm resolution meshes of the gray/white matter border and of the pial surface exist (L. Lewis et al., 2014). These meshes are sliced and transformed to the high-resolution histological sections using the existing registration. This results in 200 µm resolution gray/white matter masks for each section. Figure 2.16 shows example sections from this dataset, their gray/white matter masks and the gray/white matter border and pial surface meshes. This dataset is used in Chapter 6 to pre-train models on a self-supervised task related to area segmentation.

[8]The entire BigBrain volume contains 7404 sections. The BigBrainCollection contains those sections that were at the time of dataset assembly scanned at 1 µm resolution, that did not have major artifacts, and for which the gray/white matter masks could be correctly calculated.

[9]Thanks to Claude Lepage (McGill University) for providing the transformation files to the 3D BigBrain volume.



(a) Example sections with gray/white matter masks



(b) Gray/white matter border (top) and pial surface (bottom) meshes

Figure 2.16: The `BigBrainCollection`

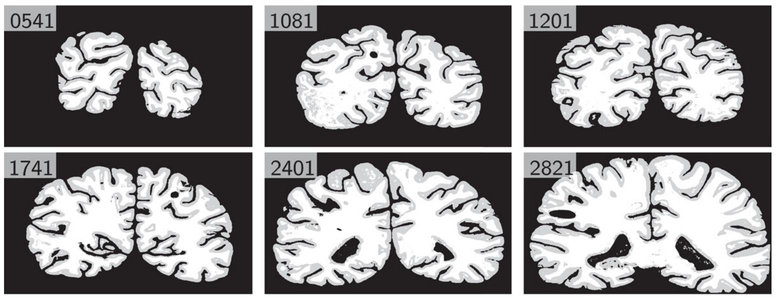


Figure 2.17: Automatic gray/white matter segmentation of several sections of the `B01Collection`

Train/val/test split This dataset is split in train/val and test sections by alternating 12 train/val sections and 1 test section. Thus, 3416 of the 3727 sections in the dataset are in the train/val split and 311 sections in the test split. For the evaluation of the self-supervised models in Section 6.4 the dataset is further limited to contain only the left or the right hemisphere. The models used for pre-training the segmentation models in Section 6.3 are trained and validated on 1796 sections with section numbers 1-3000 from the occipital lobe of the brain (resulting in 1646 sections in the train/val split and 150 sections in the test split).

2.3.4 Gray/white matter segmentation of labeled dataset

Based on the gray/white matter masks of the `BigBrainCollection`, a CNN was trained for automatic gray/white matter segmentation of the `AtlasBrainsCollection`. This model was developed by Christian Schiffer (INM-1, FZJ) in the context of his master's thesis and is described in Schiffer (2018). The topic of the master's thesis was the transferability of neural network models from one brain volume to other brain volumes. Average gray values were identified as one of the main sources of differences between different brain volumes. One possible solution is based on gamma augmentation (Equation (5.3), Page 78) of the input data to simulate gray value distributions of other brain volumes. Fifty percent of the inputs were gamma augmented with $\gamma \in [1.5, 2.5]$. With this augmentation, a `BaseNetwork` model (Section 5.1.1) was trained on 60 sections from the `BigBrainCollection` on the gray/white matter segmentation task. This model could be successfully used to segment sections from the `AtlasBrainsCollection`. Figure 2.17 shows the resulting segmentation of sections from the `B01Collection`.

3 Related work

The aim of this thesis is the development of automatic methods for the segmentation of brain areas on high-resolution histological sections. Section 1.1 identifies five key challenges of this aim: localization of cortical areas in histological sections, respecting the topology of brain areas, dealing with artifacts in histology, limited training data, and large input sizes. Other researchers already faced several of these challenges in slightly different contexts. This chapter introduces these related works and motivates the approaches presented in this thesis.

Previous approaches localize cortical areas by analyzing the cortical laminae (Section 3.1). The challenge of topologically consistent predictions is discussed by several works on automatic brain MRI segmentation (Section 3.2). As histological sections have distinctive characteristics that distinguish them from natural images, Section 3.3 introduces automatic histological image analysis methods that need to deal with large input sizes and staining artifacts. Finally, Section 3.4 highlights several approaches for learning deep models with limited training data by using data augmentation, transfer learning, and unsupervised learning.

3.1 Analysis of the laminar structure of the cortex

Cortical areas are distinguished by the arrangement of neurons in cortical laminae. The current state of the art method for observer-independent border detection by Schleicher, Amunts, et al. (1999) (see Section 2.1.3) exploits this structure by calculating GLI profiles that contain approximate cell-densities for each lamina, and comparing neighboring profiles to find significant differences.

Thus, in order to segment the cortex, in theory it should be sufficient to identify the cortical laminae and find the places where their configuration changes. This direct approach to automatic brain area mapping was explored by Adamson et al. (2005). They used a tracking method to follow along cortical laminae and detect changes in the thickness of the laminae. With this, they were able to identify some borders between cortical areas. Although promising, this work has several drawbacks that it shares with the method of Schleicher, Amunts, et al. (1999): It will only work on perpendicularly sectioned parts of the cortex, where all laminae are clearly distinguishable. Due to the highly folded surface of the brain, however, many parts of the cortex on any 2D section will be oblique. Additionally, the tracking of the lamina will be affected by any artifact that crosses it.

Wagstyl, Lepage, et al. (2018) take a slightly different approach to identify laminae in the cortex: The authors use the reconstructed 20 µm BigBrain volume (Amunts, Lepage, et al., 2013) to extract profiles as traversals of the cortex in 3D. Then they use an automatic method that identifies the different laminae using the profiles as input. By leveraging the 3D data of the BigBrain this approach has the advantage that there is no obliquely sectioned data. However, this approach only identifies different laminae, but does not attempt to parcellate brain areas based on this information.

The previously introduced approaches have in common that they use high-resolution histological sections to extract features that encode information about the laminar structure of the cortex. This has the advantage of directly encoding the relevant information in a feature vector and removing any information about the curvature and other anatomical landmarks in the tissue. Motivated by this, Chapter 4 introduces a CNN that learns to classify different cortical areas based on such features as input. In Chapter 5 this feature-based approach is then compared with the image-based approach introduced in Chapter 5.

3.2 Brain area parcellation in magnetic resonance imaging

To the best of our knowledge, there are no previous works on fully automatic segmentation of brain areas in histological sections. However, there are several approaches for automatic segmentation of brain areas in MRI that respect the topology of their labels. They can be divided in two categories: 1. *Atlas registration approaches*, which directly model the topology of areas by registering an atlas to the volume of interest (Section 3.2.1) and 2. *Deep Learning approaches*, which indirectly enforce spatial consistency and topological correctness by including context or other prior knowledge (Section 3.2.2).

3.2.1 Atlas registration approaches

Atlas registration approaches are commonly used to segment brain MRI volumes (Desikan et al., 2006; Fischl, Salat, et al., 2002; Heckemann et al., 2006; Klein et al., 2005; H. Wang and Yushkevich, 2013). Given one or more existing segmentations of source volumes (the atlases), a registration from these volumes is calculated to the target volume that should be segmented (Rohlfing et al., 2005). Figure 3.1 visualizes this process. The individual methods mostly differ in the number of atlases that are used for a new segmentation, the registration methods used for mapping the atlases

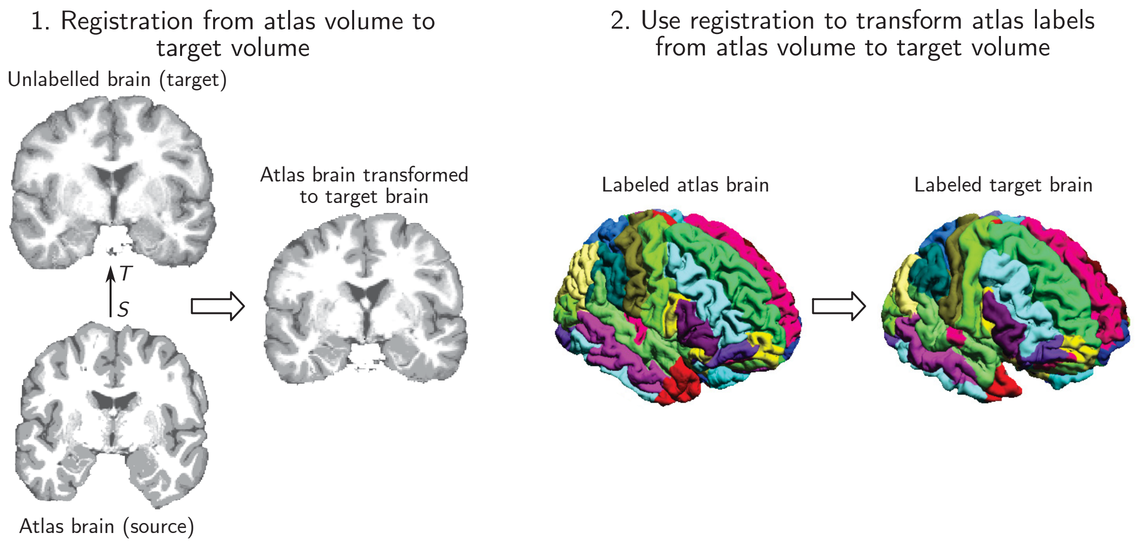

1. Registration from atlas volume to target volume
Unlabelled brain (target)
T
S
Atlas brain (source)
Atlas brain transformed to target brain
2. Use registration to transform atlas labels from atlas volume to target volume
Labeled atlas brain
Labeled target brain

Figure 3.1: Altas registration approach for segmenting brain MRI volumes.[1] The atlas volume is registered to a new subject (target) enabling the transfer of labels defined on the atlas to the new subject.

to the new image, and the way the labels are fused for generating topologically correct segmentations. See Section 2.1.4 for a short introduction to image registration. Due to the high inter-individual variability of folding patterns, brain-to-brain registration is a challenging task. To get good correspondence between source and target volumes, a simple intensity-based approach is often not sufficient. Instead, landmark-based registration can be used to align the sulci of the source and target volumes (Mangin, Lebenberg, et al., 2016).

Atlas registration methods have the advantage that they produce topologically correct parcellations since the new segmentation is always based on topologically correct atlases. However, they have several drawbacks that prevent us to use them for high-resolution histological parcellation: The quality of the resulting segmentation is highly dependent on the registration, which is very difficult to calculate given the inter-individual differences between brains. Additionally, the atlas registration approaches do not take into account the relationship between image data and the atlas labels. Instead they assume that the labels are correlated with anatomical landmarks and use these landmarks (e.g., sulci and gyri) to calculate the mapping from atlas to new MRI volume. This limits the accuracy of the resulting segmentation.

For high-resolution histological image parcellations, the correlation between labels (brain areas) and anatomical landmarks cannot be assumed (Amunts and Zilles, 2015). On the contrary, the mappings of cytoarchitectonic areas can be used to analyze the correlation between labels and anatomical landmarks. Thus, for the

[1]Brain images retrieved from `http://brainsuite.org/processing/svreg/`.

purposes of this thesis, pure atlas-based methods are not appropriate. However, Chapter 5 shows that combining high-resolution texture features and atlas data is advantageous for the performance of automatic area segmentation.

3.2.2 Deep learning approaches

Akkus et al. (2017) provide a good overview of deep learning approaches for brain MRI segmentation. In this section, the focus is on models that segment the brain into different tissue classes or brain areas and achieve spatially consistent predictions. Several neural network architectures that are trained on 2D or 3D image patches extracted from the MRI volume to segment it in different classes have been proposed. Dividing a large input image or volume in smaller image patches and training a neural network on these patches is a common strategy to deal with large inputs and is further discussed in Section 3.3.1. The approaches presented in this section combine the image patches with lower resolution context patches (de Brébisson and Montana, 2015; Moeskops et al., 2016) and/or with location information (de Brébisson and Montana, 2015; Wachinger et al., 2017).

De Brébisson and Montana (2015) train a neural network for anatomical brain area segmentation in $k = 134$ classes. They combine local information with global information to enforce spatial consistency in the resulting segmentation. As local information, small 3D patches centered around the pixel of interest are used. For the global information, larger, low resolution orthogonal 2D patches are used. To further improve the spatial consistency, the location of the input patch p is encoded as the Euclidean distance between coordinate p and the centroids c_k of each of the k classes. During training, the centroid c_k is calculated as the center of mass of all voxels labeled with class k. Since during evaluation labels are usually not available, they use an iterative procedure to estimate the centroids from the automatic segmentation. The distance to the centroids thus encodes the location relative to all regions and can be used by the model to create spatially consistent segmentations.

Wachinger et al. (2017) consider a $k = 25$ class segmentation task of MRI volumes (see Figure 3.2). To deal with the imbalance of background and brain tissue on an MRI scan, in a first step a CNN is trained to segment foreground and background, and in a second step another CNN is trained to segment the foreground in the 25 different tissue classes. To allow the network to model more complex relationships, a deeper network than the network proposed by de Brébisson and Montana (2015) and large 3D input patches are used. Wachinger et al. (2017) train the model to predict not only the center voxel of a given patch, but also its 6 direct neighbors and average predictions for each voxel for the final segmentation. This multi-task formulation can be considered an ensemble method and results in better segmentation performance. Similar to de Brébisson and Montana (2015) location information is

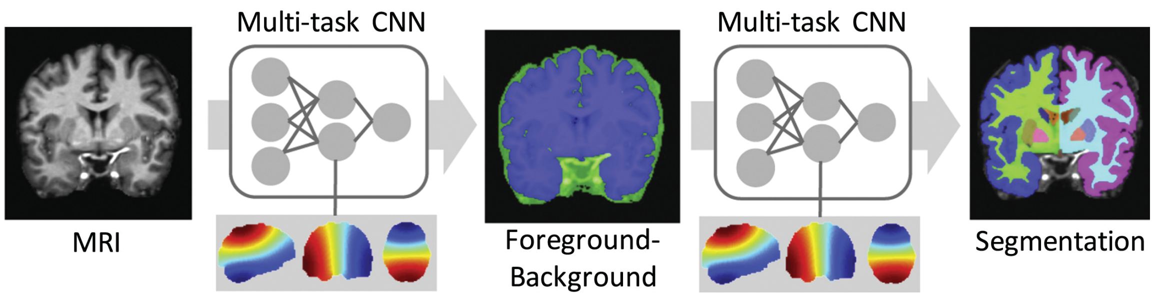


Figure 3.2: Model proposed by Wachinger et al. (2017) for MRI segmentation.[2] First, a background/foreground segmentation is calculated. Then, the foreground is segmented in 25 tissue classes. Spectral coordinates are provided to the model to facilitate the localization of the input patches in the brain volume.

included for each patch to resolve potential ambiguities. Here, as location information an intrinsic brain parametrization given by spectral coordinates is used. The spectral coordinates are defined as the eigenvectors of the Laplacian matrix of the graph representing the brain mask and are an alternative to Cartesian coordinates. To get a smooth output segmentation, a conditional random field (CRF) is applied to the predicted segmentation. CRFs model the spatial relationship between neighboring pixels and can for example penalize the assignment of different classes to neighboring pixels to reduce noise and avoid scattered segmentations (Lafferty et al., 2001). Thus, using a CRF on top of the neural network predictions can help to obtain smoother segmentations.

Moeskops et al. (2016) train a CNN to segment an MRI brain volume into 6-8 tissue classes. This method uses input patches at three different scales and sizes to provide context. In contrast to de Brébisson and Montana (2015) and Wachinger et al. (2017) they do not include explicit location information but rather argue that the large field of view of their model naturally leads to smoother segmentations. However, as only the segmentation into large tissue classes is considered, this finding is not directly applicable to brain area segmentation.

In contrast to the previous approaches, Glasser et al. (2016) leverage multi-modal data not only from MRI, but also from task fMRI and resting state fMRI to build a manual parcellation of the cerebral cortex. Using this manual parcellation, they train a multi-layer perceptron on features extracted from the multi-modal input data. In order to simplify the classification problem and to enforce topological correctness, the authors train one binary classifier for each area and only apply it on the region of the brain that contains the area after a registration from the manual parcellation (*spotlight classification*). This restriction on the regions where a classifier is applied constitutes an implicit encoding of location information in the classification process.

[2]Reprinted from Wachinger et al. (2017), Copyright (2017), with permission from Elsevier.

The presented automatic MRI segmentation approaches use several concepts (i.e., context information, location information, post-processing with CRF) to get topologically correct, spatially consistent and smooth segmentations. Although the above mentioned CNN models are not directly applicable to histological area segmentation due to the different image resolution, dimensionality and imaging modality, spatial consistency and topological correctness are also important in the context of histological area parcellation.

Context information[3] is used by several approaches presented in this thesis, e.g., the profile classification approach (block of profiles) and the deep label propagation network. Directly including location information in the brain area segmentation models is not desirable, because that would require a 3D reconstruction of the 2D sections. This reconstruction is not always available and is time-consuming and error-prone to compute. In addition, often not all sections of one brain volume are available which makes registration and robust location information difficult. However, similar to the spotlight classification of Glasser et al., 2016, the two stream network introduced in Chapter 5 indirectly includes location information in the model by adding probabilistic maps of which areas are likely to occur at the location of the input patch. These probabilistic maps can be seen as a soft constraint for the topology of brain areas. Using a graphical model (CRF) to enforce spatially consistent and smooth segmentations is a popular approach. However, a CRF is applied as a post-processing step to the predictions of a model which still requires good initial segmentations from the CNN model. The focus of this thesis is on developing a model to create these good initial segmentations. Applying CRF-based methods to these segmentations is left to future work.

3.3 Automatic analysis of high-resolution histological images

For precise identification of cytoarchitectonic brain areas it is necessary to analyze cell-body stained histological images. This modality has distinctive characteristics that distinguish it from natural images (Komura and Ishikawa, 2018):

- large image sizes, e.g., an average human brain section imaged with a light microscope has a size of 75 000×100 000 pixels at 1 µm resolution (7 GB with 8 bit precision),

[3]The segmentation network architectures presented in this thesis intrinsically use context information in the sense that a pixel classification depends on neighboring inputs (the receptive field). The context information discussed here refers to information provided by additional inputs that goes beyond the small context that the CNN sees.

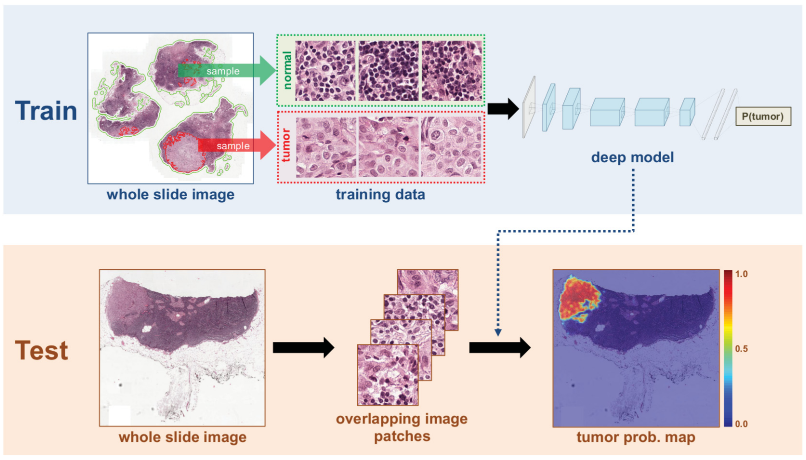


Figure 3.3: Patch-based approach for cancer classification in histological sections (D. Wang et al., 2016).[4]

- natural restriction of the amount of available training data, as annotations are very time-consuming and require trained experts,
- staining inhomogeneities due to the preparation of the tissue or the staining protocol, and
- texture-like structure of brain areas due to the repetitive pattern of neurons which distinguishes the areas.

Approaches for deep learning with a small amount of labeled training data are discussed in Section 3.4, including the typically used data augmentation techniques for histological images. Related work on analysis of histological sections addressing the remaining challenges (large image sizes, staining inhomogeneities, and texture-like structure of objects) is introduced in this section. First, patch-based solutions dealing with large image sizes are discussed in Section 3.3.1. Then Section 3.3.2 introduces approaches addressing staining inhomogeneities through stain normalization. Finally, CNNs that explicitly encode texture representations are introduced in Section 3.3.2.

3.3.1 Patch-based approaches

Most approaches analyzing histological images are motivated by pathological classification, detection and segmentation tasks in a computer aided diagnosis setting. A typical example for such a task is the identification of different tissue types or the segmentation of cancerous cells (Komura and Ishikawa, 2018; Madabhushi and G. Lee, 2016). For these tasks the object of interest covers a relatively small region of the histological section. Thus most approaches subdivide the very large input image

[4]Image from D. Wang et al. (2016).

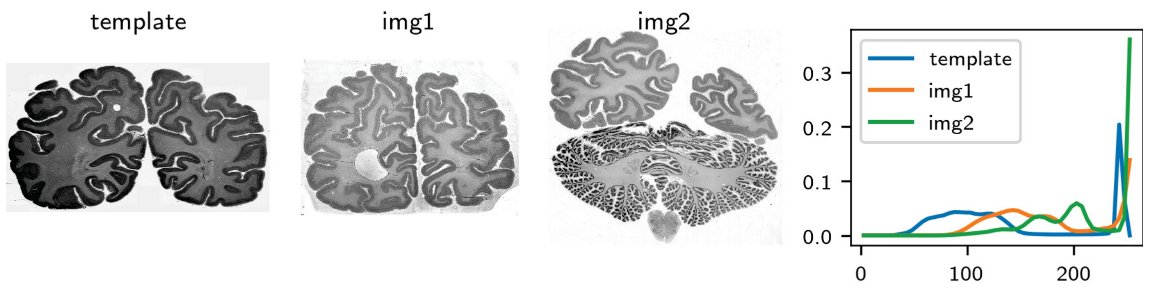
in small (typically 20 × 20 to 200 × 200 pixel) image patches and train a neural network to classify or segment the individual patches. Patch level decisions are then aggregated to form coarse segmentation maps (Kainz et al., 2017; Y. Liu et al., 2017; D. Wang et al., 2016; Xu et al., 2017). Figure 3.3 shows how the patch-based training and prediction works. During training, the image patches are extracted randomly from the training sections. For the prediction of a test section, the image patches that are extracted and predicted lie on a grid, ensuring that every pixel is predicted.

For the task of histological brain area parcellation the object of interest (the brain area) covers a much larger portion of the section. A cytoarchitectonic brain area is a part of the 2 mm to 4 mm wide cortical ribbon. In order to identify a brain area, the field of view of the network needs to contain at least the width of the cortex. This results in minimal patch sizes of at least 2000 × 2000 pixels at 1 µm resolution. Thus, the inputs for brain area segmentation are over 100 times larger than the inputs considered in the above approaches. Therefore it was necessary to design a custom network architecture with a large receptive field. However, similar to the above approaches, the segmentation models introduced in this thesis also use a patch-based approach to make training and prediction on large histological brain sections feasible.

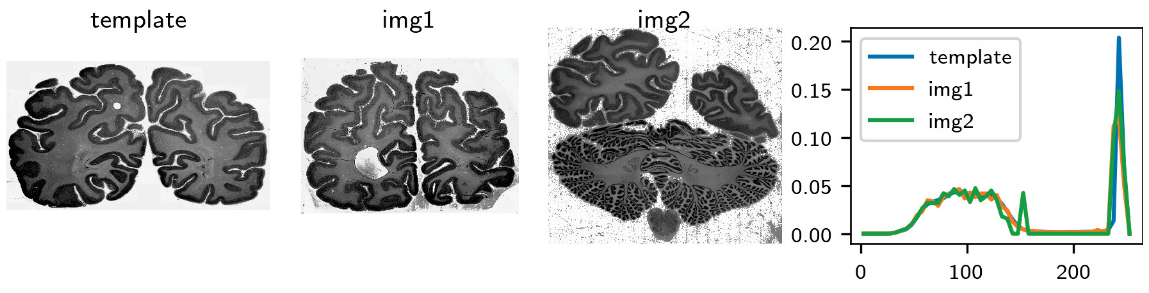
3.3.2 Stain-normalization methods

For brain area analysis, the histological sections are stained with silver staining (Merker, 1983) which visualizes neuronal cell bodies. This results in gray-value scans with the neurons appearing darker than the rest of the tissue. Due to the preparation of the histological sections (fixation, dewaxing, and drying), the staining can have variations in the same section and across sections. The microscope and camera used for imaging can introduce additional variations (Pichat et al., 2018).

In order to normalize these staining differences, techniques based on histogram matching are used (Gonzalez and Woods, 2002, p. 88) (see Figure 3.4). Malandain and Bardinet (2003) propose a method using histogram matching in the context for medical image processing. They calculate the continuous probability density function for two sections and compute the optimal affine transformation between them. In order to normalize a stack of consecutive sections, the histograms of all sections are matched to either a representative template image or an average distribution (Pichat et al., 2018). Recently, other stain-normalization and stain-transfer techniques based on deep generative models have been proposed (Bentaieb and Hamarneh, 2018; Zanjani et al., 2018). For example Zanjani et al. (2018) propose to normalize staining differences by training a variational autoencoder (VAE) to produce realistic images. The VAE is trained by playing against an adversarial discriminator that tries to distinguish the images created by the VAE from real images. The VAE learns la-



(a) Images and histograms before histogram matching



(b) Images and histograms after histogram matching

Figure 3.4: Example of histogram matching between tissue sections with different staining. Note that the staining differences inside the sections (e.g., img1 is darker at the bottom than the top) are not evened.

tent variables which control the staining properties of the created image and can be adapted to follow the staining distribution of a template image. This method has the advantage that it can learn to treat different tissue classes differently, which is not the case for simple histogram matching approaches (e.g., Malandain and Bardinet, 2003). Although effective for most applications, these normalization approaches are computationally expensive and require an extra preprocessing step during training and prediction and depend on the choice of template image. Especially when dealing with high-resolution data, this additional time and memory constraint cannot be neglected.

As an alternative to stain normalization, data augmentation methods that randomly change the gray-value distribution can be used on the input patches. This results in models that are invariant to different gray-value distributions in the input patches. Y. Liu et al. (2017) reported that color normalizing images did not yield better performance than using color augmentation. Since the augmentation of an image patch can be done on-the-fly while assembling the batch, this method is computationally less expensive. For this reason, the segmentation models that are trained directly on the histological images (Chapters 5 and 6) use gray-value augmentation instead of stain normalization. For intensity profiles sampled across cortical traverses, however, augmenting the sampled profile has not the same effect as normalizing or augmenting the underlying image. Thus, for the profile classification model from Chapter 4 the images are normalized by their mean and standard deviation before extracting the profiles. The 8 times smaller resolution of the images for this task (16 µm compared to 2 µm for the model in Chapter 5) reduces the additional time and memory needed for this step.

3.3.3 Texture representations

Brain areas are composed of neurons that are arranged in a certain repetitive pattern. This texture-like structure distinguishes them from typical objects in natural images. Thus, segmentation of brain areas in histological images has properties in common with texture classification tasks. Texture classification has long been studied in computer vision (see Tuceryan and Jain (1993) for an overview), using a variety of statistical, geometrical, and model-based methods. Due to the use of convolutional and pooling layers, CNNs learn translation invariant features and are well-suited to recognize textures (L. Liu et al., 2018).

Recently, CNN architectures for fine-grained visual recognition and texture classification have been proposed (Lin et al., 2015; C. Wang et al., 2017). The bilinear CNN (Lin et al., 2015) generalizes traditional orderless feature descriptors and models local pairwise interactions between learned features. It consists of two convolutional streams f_A and f_B (that can be the same or different) that extract features $f_A(X)$ and $f_B(X)$ from the input image X. The features are then combined to a bilinear feature by computing their outer product. C. Wang et al. (2017) use a bilinear CNN on histological images to classify image patches in eight tissue types and report better performance than a standard CNN. A drawback of bilinear CNNs is that the bilinear feature representation is very high dimensional. Gao et al. (2016) propose compact bilinear pooling that reduces the dimensionality of the bilinear features. Addressing these recent approaches was out of the scope for this work, but in the future it could be evaluated whether these bilinear feature representations are also applicable to very large patches of histological images.

3.4 Deep learning with limited training data

One of the central challenges for applying deep learning to brain area segmentation is the limited amount of training data. Besides manually annotating more training data or reducing the complexity of the model, there are two complementary approaches for this problem:

1. automatically increase the training data set (data augmentation), and
2. use transfer learning to leverage features learned on an auxiliary task.

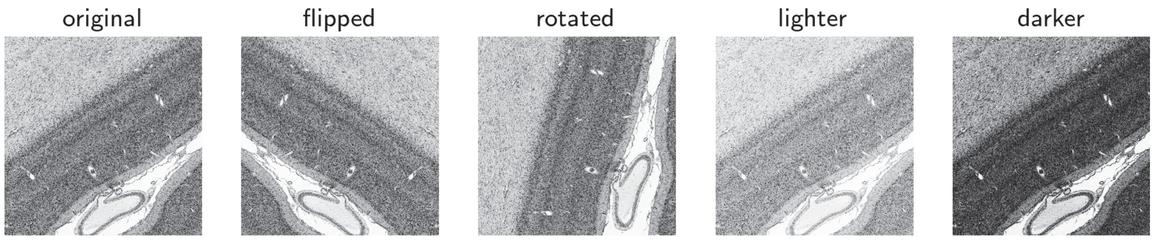


Figure 3.5: Commonly used data augmentation methods applied to a patch extracted from a histological brain section.

The most commonly used method to increase the size of the training dataset is to use data augmentation by applying geometrical and color transformations to the input patches and adding them to the training dataset. Related works using data augmentation are presented in Section 3.4.1. In transfer learning, models are initialized from weights learned on an auxiliary task. The auxiliary tasks can be general image classification tasks on natural images. Approaches using transfer learning for medical imaging tasks are described in Section 3.4.2. An alternative to using representations learned on natural images is to use unsupervised learning on histological images to generate good initial weights for supervised models. Section 3.4.3 considers several unsupervised tasks for learning representations on histological images.

3.4.1 Data augmentation: Generate more training data

When training with a limited number of training samples, it is useful to enhance the training dataset by using data augmentation (e.g., Kamnitsas et al., 2017; Y. Liu et al., 2017; Pereira et al., 2016; Ronneberger et al., 2015). Data augmentation increases the size of the training dataset $\mathcal{X}$ by modifying each element $X \in \mathcal{X}$ with a function $\Psi : \mathcal{X} \to \mathcal{X}_{\text{aug}}$. A model which was trained with an augmented dataset is invariant to the variation introduced by the augmentation function and thus more generalizable. In addition, data augmentation decreases the risk that the model learns to classify a class based on irrelevant variation in the data. The type of the augmentation function depends on the type of invariance that should be modeled. Commonly used augmentation functions include affine transformations (e.g., rotations) and elastic deformations of the input images and adapting the gray-value intensities or color channels of each input image (Ronneberger et al., 2015). Figure 3.5 shows examples of commonly used data augmentation functions applied to a histological image patch.

Y. Liu et al., 2017 train a classifier to detect cancerous cells on histological sections and augment their training dataset with rotations by multiples of 90 degrees and left-right flips of the input patches. To increase the robustness to color variations, they perturb color values by changing brightness and hue. Similar to their work, the patch segmentation and classification CNNs introduced in this thesis (Chapters 5 and 6) leverage data augmentation to train more robust models by random rotations, flips and gray-value changes.

3.4.2 Transfer learning: Leverage already trained models

As introduced in Section 2.2.6, transfer learning allows to train large architectures on small datasets by initializing the model with general-purpose features that were learned on an auxiliary task. Several studies have shown that the transferability of features decreases as the difference between the base task and the target task increases (Azizpour et al., 2015; Mahajan et al., 2018; Yosinski et al., 2014). Despite the differences between natural images and medical images, there are several successful examples where fine-tuned CNN models outperformed CNN models trained from scratch and other handcrafted approaches. Tajbakhsh et al. (2016) analyze transfer learning from AlexNet (Krizhevsky et al., 2012) which was trained on the ImageNet dataset containing over 14 million natural images in over 1000 categories (Deng et al., 2009), to four different medical applications. They come to the conclusion that for tasks that deal with images that are similar to natural images, only the last few layers of the model need to the fine-tuned, whereas for other applications fine-tuning of all layers was necessary to get optimal performance.

Pre-training models on the ImageNet dataset was found to be useful for lymph node detection and lung disease classification (Shin et al., 2016), and for mammogram analysis (Carneiro et al., 2015). The most successful application of transfer learning in medical imaging as of date is a model that was pre-trained on the ImageNet dataset and which is able to classify skin lesions at better-than-human precision (Esteva et al., 2017). Fine-tuning models trained on ImageNet for tissue classification and segmentation tasks on histological images was shown to be advantageous as well (Bayramoglu and Heikkilä, 2016; Mormont et al., 2018; Xu et al., 2017).

The previously introduced approaches deal with classification or detection problems. However, the subject of this thesis is a semantic segmentation task. Although architectures designed for classification can be adapted for segmentation by replacing fully-connected layers with convolutional layers (Long et al., 2015), specialized architectures like the U-Net (Section 2.2.3) are better suited to efficiently compute high-resolution output segmentations. Unfortunately, pre-trained models for segmentation tasks are not yet widely available. Additionally, for high-resolution image segmentation, a receptive field of at least 2000×2000 pixels at 1 µm resolution is needed (Spitzer, Amunts, et al., 2017). None of the existing pre-trained architectures fulfill this requirement, as images from ImageNet are only 225×225 pixels in size. Thus, for our application it is necessary to train a custom architecture from scratch. However, in order to deal with limited training data, it is possible to pre-train the custom architectures using an unsupervised or self-supervised task (see Section 3.4.3).

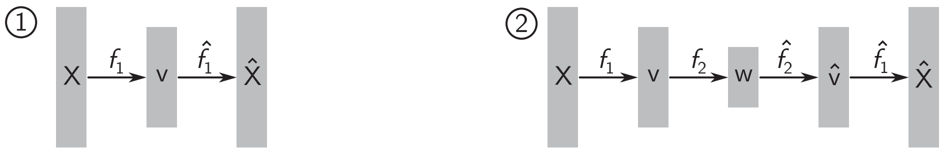


Figure 3.6: Two-layer stacked autoencoder. (1) A stacked autoencoder is trained by first training the outer encoder-decoder layers f_1 and $\hat{f}_1$ to reconstruct inputs X. (2) Then, the inner encoder-decoder pair f_2 and $\hat{f}_2$ is trained to reconstruct the latent feature representation $v = f_1(X)$ of the encoder f_1.

3.4.3 Unsupervised pre-training: Leverage unlabeled data

Like supervised auxiliary tasks, unsupervised learning can be used to initialize or pre-train networks for supervised tasks where not much labeled training data is available. The advantage of using unsupervised learning tasks for pre-training is that they can be trained on the same domain as the target task (e.g. histological images), and can thus learn features that are already adapted to the domain. Unsupervised representation learning has a long history (Bengio et al., 2013). In the following the focus is on two recent learning tasks that can be used for pre-training supervised models: autoencoders and self-supervised learning.

Autoencoders

An autoencoder consists of an encoder f that maps input images X to a latent feature representation $f(X)$ and a decoder $\hat{f}$ that reconstructs the input from the feature representation: $\hat{f}(f(X)) = \hat{X}$. Autoencoders are trained to minimize the reconstruction error between the input and the output (Olshausen and Field, 1996). In their most basic form, encoder and decoder consist of one layer each. To encode more complex features, autoencoders can be stacked and trained layer by layer (Hinton et al., 2006). Figure 3.6 visualizes a two-layer stacked autoencoder. Vincent et al. (2010) proposed stacked denoising autoencoders that learn to reconstruct the original image from a noisy input. Using noisy inputs forces the autoencoder to learn more robust features. This allows to use denoising autoencoders as a layer-wise pre-training strategy for supervised tasks (Vincent et al., 2010).

Unfortunately, as of now there is no example of an autoencoder that was trained successfully on very large images (e.g., 2000×2000 pixels that are necessary for brain area classification). Additionally, for area classification on histological images, very fine-grained patterns in the arrangement of neurons and not the overall structure of the image are important for classification. When training with a reconstruction loss, the model will first learn the general structure of the image, and not the fine-grained patterns. Since for the task at hand the details are essential, it remains unclear what objective to use when training an autoencoder. Therefore this thesis does not use autoencoders for pre-training supervised models.

Self-supervised learning

Self-supervised learning is a recent variation of unsupervised learning. It exploits labels that are easily accessible or obtainable from the inherent structure of the data. The motivation for self-supervised learning is that in order to solve the self-supervised task, the model has to semantically understand the input data and learn general purpose image features that can be used for supervised tasks.

Several approaches use external labels that are easily accessible with the data. Agrawal et al. (2015) and Jayaraman and Grauman (2015) use egomotion data provided by sensors to train a Siamese network to predict the camera transformation between a pair of images. They show that egomotion is a useful supervisory signal for learning feature representations. Owens et al. (2016) use ambient sound as labels and train a model to predict a statistical summary of the sound associated with a video frame. A second line of approaches uses structure contained in the data itself to design the self-supervised labels. X. Wang and Gupta (2015) used temporal data of videos to generate the training labels: Extracting two matching frames using an unsupervised tracking method and an arbitrary frame, they train a triplet network model to create an embedding that is close for the two matching patches and far for the non matching frame. H.-Y. Lee et al. (2017) also leverage the temporal coherence in videos and design a self-supervised task based on sorting sequences.

Considering single images, Dosovitskiy et al. (2016) trained a model to be invariant to geometric and photometric transformations. Recently, Gidaris et al. (2018) showed that predicting image rotations is also a viable self-supervised task for natural images. Larsson et al. (2016) and Zhang et al. (2016) use color information in the input images to train CNNs for colorization of gray scale images. Pathak et al. (2016) propose a context encoder, an Autoencoder like structure that is trained with an adversarial loss to generate missing parts of an image. They show the applicability of the learned features for CNN pre-training on classification, detection and segmentation tasks.

Doersch, Gupta, et al. (2015) designed a task based on spatial context prediction from a single image (Figure 3.7). Given two patches sampled from one image, their Siamese network learns to predict their spatial configuration. Solving this task requires the model to understand scenes, objects and object parts. Extending this idea, Noroozi and Favaro (2016) train a network to solve a jigsaw puzzle. The network gets as input nine image tiles and has to predict their ordering. They show that the learned features are suitable for image retrieval. Fine-tuning their model on segmentation, detection and classification tasks they significantly outperformed existing methods on the Pascal VOC dataset containing 20 000 images with 20 object

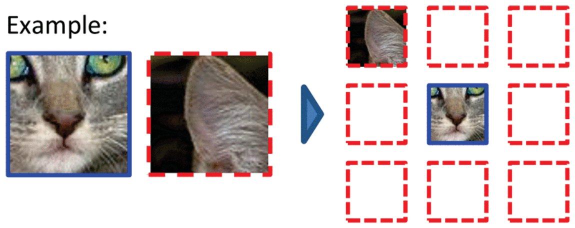


Figure 3.7: Self-supervised spatial context prediction task proposed by Doersch, Gupta, et al. (2015).[5] Given two input patches sampled from the same image, the model should learn to predict their relation to each other in the image.

categories (Everingham et al., 2010) without needing additional training data from larger datasets such as ImageNet. Recently, Doersch and Zisserman (2017) showed that when combining several self-supervised tasks, one can improve the performance even further.

The self-supervised approach combines the advantages of supervised and unsupervised training: The training is still supervised, meaning that we can take advantage of the well-understood supervised training algorithms, while lots of training data is available. The approaches introduced above show that with the right self-supervised tasks, the model has to understand the input data in a way that is useful for further supervised learning. Building on the research in self-supervised learning, Chapter 6 introduces a self-supervised task exploiting the inherent 3D structure of the histological sections.

[5]Figure adapted from Doersch, Gupta, et al. (2015). Copyright 2015 IEEE.

4 Area classification using profile features

The current state-of-the-art method for cytoarchitectonic mapping is the observer-independent method developed by Schleicher, Amunts, et al. (1999) (Section 2.1.3). It is based on the comparison of features extracted from gray-value intensities (*profiles*) beneath cortical traverses (*streamlines*) from the pial surface to the gray/white matter border. These profiles traverse the cortical laminae and thus contain information about the composition and thickness of each lamina. Since laminar variations are crucial factors to determine the cortical area, profiles represent relevant features for automatic area classification.

Motivated by the success of profile features in the observer-independent method, this chapter investigates whether these profile features can be used for automatic area classification. Saliency analysis of the resultant model shows which cortical laminae are most discriminative for the prediction of certain areas. Although the performance of the classifier is not satisfactory on 2D sections, the advantages of using profiles in the analysis of 3D volumes is discussed.

The contents of this chapter are the result of a collaboration between Konrad Wagstyl (University of Cambridge) and Hannah Spitzer, where Konrad Wagstyl contributed expertise in extracting and using profile features and Hannah Spitzer expertise of deep learning for cortical areas. The main part of the work was done during a stay at the ACELab under the direction of Prof. Alan Evans at the Montréal Neurological Institute (MNI), McGill, Montréal. The automated streamline extraction pipeline, the CNN architecture and hyper-parameters were developed together. For the present thesis, the models were trained on profiles extracted from the `B01Collection` to allow comparison to models trained in later chapters.

In the following, Section 4.1 describes the automatic extraction of streamlines and profiles from the `B01Collection`. The CNN architecture that includes additional spatial context in the training process is introduced in Section 4.2. Section 4.3 shows the qualitative and quantitative performance of the model for classifying a subset of visual areas and analyzes salient profile features. Finally, Section 4.4 discusses the results and potential applications for profiles in the automated analysis of 3D volumes. In Section 5.3.4 this approach is compared to a CNN model using the high-resolution images directly as input without a separate feature extraction step.

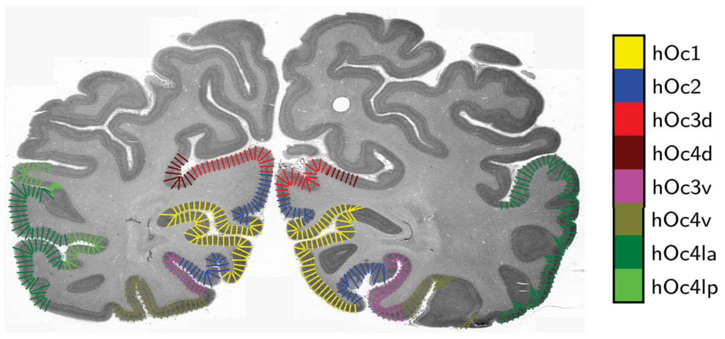


Figure 4.1: Selection of streamlines extracted from section 1141 of B01 colored by their area label. Streamlines are straight lines from pial surface to gray/white matter border and were sampled with a distance of 16 µm. Streamlines could not be extracted in very oblique parts of the cortex.

4.1 Automatic profile extraction workflow

The aim of this chapter is to develop an automatic pipeline for profile feature extraction and area learning. In total, 960 000 profiles were automatically extracted from annotated regions of the `B01Collection` with a similar procedure to the one described in Schleicher, Amunts, et al. (1999). Since the method of Schleicher, Amunts, et al. (1999) was not developed for automatic high-throughput processing of histological sections and requires manual delineations of the inner and outer cortical boundary, the workflow described in the following deviates in several details from the original profile-extraction procedure. Section 4.1.1 describes how the streamlines, i.e., the traverses of the cortex, are defined, and Section 4.1.2 explains the generation of profiles from original and smoothed versions of the histological sections.

4.1.1 Streamlines: Traverses of the cortex

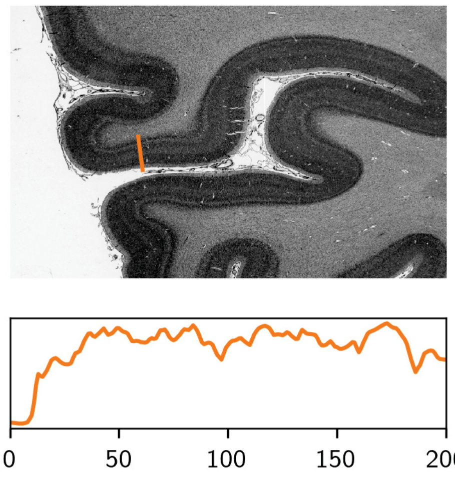
To extract the streamlines, Schleicher, Amunts, et al. (1999) use precise, hand-drawn delineations of the inner (between lamina IV and white matter) and outer (between lamina I and II) cortical boundary. In contrast, here approximate delineations based on 20 µm resolution are used. The advantage is that these delineations are fast to annotate or could be provided by an automatic contour finding method. For the present work, the delineations were made manually. Corresponding points between the inner and outer cortical boundary are found by following the gradient of a potential field between the boundaries calculated with Laplace's equation (similar to the electric field between two charged surfaces) (Jones et al., 2000). In very oblique parts of the cortex, this method cannot find corresponding points due to low gradients in the potential field. However, since most oblique parts of the cortex do not offer sufficient information to recognize cortical areas (Amunts, Lepage, et al., 2013), the missed profiles would not have contributed towards learning to discriminate cortical areas.

Like in Schleicher, Amunts, et al. (1999), these points are used to calculate a midline through the cortex. Starting from this midline, straight equidistant streamlines with a distance of 16 µm are calculated and expanded by 50 µm on both ends to make sure that the whole cortex is covered (owing to the potentially imprecise delineations of inner and outer cortical boundary). Due to the inherent translational invariance of the convolutional layers in the neural network, the expansion will not have an adverse effect on the performance. Figure 4.1 shows example streamlines extracted from one section.

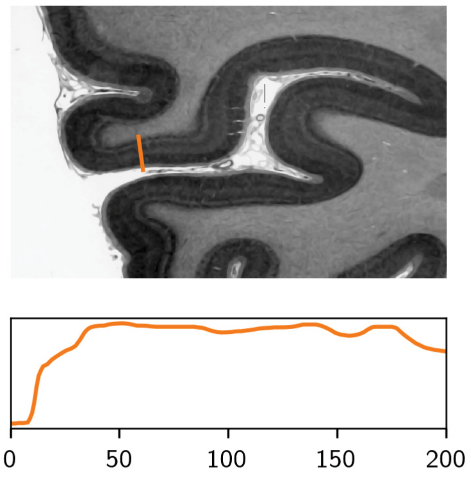
4.1.2 Profiles: Gray values beneath the streamlines

Profiles $p \in \mathbb{R}^{200}$ are extracted by sampling 200 points along each streamline from the pial surface to the gray/white matter border using downsampled 16 µm resolution versions of the histological sections. This is the resolution that Schleicher, Amunts, et al. (1999) use to extract their profile features. In addition to sampling profiles from the original section, profiles are extracted from an anisotropically smoothed version of the histological sections, resulting in a *raw profile* p_{raw} and a *smooth profile* p_{smooth} for each streamline. The images were smoothed using geometric heat equation and nonlinear diffusion as proposed by Kimia and Siddiqi (1996) and previously used in Wagstyl, Lepage, et al. (2018) for creating smoothed profiles. This method smooths images by smoothing each iso-intensity level set by curvature deformation. Shapes with higher curvature are smoothed faster, resulting in the removal of small-scale intensity changes, while preserving larger scale changes. In contrast to isotropic smoothing, the anisotropic smoothing maximally smooths in the direction tangential to the cortical layers. The smoothing preserves intra-laminar intensity differences, but removes individual cell bodies, small blood vessels, and histological artifacts from the image. Figure 4.2 compares raw and smoothed images and resulting profiles. Raw profiles capture local cellular structure, while smooth profiles capture broader, layer-specific features.

The profiles used by Schleicher, Amunts, et al. (1999) are sampled from gray-level index (GLI) images, which are calculated from 1 µm tissue sections using a cell segmentation. In contrast, the profiles used in the automatic workflow described here are sampled from downsampled tissue sections. This avoids the manual tuning of the cell-segmentation step which needs to be done for each region of interest in the method of Schleicher, Amunts, et al. (1999). Figure 4.3 compares a profile extracted from a GLI image to a profile extracted from a downsampled gray-value histological section at the same position. While the GLI profiles show more detail, the general peaks and troughs can be seen in the profile extracted from the downsampled section



(a) Raw image and resulting profile



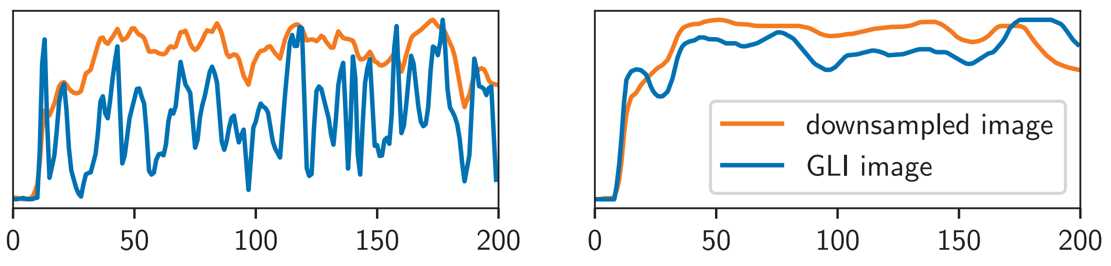
(b) Smoothed image and resulting profile

Figure 4.2: Comparison of raw and smooth profiles sampled along a streamline. Smooth profiles are extracted from anisotropically smoothed versions of the high-resolution image. Raw profiles capture local cellular information and smooth profiles broader, layer-specific features.

as well. Thus, it can be expected that profiles extracted from downsampled sections can be used to identify cortical laminae as well.

The profiles p were normalized according to the mean μ and standard deviation σ of cortex and white matter: $\hat{p} = (p - \mu)/\sigma$. Statistics were calculated on a per-section-basis using an automatically calculated tissue mask (see Section 2.3.4) to account for staining differences between individual sections. As this is a linear transformation, it does not affect the intensity relationships between profiles from a single image. Figure 4.4 shows raw and smooth profiles sampled from two sections before and after normalization. After normalization, the profiles have the same value range.

A trade-off of this automatic profile extraction pipeline is that this dataset will contain profiles from highly curved and oblique regions of the cortex. In oblique regions, the laminae are not cut perpendicularly, which can let some laminae appear thicker and other thinner. In regions with high curvature the relative thickness of the laminae changes. According to the accepted equivolumetric model of Bok (1929), each lamina has the same volume in straight and curved parts of the cortex. This means that depending on the curvature, a lamina will appear thicker or thinner on the cross-section. A 3D volume is necessary to calculate cortical depths with the equivolumetric model; thus a normalization of the profiles extracted from 2D sections to follow the equivolumetric model is not possible. Due to these problems, the observer-independent border detection method is usually only applied to straight,



(a) Raw profiles **(b)** Smooth profiles

Figure 4.3: Comparison of profiles extracted from GLI and downsampled cell-body stained images. The smoothed profiles from both data sources have similar peaks and troughs and thus both contain information about the cortical laminae.

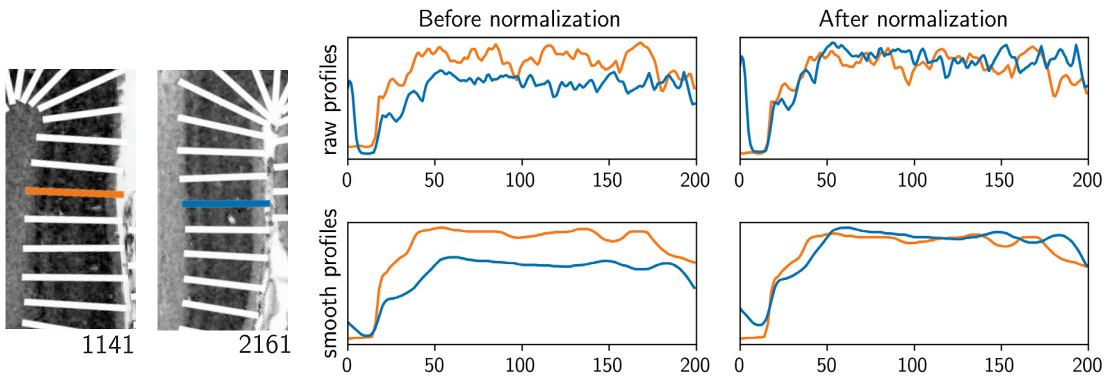


Figure 4.4: Effect of normalization on raw and smooth profiles from area `hOc1` sampled from different sections. After normalization, the intensity of the profiles is more similar.

non-oblique parts of the cortex to ensure that the cortical laminae are not distorted. In the following, we will evaluate whether it is possible for an automatic method to learn to classify areas using these highly variable profiles.

4.2 1D convolutional neural network for block-wise profile classification

Based on the extracted profiles, a CNN model for classifying single profiles or blocks of profiles in different cortical areas is trained. Section 4.2.1 introduces the network architecture used for this task and Section 4.2.2 describes the training procedure.

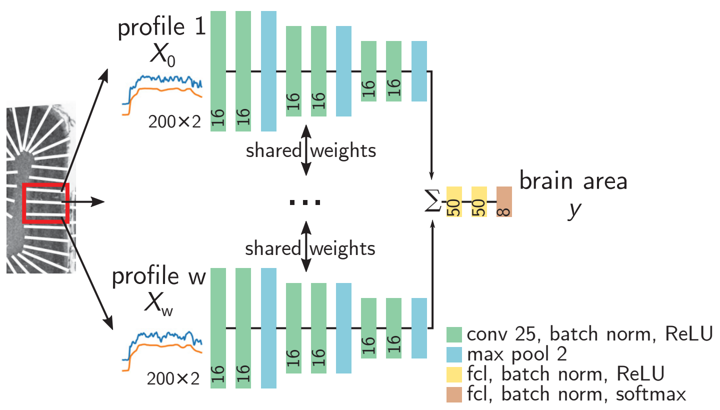


Figure 4.5: The ProfileNetwork classifies profiles in different brain areas. The network takes a block of raw and smooth profiles as input, and processes them using 1D convolutional layers. The mean of the resulting activations is then processed in fully connected layers.

4.2.1 Network architecture

The network architecture is shown in Figure 4.5. The network uses 1D convolutional layers with 16 kernels of size 25. Each convolution is followed by batch normalization and a ReLU nonlinearity. The network architecture contains three blocks of two convolutional layers and one max-pooling layer. The convolutional part of the model is followed by two dense layers with 50 nodes each and a softmax output layer.

Since a single profile can be noisy (due to large blood vessels, or tissue artifacts) and represents only a thin strip of tissue, training on a block of adjacent profiles is recommended to increase robustness. When training with a block of profiles, each profile is processed individually by the convolutional part of the network. After the last pooling layer, the mean of the activations for each profile is calculated and passed to the first dense layer. Thus, the network gets as input a block of adjacent raw and smooth profiles $X_i, ..., X_{i+w}$ with $(p_{\text{raw}}, p_{\text{smooth}}) = X \in \mathbb{R}^{2\times200}$ and block size w and predicts a brain area $\hat{y}$. This architecture contains 56 000 parameters and will be called ProfileNetwork in the following.

4.2.2 Training procedure

As described above, raw and smooth profiles are extracted from the B01Collection. Formally, let $\mathcal{X}_{\text{pr}} = \{X_1, ..., X_n \mid X_i \in \mathbb{R}^{2\times200}\}$ be the profile dataset with $n = 960\,000$ samples, consisting of raw profiles $X^{(0)} = p_{\text{raw}}$, and smooth profiles $X^{(1)} = p_{\text{smooth}}$. Each element X has a corresponding groundtruth brain area $y \in \mathcal{Y}_{\text{pr}}$. The dataset is divided in train/val and test splits on a per-section-basis according to the split of the B01Collection defined in Section 2.3.1. This results in 810 000 train/val profiles in $\mathcal{X}_{\text{pr}}^{\text{tr}} \cup \mathcal{X}_{\text{pr}}^{\text{val}}$ and 150 000 test profiles in $\mathcal{X}_{\text{pr}}^{\text{te}}$. Annotated regions on the train/val sections are further subdivided in train and validation sets, such that train and validation contain equal proportions of each annotated area and that validation contains 5% of the profiles in train/val. This ensures comparable train and validation splits, and an independent test set sampled from previously unseen sections.

The `ProfileNetwork` is trained on $\mathcal{X}_{\text{pr}}$ by minimizing the weighted cross-entropy loss between predicted area and groundtruth area (Equation (2.18)). In order to account for the imbalanced number of profiles for each area in the training set, the loss is balanced for each area with its inverse frequency (calculated on the validation set). The loss includes a weight decay with factor $\lambda = 0.002$.

The model was trained for 60 epochs with a batch size of 512. The initial learning rate of $\eta_0 = 0.001$ was decayed by a factor ν every epoch, with $\eta_i = \eta_0 \nu^i$ the learning rate after the i-th epoch. Since a model trained with a block size of w profiles effectively sees w times as much data in one epoch as a model with a block size of one, the learning rate was decayed faster for these models, with $\nu_w = \nu_1 / w$, ν_w the learning rate decay for a model with block size w, and $\nu_1 = 0.875$.

4.3 Results

First experiments on all areas in the dataset showed that there was a set of underrepresented areas that the model was not able to learn. These were either areas where some part of the profile extraction pipeline failed and that thus had only a small number of profiles representing them, or areas with high curvature and many oblique parts which the profiles cannot represent well (Section 4.1.2). Since these "difficult" areas resulted in noisy training gradients that made convergence very difficult, the following models were trained on the subset areas that the initial models could learn, and the remaining areas were summarized in a class `all`. The subset of visual areas that the following models are trained on is thus: `hOc1`, `hOc2`, `hOc4la`, `hOc4lp`, `FG3`, `FG4`, and `all`. Section 4.3.1 shows the performance of the `ProfileNetwork` trained with different window sizes on this subset of areas. Then, Section 4.3.2 evaluates which parts of the profiles the model focuses on to make its prediction.

4.3.1 Performance on the area classification task

To evaluate the effect of different block sizes, models with block size $w = 1$ (`profile-1`) and $w = 7$ (`profile-7`) were trained using the adapted learning rate decay to ensure that each model learns at the same speed. The `profile-1` model had a Dice score of 0.59, and the `profile-7` model a Dice score of 0.62 on the test set $\mathcal{X}_{\text{pr}}^{\text{te}}$. The detailed per-class performance is shown in Figure 4.6 with confusion matrices calculated on the test set. It shows that training with a block of 7 profiles slightly improves the

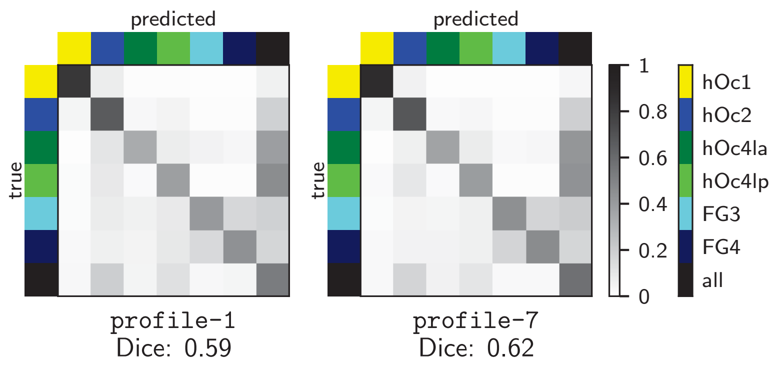
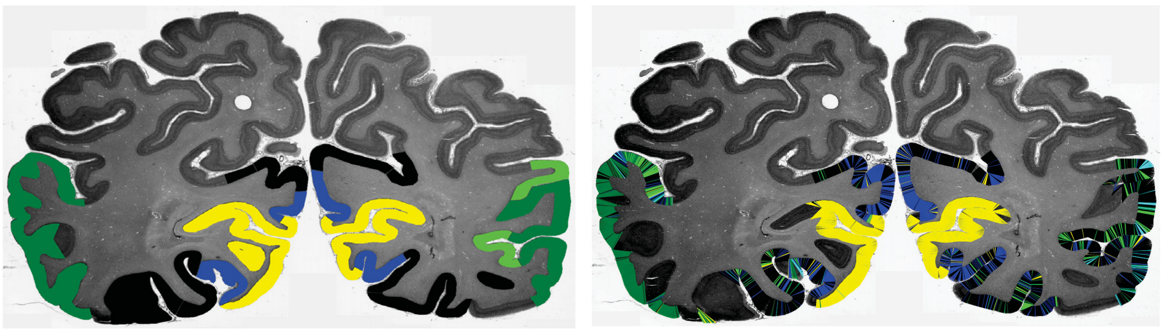


Figure 4.6: Confusion matrices for `profile-1` and `profile-7` trained with block sizes 1 and 7.



(a) Groundtruth annotation **(b)** Prediction of profiles

Figure 4.7: Qualitative results for `profile-7`. Legend in Figure 4.6.

performance for each area compared to training with only 1 profile. The single areas can be predicted well, but the aggregated label `all` is difficult for the model to learn as it is very diverse and contains per definition those areas that are difficult to learn. Figure 4.7 shows qualitative results of `profile-7` on a representative test section. Including context improves the performance of the model. Models with larger window sizes than 7 do not improve further, indicating that a context of 7 profiles, which corresponds to a window of approximately 100 µm, already holds sufficient information to classify the subset of “learnable" areas.

Since the automatically extracted profile dataset contains many noisy profiles due to artifacts, oblique sections, and imprecise starting and end points, several data augmentation techniques were tested. However, none of the following data augmentation methods improved the Dice score on the validation set $\mathcal{X}_{\text{pr}}^{\text{val}}$:

- Dropout of several input nodes, or of intermediate nodes to regularize the model.
- Adding random uniform or pink noise to the input profiles to model noise in the profiles due to artifacts or staining differences.

- Random clipping of the ends of the profiles (and subsequent resampling to 200 data points) to explicitly train the model on profiles with different start and end points.

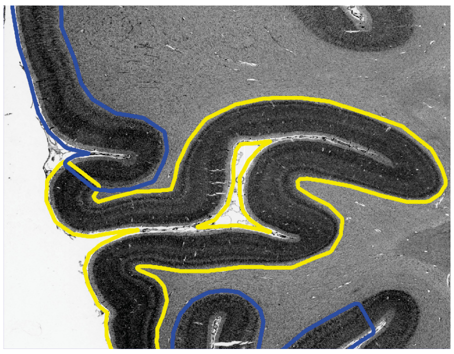
Due to the dense sampling of profiles from the high-resolution sections, all actually existing variations of profiles are already in the dataset, thus the augmentation with random noise is redundant. The normalization of histological sections before extracting profiles additionally removes some of the noise of the data. Since random clipping has no positive effect on the model, we must assume that the use of convolutional layers is already sufficient for the model to learn to detect the correct starting point of the profile.

4.3.2 Saliency analysis: What did the model learn?

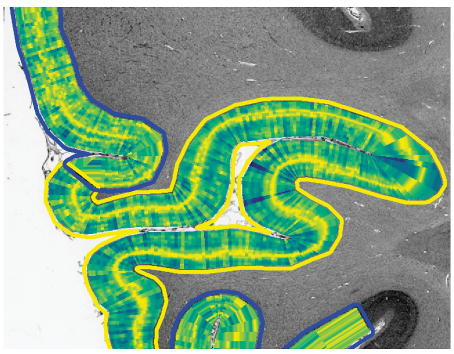
In the previous section, we have seen that the models perform reasonably well on a subset of areas. This section analyzes the `profile-7` model and asks: Why does the model make a certain prediction? What features in the input are most descriptive for this classification? First, to get an estimate of the extend that the `profile-7` model focuses on the raw and smooth profiles, the absolute sum of all gradients on the raw and on the smooth profile input was calculated over the validation set. 46% of the gradients go to the raw profile, 53% to the smoothed, indicating that both inputs are important.

Then, Grad-CAM was used to calculate the saliency of the profiles with respect to their predicted class (see Section 2.2.5 and Selvaraju et al., 2017). This provides a saliency map S for each profile which shows the location of high-level features whose presence is important for classifying the profile as the predicted class. The advantage of using profiles as inputs as opposed to high-resolution 2D image patches is that this map can be easily interpreted as highlighting one or several cortical laminae. This allows us to gain an understanding of what features (i.e., cortical laminae) the neural network uses to classify the areas. Since these features are similar to the features that neuroscientists use when identifying cortical areas, there are possibilities to compare neuroscientists to the automatic model and even learn from the neural network. Note that the calculated saliency maps are dependent on the set of areas that the model was trained on, because the model will learn different features for the same area if it has to distinguish it from a very similar area as opposed to a very different area. However, these analyses enable us to take a first step towards defining objective measures of which parts of the profile (at the resolution of cortical laminae) are important for the discrimination of specific cortical areas.

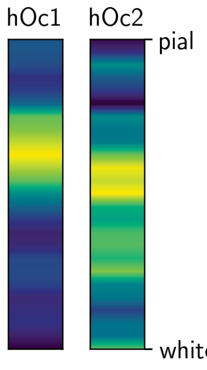
Analysis of the saliency for correctly classified `hOc1` profiles shows that `profile-7` mainly focuses on lamina IVc and the lower lamina IVb in order to classify a profile as `hOc1`. Plotting the saliency maps for each profile on the 2D histological sections



(a) Extract of section 1141 of B01 with areas hOc1 and hOc2



(b) Interpolated saliency maps



(c) Mean saliency

Figure 4.8: Saliency maps for profiles from hOc1 (yellow) and hOc2 (blue). For hOc1, lamina IVc is highlighted, showing that the model focuses on the distinctive laminar pattern of the primary visual cortex to classify hOc1. The mean saliency maps computed over the entire test set show differences between hOc1 and hOc2 saliency.

reveals that this focus on lamina IVc is consistent throughout the entirety of hOc1 (Figure 4.8). Focusing on lamina IVc and IVb is a sensible feature when distinguishing hOc1 from surrounding visual areas, because only the primary visual cortex has this very wide lamina IV which can be subdivided in the three sublaminae. The saliency for area hOc2 is more varied, however the focus is mostly on lamina IV and VI. This can also be seen in the mean saliency map for hOc2 (Figure 4.8c) which has two light regions, whereas the mean saliency map for hOc1 only has one light region.

4.4 Discussion

The previous experiments show that it is possible to classify some visual areas based on automatically extracted profiles from histological sections. The saliency analysis of `profile-7` shows that the model mainly focuses on lamina IVc to classify area hOc1. As the structure of lamina IV is a distinguishing feature of area hOc1 (Amunts, Malikovic, et al., 2000), this shows that the model correctly identified this feature as important to distinguish between hOc1 and other areas.

However, the model was only able to learn some areas, and failed in others, which disrupted the entire training process and made it necessary to exclude these “difficult" areas from training. This has to do with the quality of the profile dataset: as discussed in Section 4.1.2, including regions with high curvature, oblique sectioning, and artifacts results in a high variability of profiles for a single area that might make it too difficult to learn this area. The observer-independent method of Schleicher, Amunts, et al. (1999) is highly sensitive to slight surface misplacement and extreme care is taken by neuroscientists to ensure poor quality data is not included. With the automatic data-extraction approach, such care cannot be taken. This has a significant impact on what the network is able to learn. For a dataset with sufficiently good profiles, however it is possible to learn area classification as demonstrated in Section 4.3.1.

Although it is difficult to extract high-quality profiles from 2D sections, they can be extracted from reconstructed 3D histological volumes (e.g., the BigBrain) or from MRI volumes. In (reconstructed) 3D volumes, cutting artifacts and oblique sections are not an issue anymore. In addition, profiles can be normalized using the equivolumetric model of Bok (1929) (Waehnert et al., 2014), mitigating the effect of curvature on the profiles. For such volumes it is possible to automatically extract high-quality profiles and analyze salient profile features in a large scale and define objective measures of the profile features that are important for the discrimination of cytoarchitectonic areas. This idea should be further examined in future work.

Thus we can conclude that profiles are very useful to provide interpretable features in straight, non-oblique parts of the cortex, but their usefulness for automatic segmentation of cortical areas in entire histological sections seems limited. In addition, with the rise of deep learning methods, an end-to-end pipeline of automatic feature extraction and classification has shown to be superior to hand-crafted features in many different applications. Such an end-to-end approach for brain area segmentation is introduced in Chapter 5 and compared to the profile-based model from this chapter.

5 Area segmentation with two-stream convolutional neural network

In Chapter 4 hand-crafted features were used as input to the deep learning model. Although these features are well-established in the neuroscientific community for analyzing cytoarchitecture, we have seen that leveraging them in a completely automatic manner for brain area classification is problematic due to their failure to deal with imperfect samples of cortex. When using deep learning methods, features are learned automatically from the input images, potentially resulting in superior performance.

This chapter proposes an end-to-end CNN model that segments high-resolution brain sections into areas of the visual system, thus skipping the feature extraction step needed for the profile classification model. Enabling the model to learn features from the high-resolution brain sections as opposed to using profiles as input, gives it the advantage that, unlike the profile model (Section 4.4), it can learn to deal with slightly oblique tissue parts. This model combines prior knowledge about the topology of areas given by existing probabilistic atlases with precise local features extracted from the high-resolution cell-body stained images of brain tissue. It is trained with partial delineations of the visual system and produces high-resolution segmentations. It is invariant to rotations and gray-value changes of the input data and outperforms the `profile-`w model from Chapter 4.

The area segmentation model presented in this chapter was previously published in the conference paper "Parcellation of visual cortex on high-resolution histological brain sections using convolutional neural networks" (Spitzer, Amunts, et al., 2017). The architecture, and general training procedure introduced in the following are already described in this contribution. This chapter contains more evaluations than given in the conference paper, including the rotation and gray-value invariance shown in 5.3.1, and the feature map analysis in Section 5.3.3. Since the models presented in this chapter were trained on a larger dataset with a slightly different learning rate and label weighting scheme than the models presented in Spitzer, Amunts, et al. (2017), the reported scores differ.

In the following, Section 5.1 introduces the proposed CNN model. Section 5.2 discusses how to deal with partial groundtruth labels, introduces the data selection and

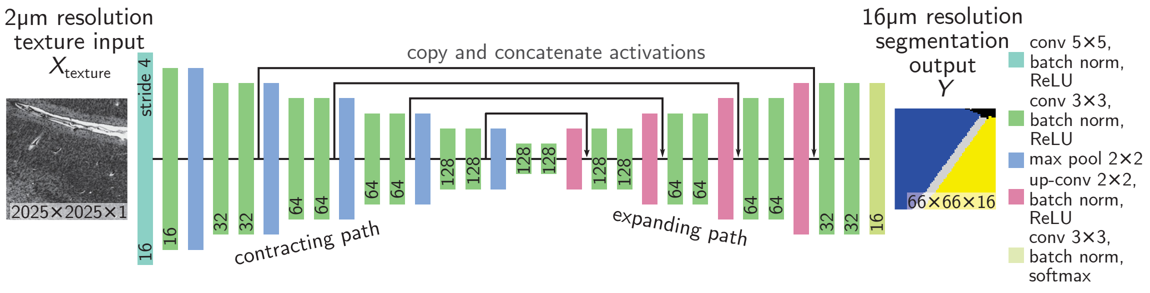

2μm resolution texture input
X_{texture}
2025×2025×1
stride 4
16
16
32
32
64
64
64
64
128
128
128
128
128
128
64
64
64
64
32
32
16
contracting path
expanding path
copy and concatenate activations
16μm resolution segmentation output
Y
66×66×16
conv 5×5, batch norm, ReLU
conv 3×3, batch norm, ReLU
max pool 2×2
up-conv 2×2, batch norm, ReLU
conv 3×3, batch norm, softmax

Figure 5.1: BaseNetwork for area segmentation. U-Net architecture with a receptive field of 1497 px. Produces 16 µm resolution semantic segmentations of a 2 µm resolution input image patch.

augmentation algorithms used for training the model, and examines computational challenges that arise during training. The performance of the model is evaluated in Section 5.3, by showing the influence of the atlas prior and the data augmentation techniques. Additionally, the activations of the model are analyzed at different depths of the model to gain an understanding of the types of features that the model has learned. Section 5.3.4 shows that this end-to-end segmentation model performs better on the area segmentation task than the model from Chapter 4, which was trained on profiles.

5.1 Two-stream U-Net for area segmentation

The CNN architecture for area segmentation is based on the U-Net (Ronneberger et al., 2015) which is introduced in Section 2.2.3. In the following, the changes to the U-Net architecture are described. First, the basic network architecture for segmenting high-resolution histological sections is introduced in Section 5.1.1. Then, Section 5.1.2 describes the extension of this architecture to a two-stream network combining image data with an atlas prior. The architectures presented in the following were published in Spitzer, Amunts, et al. (2017).

5.1.1 Basic network architecture

In order to distinguish cytoarchitectonic areas, the cortical laminae and columnar structures in the cortex are analyzed on 2 µm resolution image patches of cell-body stained sections. For successful classification of cortical pixels in cytoarchitectonic areas, all cortical laminae (i.e., the entire depth of the cortex) should be visible to the model when making a decision (Section 2.1.2). With an average cortical depth of 2 mm to 4 mm, and a 2 µm input resolution, the receptive field of the CNN should be about 1000 px to 2000 px. Thus, the receptive field of 186 px of the U-Net architecture needs to be increased by adding convolutional or pooling layers (Section 2.2.2).

In the presented architecture (Figure 5.1), the receptive field is increased to 1497 px by adding additional depth to the U-Net with an input convolutional layer with stride 4, an additional convolutional layer and a pooling layer. In order to reduce the memory footprint of the model, the number of convolutional kernels per layer are reduced compared to the original U-Net architecture.

The large stride of 4 in the first convolutional layer quickly reduces the input resolution. An alternative architecture to achieving the receptive field without a strided convolution is for example to add three blocks with two convolutional and one pooling layer before the U-Net. Such an architecture would allow the model to focus more on individual cells and their distribution at high resolution. However, training such an architecture is not feasible due to the large memory requirement resulting from the high-resolution feature maps of the first convolutional layers. In addition, a model trained on this alternative architecture (with batch size 4), did not perform better on the area segmentation task than the proposed architecture.

The architecture is designed in such a way that the right amount of context necessary to allow segmentation of areas is shown to the network. If the receptive field was larger, or if the architecture was to follow a multi-scale approach with additional low-resolution input as context, the added contextual information would allow the network to consider the local gyrification as a relevant characteristic for parcellating areas. However, cytoarchitectonic brain areas are not necessarily correlated with local gyrification patterns (Amunts, Lepage, et al., 2013; Fischl, Rajendran, et al., 2007). Additionally, the high inter-subject variability of the local folding patterns (Van Essen and Dierker, 2007) has the effect that local patterns appearing in one brain might not occur in others. Thus a larger receptive field, or a multi-scale approach is not desirable in our case, because it may lead to irrelevant features being learned.

In order to train a converging network, it was necessary to include a batch normalization layer after each convolutional layer and to use a relatively high learning rate. Section 5.2 gives more detail about the training process. In total, this fully convolutional model has 1.49×10^6 parameters. For an input $X_{\text{texture}} \in \mathbb{R}^{2025 \times 2025}$ at 2 µm resolution and a k-class segmentation task, the model outputs corresponding segmentation maps $Y \in \mathbb{R}^{66 \times 66 \times k}$ at 16 µm resolution. Note that although a resolution of 2 µm is required to see the relevant cytoarchitectonic features, the expected practical localization accuracy of the resulting cytoarchitectonic borders is much lower. Thus an output resolution of 16 µm is adequate. In the following, we call this architecture the `BaseNetwork`.

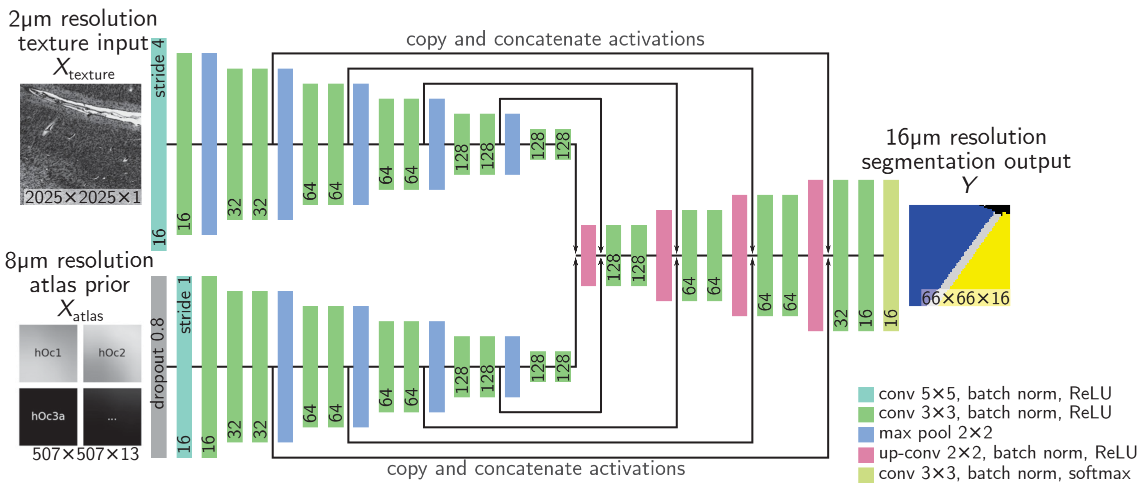

2µm resolution texture input
X_{texture}
2025×2025×1
8µm resolution atlas prior
X_{atlas}
hOc1
hOc2
hOc3a
...
507×507×13
stride 4
stride 1
dropout 0.8
16
32
64
128
copy and concatenate activations
16µm resolution segmentation output
Y
66×66×16
conv 5×5, batch norm, ReLU
conv 3×3, batch norm, ReLU
max pool 2×2
up-conv 2×2, batch norm, ReLU
conv 3×3, batch norm, softmax

Figure 5.2: `AtlasAwareNetwork` for area segmentation. Combines high-resolution (2 µm) texture information with prior information given by probabilistic maps of each area (8 µm).

5.1.2 Exploiting probabilistic atlas information

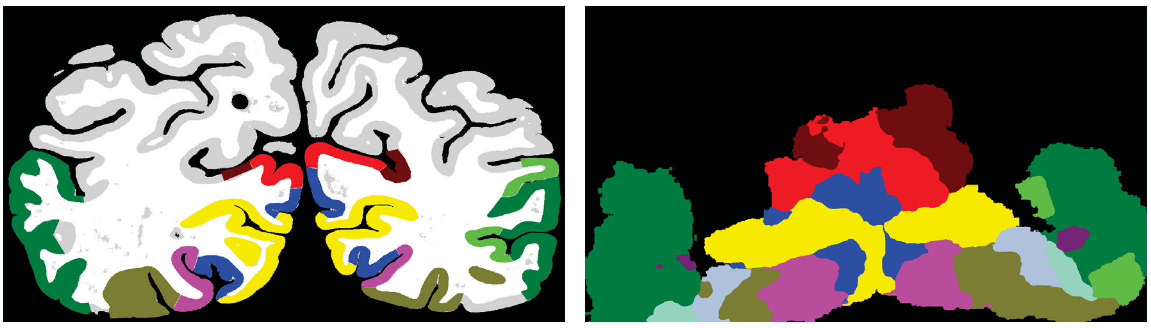
Inferring the cytoarchitectonic area from a $2 \times 2\,\text{mm}^2$ patch is a difficult task. When manually mapping human brain areas, neuroanatomists draw on their expert knowledge of the arrangement and topology of areas. First, they locate the point of interest in the brain volume on a low resolution global view. Already knowing which areas are likely to occur at the given section and how they usually look like, the neuroanatomist then considers the high-resolution histological section to identify the precise border between the areas.

Inspired by this process, the `BaseNetwork` is extended to combine high-resolution input patches from cell-body stained sections with a probabilistic atlas prior. For the atlas prior, the probabilistic JuBrain atlas is registered to the individual cell-body stained sections as described in Section 2.3.2. This results in a rough estimate of the brain areas on the individual sections as visualized in Figure 5.3. During the training, the extended network gets for each high-resolution texture input an additional input with corresponding projected probabilistic maps of each area. With this setup, the probabilistic atlas prior gives the model information about which areas are likely to occur at the location of the high-resolution texture input. Since the JuBrain atlas is created by aggregating delineations from 10 different brains this information constitutes a strong, general prior. However, due to the low-resolution registration and the inter-individual variability between brains, the prior alone is not enough to do a high-resolution segmentation (see Section 5.3.2). For precise borders, the arrangement of cells using the high-resolution texture input needs to be considered. Thus, the probabilistic maps provide a prior about the arrangement and topology of areas in general, and the high-resolution texture information is used to refine this prior based on the actual conditions.

This idea is implemented by the `AtlasAwareNetwork`, which is a `BaseNetwork` with a second contracting path, the atlas prior stream (Figure 5.2). The inputs to this stream are probabilistic maps (one for each area) with 8 µm resolution. The activations at the bottom of the two contracting paths are concatenated and jointly processed in the upsampling path. This enables the network to process each input type individually in the contracting paths and jointly in the expansive path. The hypothesis is that this atlas prior will help the model to disambiguate areas that have similar texture but are located in different regions of the brain.

The design of the atlas path and the point in the network where the two paths are concatenated is by no means the only possibility. In the proposed architecture, the atlas prior stream was built to mirror the texture stream. Considering that the atlas information consists of mainly uniform probabilities for each brain area, a smaller atlas prior stream with less convolutional layers might be sufficient to process the information before concatenating it to the texture path. Such a short atlas path might simplify the architecture, but should not change the performance much. Another design choice is the concatenation point of the two input streams. Preliminary experiments with an architecture that concatenates atlas and texture information shortly before the final segmentation layer resulted in worse performance, indicating that the model needs to process the two types of information together for several layers. However, an earlier concatenation point closer to the input layers than the end of the contracting path is also possible. This option was not explored in this thesis, but is an interesting premise for future work, as it increases the number of layers that process the texture and the atlas together and would allow the network to learn location specific filters in early layers.

The atlas data are less complex and directly represent an estimate of the output labels. Experiments have indicated that the network is able to learn features from the atlas prior faster than from the high-resolution image patch. This bears the danger that the model only bases its prediction on the atlas data input, without considering the texture patterns contained in the cell-body stained image. Although a local optimum, this is clearly not the best solution. This problem particularly occurred in architectures that were joining the atlas and cell-stained image patches earlier in the contractive paths. To overcome this challenge, noise was added to the atlas input (set every pixel to 0 with a chance of 20%) and an iterative training procedure was used: First, the model is trained on only the cell-body stained images and only after convergence training is continued using both cell-body stained image and atlas information. Technically, this is realized by first training the `BaseNetwork`, and then transferring the weights to the `AtlasAwareNetwork` and fine-tuning it. This procedure ensures that the model first learns to use the information contained in the



(a) Groundtruth combining partial manual delineations and automatic gray/white matter segmentation

(b) Maximum probability map of projected JuBrain areas thresholded at $p > 0.4$

Figure 5.3: Manual annotations and projected probability maps of brain areas of section 1141 of the B01. The projected probability maps are a coarse estimate of the actual brain areas.

cell-body stained image, and then adapts to include information gained from the atlas.

The final model contains 2.45×10^6 trainable parameters. It produces 16 µm resolution segmentations $Y \in \mathbb{R}^{66\times66\times k}$ for a 2 µm texture input $X_{\text{texture}} \in \mathbb{R}^{2025\times2025}$ and a 8 µm atlas prior input $X_{\text{atlas}} \in \mathbb{R}^{507\times507\times k}$. The atlas prior input in form of probabilistic maps provides a coarse prior for the localization and topology of cortical areas.

5.2 Data preparation and training

The models were trained on the `B01Collection` using all 13 visual areas annotated in the dataset. Section 5.2.1 describes how the groundtruth for the segmentation task was constructed. The training data set $\mathcal{X}^{\text{tr}}_{\text{B01}}$, consisting of randomly sampled patches from the `B01Collection`, is described in Section 5.2.2. The data augmentation techniques used for enhancing $\mathcal{X}^{\text{tr}}_{\text{B01}}$ are introduced in Section 5.2.3. Section 5.2.4 gives details about the loss function and training hyper-parameters. Finally, Section 5.2.5 discusses computational challenges posed by this task. Sections 5.2.1 to 5.2.4 contain information from Spitzer, Amunts, et al. (2017).

5.2.1 Divide complex background class

As discussed in Section 2.3.1, for the `B01Collection` only partial annotations are available. Thus, the background class for the segmentation task is inhomogeneous. It encompasses distinct concepts like white matter, unlabeled cortex, cerebellum, and background. Therefore, the background class is split in the classes “white matter", “unlabeled cortex", and “background" using the automatic gray/white matter segmentation of the `B01Collection` that was introduced in Section 2.3.4. The groundtruth for the area segmentation task contains 16 labels in total: The 13 visual areas annotated in the dataset (`hOc1`, `hOc2`, `hOc3d`, `hOc3v`, `hOc4d`, `hOc4v`, `hOc4la`, `hOc4lp`, `hOc5`, `FG1`, `FG2`, `FG3`, and `FG4`) and the labels `cor` (unlabeled cortex), `bg` (background), and `wm` (white matter). Splitting the background class has the advantage that during prediction the network will be performing gray/white matter segmentation and area segmentation in one pass. Figure 5.3a shows an example section containing the final groundtruth labels used for training.

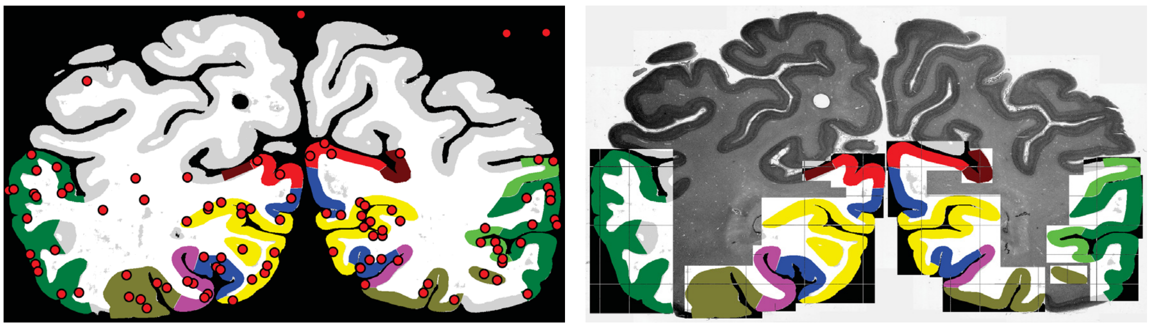
5.2.2 Training dataset generation

With the above considerations for the background class, we can now describe the construction of the training dataset $\mathcal{X}^{\text{tr}}_{\text{B01}}$ used for training the area segmentation models. A random location is selected from the train split of the `B01Collection`. To ensure that the majority of the patches that are shown to the model contain brain areas, 85% of the selected locations are inside labeled areas. The remainder is sampled from the classes `bg`, `wm`, and `cor`. Figure 5.4a shows 100 random sampling locations on one section selected following this algorithm. An image patch $X_{\text{texture}} \in \mathbb{R}^{2025\times2025}$ centered around the selected location is extracted from the 2 µm resolution cell-body stained image. Additionally, a 13 channel patch $X_{\text{atlas}} \in \mathbb{R}^{507\times507\times13}$ is selected from the 13 projected probabilistic maps for the labeled areas (Figure 5.3b). Each high-resolution image patch X_{texture} is normalized by the mean and standard deviation of gray-values in the `B01Collection`.

These image patches constitute one neural network input $(X_{\text{texture}}, X_{\text{atlas}}) = X \in \mathcal{X}^{\text{tr}}_{\text{B01}}$ (see Figure 5.2). Due to the random selection of the sampling locations, the training dataset $\mathcal{X}^{\text{tr}}_{\text{B01}}$ can be in theory quite large. It is limited by the number of pixels in the training sections, 1.3×10^{11}, although the number of non-overlapping patches from labeled areas, 6276, provides a more realistic estimate of the number of different training samples in $\mathcal{X}^{\text{tr}}_{\text{B01}}$. $\mathcal{X}^{\text{val}}_{\text{B01}}$ is created analogous to $\mathcal{X}^{\text{tr}}_{\text{B01}}$ and contains 2000 patches sampled from the validation split of the `B01Collection`. For each $X \in \mathcal{X}_{\text{B01}}$ a groundtruth output patch $Y \in \mathbb{R}^{66\times66\times16}$ is sampled from the combined groundtruth labels.

5.2.3 Data augmentation

The model should be robust to the local shape of the cortex (e.g. gyri, sulci) and learn to recognize cortical areas only by their texture as motivated in Section 1.1. To support this, each input patch $X \in \mathcal{X}^{\text{tr}}_{\text{B01}}$ is randomly rotated round the center of



(a) 100 random sampling locations

(b) Boxes around manual delineations that are evaluated during testing

Figure 5.4: Training and evaluation locations on section 1141 of the B01.

the patch by the random rotation function

$$\Psi_{\text{rot}}(X, \alpha)\,, \quad \alpha \sim U(0, 2\pi) \tag{5.1}$$

and flipped in left-right direction by the random flipping function

$$\Psi_{\text{flip}}(X)\,, \tag{5.2}$$

which flips the image with a 50% chance. To avoid missing pixels after the rotation, larger patches are sampled. For a desired input size of $s \times s$ pixels, a patch $\hat{X} \in \mathbb{R}^{\sqrt{2}s \times \sqrt{2}s}$ is sampled from the data. After data augmentation, the rotated image $\Psi_{\text{rot}}(\hat{X}, \alpha)$ is cropped to the desired input size resulting in the input image patch $X \in \mathbb{R}^{s \times s}$.

Due to staining inhomogeneities, local variations in gray values occur both inside single sections and across sections. To further augment the data, a variable gamma correction is applied to each input path $X \in \mathcal{X}^{\text{tr}}_{\text{B01}}$ by taking it to varying powers γ (*gamma augmentation*):

$$\Psi_{\text{gamma}}(X, \gamma) = X^{\odot\gamma}\,, \quad \gamma \sim U(0.5, 2) \tag{5.3}$$

with $^{\odot}$ denoting the element-wise (Hadamard) power.

When evaluating models trained with random rotations and mirrorings, the prediction for one input is calculated by predicting 8 versions of the input: the original input, and its rotations by 90, 180, and 270 degrees, and the mirrored version of the input and its rotations by 90, 180 and 270 degrees. The resulting segmentations are averaged (after reversing their transformation) to form the final prediction. Section 5.3.1 shows that training with these augmented images makes the model robust to rotations and staining inhomogeneities.[1] Other, more complex augmentations such as augmenting histological artifacts were outside the scope of this work, but might be helpful in the future to train more invariant models.

5.2.4 Training and evaluation

The model was trained with a weighted cross-entropy loss (2.18), minimizing the pixel-wise difference between the predicted segmentation and the groundtruth segmentation. The loss includes a weight decay with weight decay factor $\lambda = 0.001$. To account for the class imbalance in the dataset, the loss is weighted for each class k with a weight ω_k. Let $\mathcal{G}_k$ be the set of all indices of pixels labeled as class k in the training set $\mathcal{X}_{\mathrm{B01}}^{\mathrm{tr}}$ (for precise definition see Section 2.2.5). The frequency of each class k in the training set is defined by

$$n_k = \begin{cases} 0.85\dfrac{|\mathcal{G}_k|}{\sum_k |\mathcal{G}_k|}, & k \in \mathcal{L} \setminus \{\texttt{bg, cor, wm}\} \\ 0.15/3\,, & k \in \{\texttt{bg, cor, wm}\} \end{cases} \tag{5.4}$$

with $\mathcal{L}$ the 16 labels the model is trained on and if 85% of the training samples are sampled from labeled brain areas as described above. Then the class weights ω_k are defined as

$$\omega_k = \frac{\log(n_k)}{\sum_k \log(n_k)}\,. \tag{5.5}$$

This definition results in class weights that are centered around one and correspond to the logarithm of the inverse frequency of the area. Taking the logarithm mitigates the effect of very over- or under-represented classes. Otherwise very small classes get a very high importance, but for these classes training examples occur very seldom, which leads to a noisy loss. This led to model divergence in several of our experiments. The presented definition of class weights avoids this problem, while still assigning a larger weight to smaller classes and a smaller weight to larger classes.

Training was done with NAG, using a momentum factor $\mu = 0.9$ and a step-wise learning rate decay. The step sizes were chosen such that a reduction in learning rate occurs when the accuracy on the validation set $\mathcal{X}_{\mathrm{B01}}^{\mathrm{val}}$ plateaus. The `BaseNetwork` models were trained on 10 000 batches with a batch size of 40 using texture patches X_{texture}. The learning rate started with $\eta = 0.08$, and was decayed every 2000 batches by factor 2. The `AtlasAwareNetwork` models were trained by first training a `BaseNetwork`, transferring the learned weights to a `AtlasAwareNetwork`, and fine-

[1] Another commonly used data augmentation technique is to randomly shift the input image by a few pixels. Since due to the random sampling of the image patches from the histological sections, the same image patch will very rarely be seen twice, additionally introducing a random shift is not necessary here.

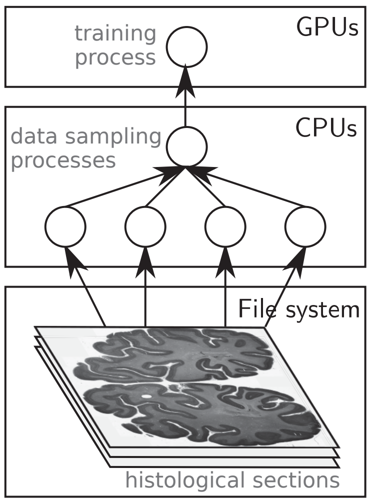


Figure 5.5: Parallel data sampling and training workflow. The data sampling processes sample a new batch from the histological sections on the file system while training process trains the model on the current batch on the GPUs.

tuning the `AtlasAwareNetwork` on additional 10 000 batches with batch size 40 using texture patches X_{texture} and atlas patches X_{atlas}. The learning rate was set to $\eta = 0.01$ at the beginning of this phase and reduced by factor 2 every 3000 batches.

After training, scores were calculated using all sections in the test split of the `B01Collection` in the following manner: Each annotated region was enclosed in its minimal bounding box. All pixels inside at least one bounding box (constituting the test dataset $\mathcal{X}^{\text{te}}_{\text{B01}}$) were predicted. From these predictions the Dice score and topological misclassification error were calculated using the annotated regions in combination with the automatic background segmentation as groundtruth. Figure 5.4b shows an example section with the bounding boxes and groundtruth used for evaluating the model.

5.2.5 Computational challenges during training

The large input patches and subsequent large model architectures lead to computational challenges:

- The sum of the internal activations gets so large that GPUs with large memory (VRAM) are required to accommodate the model during training. Typically, a `BaseNetwork` model requires around 400 MB GPU memory for one training sample.
- Large amounts of data needs to be processed leading to high I/O demands of the application. To train a `BaseNetwork`, 400 000 patches of size 2025×2025 need to be randomly cropped from the image files on the file-system and streamed to the GPU.

These challenges require the use of a high performance computing (HPC) system to train the models. The models presented in this thesis are trained on the JURECA supercomputer (Jülich Supercomputing Centre, 2018) and the JURON pilot system (*JURON* 2018) located at the Jülich Supercomputing Centre. The JURECA general-purpose cluster contains multiple GPU nodes with two NVIDIA K80 GPUs each (resulting in four visible GPUs per compute node), and the JURON GPU cluster is equipped with four NVIDIA Tesla P100 GPUs.

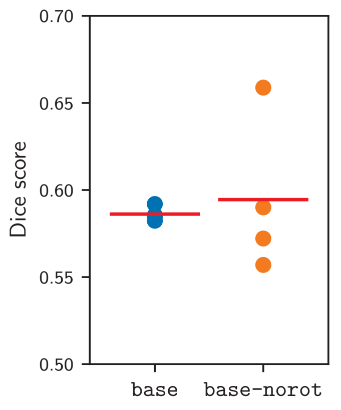
To efficiently use these resources, a custom software stack for training was implemented using TensorFlow (Abadi et al., 2015) and MPI. The models are trained on one compute node, utilizing all of the four available GPUs. For one training step, a batch needs to be sampled from the dataset on the file system, optionally processed on the CPU (e.g., applying data augmentation), sent to the GPU, and processed on the GPU with TensorFlow. The developed software utilizes several processes to sample batch $\mathcal{B}_{i+1}$ from the file system, while the model is being trained on batch $\mathcal{B}_i$ on the GPU (Figure 5.5).

Due to the large computational challenges, the automatic brain area mapping project was used on several occasions as a use-case for deep learning on HPC systems by the Jülich Supercomputing Centre. An in-depth analysis conducted by Oden et al. (2019) identified the random sampling from large files (containing the high-resolution brain sections) on the file system as the main bottleneck. Changing the data format of the section to a chunked HDF5 format, which allows faster random access, greatly increased the speed of the training (Oden et al., 2019).

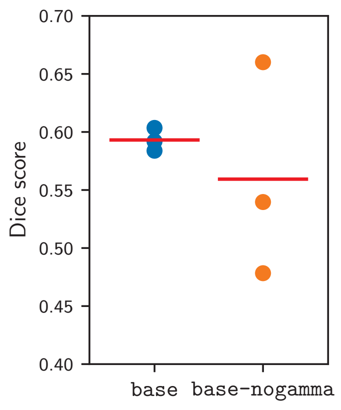
With this implementation the models presented in the following could be trained with batch sizes of around 40 utilizing all GPUs of one compute node in reasonable time. The training time of the models depends on the data that is processed; due to the I/O bottleneck and caching effects, models that are trained on fewer histological sections take less time to train than models trained on more histological sections (even when both are trained for the same number of iterations). Training times varied between 6-12 hours on one JURECA node for the patch segmentation models and 24 hours on one JURON node for the self-supervised models presented in Chapter 6.

5.3 Results

This section evaluates the `AtlasAwareNetwork` and its components. First, Section 5.3.1 shows the effect of the different data augmentation techniques. This is followed by a discussion of the influence of the atlas prior (Section 5.3.2). The influence of the atlas prior was researched in a similar manner in Spitzer, Amunts, et al. (2017). After these ablation studies, Section 5.3.3 looks inside the model and shows what features the model has learned at any given stage of the network. Finally, in Section 5.3.4 the patch segmentation approach is compared to the profile classification approach from Chapter 4.



(a) Dice score on rotated versions of $\mathcal{X}^{\text{val}}_{\text{B01}}$ ($\alpha \in \{0, \frac{1}{2}\pi, \pi, \frac{3}{2}\pi\}$)



(b) Dice score on gamma corrected versions of $\mathcal{X}^{\text{val}}_{\text{B01}}$ ($\gamma \in \{1, 0.5, 2\}$)

Figure 5.6: Effect of data augmentation on a `BaseNetwork`. Models trained with differently augmented versions of $\mathcal{X}^{\text{tr}}_{\text{B01}}$ are evaluated on augmented versions of $\mathcal{X}^{\text{val}}_{\text{B01}}$. The `base` model trained with all augmentations is invariant to rotations and gamma value differences.

5.3.1 Effect of data augmentation

During training, random rotation, random mirroring, and random gamma augmentation is used. The aim of these augmentations is to ensure that the resulting model is invariant to rotations and different gray values of the tissue, because these variations are not relevant for recognizing the cortical area in the input path. To ensure that the data augmentation was necessary, the `BaseNetwork` was trained on several augmented versions of the training dataset $\mathcal{X}^{\text{tr}}_{\text{B01}}$, resulting in the following trained models:

- `base`, which was trained on a rotation- and gamma-augmented dataset $\Psi_{\text{rot}}(\Psi_{\text{flip}}(\Psi_{\text{gamma}}(\mathcal{X}^{\text{tr}}_{\text{B01}})))$
- `base-norot`, which was trained on a gamma-augmented (but not rotation-augmented) dataset $\Psi_{\text{gamma}}(\mathcal{X}^{\text{tr}}_{\text{B01}})$
- `base-nogamma`, which was trained on a rotation-augmented (but not gamma-augmented) dataset $\Psi_{\text{rot}}(\Psi_{\text{flip}}(\mathcal{X}^{\text{tr}}_{\text{B01}}))$

To investigate the rotation invariance, `base` and `base-norot` were evaluated on four versions of the validation set obtained by rotating it by 0, 90, 180, and 270 degrees. Figure 5.6a shows the resulting Dice scores of the two models on these augmented versions of the validation set. While `base` has similar performance for each variant of the validation set (blue dots), `base-norot` displays a large variation (orange dots). Although it outperforms `base` for one specific rotation, in practice it cannot be guaranteed that sections are always presented to the model in this orientation, and thus the rotation-invariant `base` model should be preferred.

To evaluate the gamma augmentation, base and base-nogamma were evaluated on three versions of the validation set obtained by augmenting it with different gamma values of 1, 0.5, and 2. Again, the resulting Dice scores for each version of the validation set are close together for base, but not for base-nogamma (Figure 5.6b). In this case, the average Dice score of base (combining all three variants of the validation set, red line) is larger than the average Dice score of base-nogamma. These evaluations show that using data augmentation results in a rotation- and contrast/brightness invariant base model. The following models are trained with data augmentation. For simplicity, let $\Psi_{\text{rot}}(\Psi_{\text{flip}}(\Psi_{\text{gamma}}(\mathcal{X}^{\text{tr}}_{\text{B01}}))) \equiv \mathcal{X}^{\text{tr}}_{\text{B01}}$ in the following sections.

5.3.2 Influence of atlas prior

The AtlasAwareNetwork combines high-resolution texture information with an atlas prior extracted from the JuBrain atlas. To show the effect of the atlas prior, this section compares three models:

1. base which is trained only on high-resolution texture information,
2. only-atlas which is trained on atlas information and low-resolution texture, and
3. atlas which is trained on both high-resolution texture information and the atlas prior.

The only-atlas model is trained using the AtlasAwareNetwork with only the atlas prior path. It gets a 14-channel 8 µm resolution image as input. The first 13 channels contain probabilistic atlas information for each of the 13 visual areas in the training dataset and the 14th channel contains a 8 µm downscaled patch from the cell-body stained image. This low-resolution texture information is provided to the model to allow the segmentation of background, cortex and white matter. This information is not present in the probabilistic maps that only show how likely it is for an area to occur at the given input position. Thus the texture input is crucial to enable the comparison with the other two models. Note that a resolution of 8 µm might also allow the model to recognize very prominent areas such as hOc1 (V1) based on their cytoarchitecture, but for the most areas, a higher resolution is needed. Thus, the only-atlas model constitutes a baseline for the performance of the atlas model and shows how much information the atlas prior provides on its own.

The quantitative evaluation on the test set shows that atlas outperforms both base and only-atlas model by 11 and 6 Dice score points respectively (Table 5.1). Additionally, it has the lowest topological misclassification error, showing that the atlas in combination with the high-resolution texture information increases the topological correctness of the predictions. The predictions on three example sections from the test set (Figure 5.7), and the confusion matrices computed over the entire test set

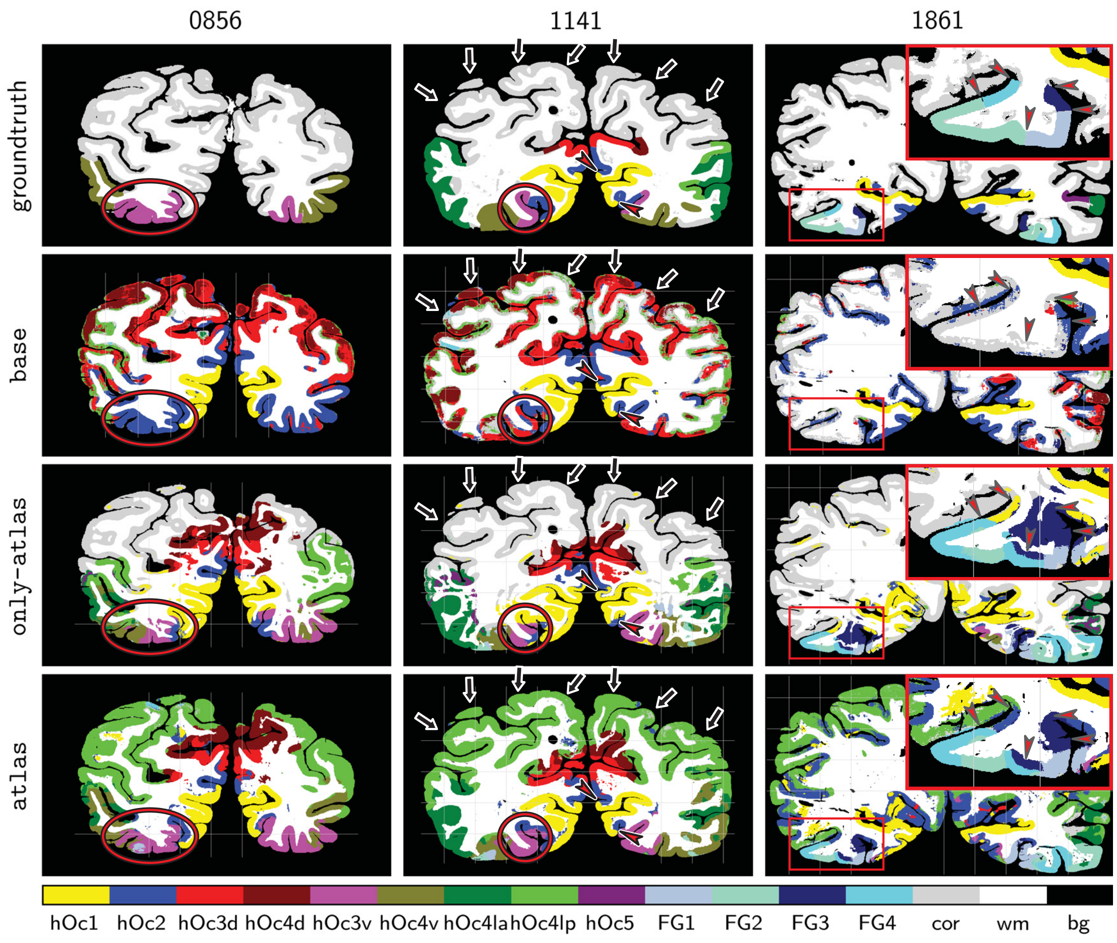


Figure 5.7: Qualitative comparison of predictions made by `base`, `only-atlas`, and `atlas` on three sections of B01. The `only-atlas` model produces topologically correct predictions, but does not segment the areas precisely (e.g., see `hOc1`/`hOc2` border marked with red marker on section 1141). The `atlas` model outperforms the other two models (e.g., red circles, and magnification on section 1861), but does not predict the “unknown cortex" label correctly (black arrows).

Table 5.1: Performance of models trained with texture (`base`), with atlas prior (`only-atlas`), and with both (`atlas`) on $\mathcal{X}_{\text{B01}}^{\text{val}}$. The influence of the atlas prior is evaluated by testing the `atlas` model on $\mathcal{X}_{\text{B01}-0}^{\text{te}}$ which contains zeros for the atlas prior.

	Dice	err
`base`	0.59	0.293
`only-atlas`	0.65	0.251
`atlas`	**0.71**	**0.211**
`atlas`($\mathcal{X}_{\text{B01}-0}^{\text{te}}$)	0.56	0.337

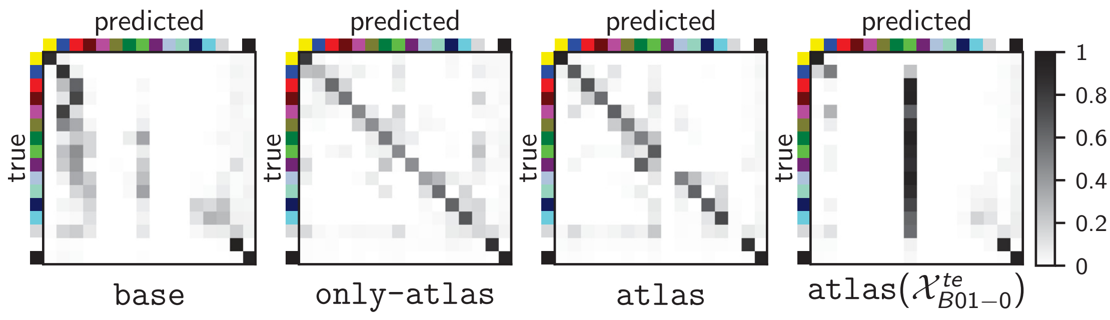


Figure 5.8: Confusion matrix of models trained with and without atlas prior. Legend in Figure 5.7.

(Figure 5.8) provide more details on the differences between the three models. All models have learned to correctly segment white matter and background. The `base` model has learned to predict areas `hOc1`, `hOc2`, `hOc3d`, and areas `FG3`, `FG4` in some locations. It has learned to predict the label `cor` for cortex that does not belong to any visual area in some locations. The `only-atlas` model predicts all areas in roughly the correct location, which can be seen in the clearly visible diagonal in the confusion matrix. It has learned to use the label `cor` for regions where the probability of a visual area is low (black arrows in Figure 5.7). However, it does not find precise borders between cytoarchitectonic areas, even when it would be possible (e.g., the `hOc1` border on section 1141 is not identified correctly; red marker in Figure 5.7). The `atlas` model, on the other hand, correctly predicts the general location of all areas except `hOc5` which is a very small area with very few examples in the training and testing set (red circles). However, `atlas` does not predict the unknown cortex class `cor` anymore for non-visual cortex, and instead prefers area `hOc4lp` (black arrows).

Overall, the effect of the atlas prior is to enable the prediction of areas that might not be directly distinguishable by the texture alone. Its power can be observed in `atlas`, which outperforms models trained only on texture and only on the atlas prior. But how much does `atlas` depend on the atlas information, and how much capabilities does it preserve from `base`? To answer this question, `atlas` was evaluated on the $\mathcal{X}^{\mathrm{te}}_{\mathrm{B01-0}}$ dataset which contains zeros for the atlas prior input. This evaluation allows us to see how dependent the model is on the atlas information. The performance of `base` gives us an upper bound for the performance that we can expect of a thusly evaluated model. The closer the scores of `atlas` evaluated on $\mathcal{X}^{\mathrm{te}}_{\mathrm{B01-0}}$ are to those of `base`, the less `atlas` actually depends on the atlas prior.

The quantitative evaluation shows that `atlas` evaluated on $\mathcal{X}^{\mathrm{te}}_{\mathrm{B01-0}}$ performs 3 Dice score points worse than `base`. Additionally it has a higher topological misclassification error. Evaluating `atlas` on $\mathcal{X}^{\mathrm{te}}_{\mathrm{B01-0}}$ still allows the prediction of areas `hOc1`, `hOc2` and in some cases `FG3`, however most of the predictions are made as area `hOc4lp`

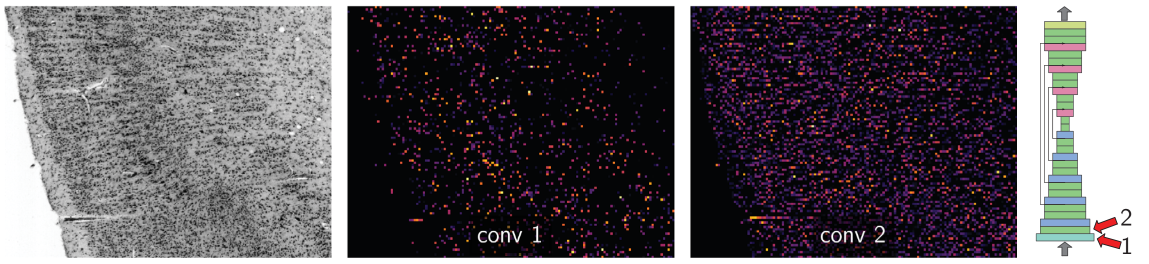
(Figure 5.8). This preference for area hOc4lp could already be seen in the normal evaluation of atlas, which predicted most unknown cortex as hOc4lp. This evaluation shows that for areas hOc1 and hOc2 the model does not need the atlas prior information. Other areas, however are dependent on a non-zero signal from the atlas prior, even if base was able learn them.

5.3.3 Feature map analysis: What did the model learn?

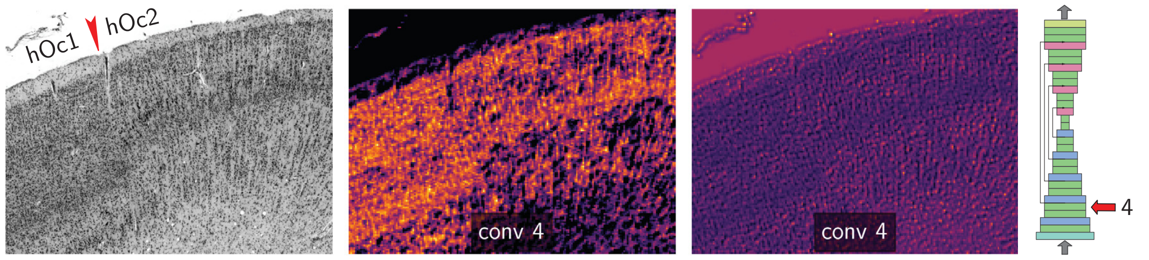
After showing the performance of the proposed model for area segmentation, we now qualitatively examine what features the model focuses on. This is similar to the saliency analyses in Section 4.3.2, however instead of using GradCAM, this section analyzes the feature maps directly. The reason is that a coarse GradCAM saliency map highlights the entire high-resolution input patch as important, because small differences in texture distributed over the whole input patch define different cortical areas. An analysis of feature maps at different network depths shows which features are learned where in the network.

In the following, feature maps of base (trained without the atlas prior) are analyzed on a section showing hOc1 (primary visual cortex) and hOc2 (secondary visual cortex).[2] Figure 5.9 shows examples of feature maps at different depths of the network. With increasing depth, the feature maps are based on a larger receptive field and thus may highlight increasingly complex structures. In the first block (before the first max pool layer), the feature maps highlight cell bodies, often depending on size, and the white matter between clusters of cells (examples in Figure 5.9a). After the first max pool layer, the feature maps are increasingly specific to laminae with different cell densities (examples in Figure 5.9b). Deeper in the network, feature maps start to distinguish between laminae in the primary and secondary visual cortex, and specialize on different parts of the background (examples in Figure 5.9c). At the bottom of the network between the downsampling and upsampling paths the feature maps show activations at descriptive cytoarchitectonic properties like laminar layers, the white matter/cortex border and the cortex/background border. Additionally, feature maps for individual cortical areas exist. In the upsampling path these specific feature maps increase in resolution and further specialize on the different cortical areas in the training set (examples in Figure 5.9d). These examples show that the model focuses on cell structure and arrangement at first and then builds up more complex features for individual cortical laminae. These concepts are similar to the feature that neuroscientists focus on when identifying cortical areas. This indicates that the model indeed uses cytoarchitecture to distinguish between cortical areas.

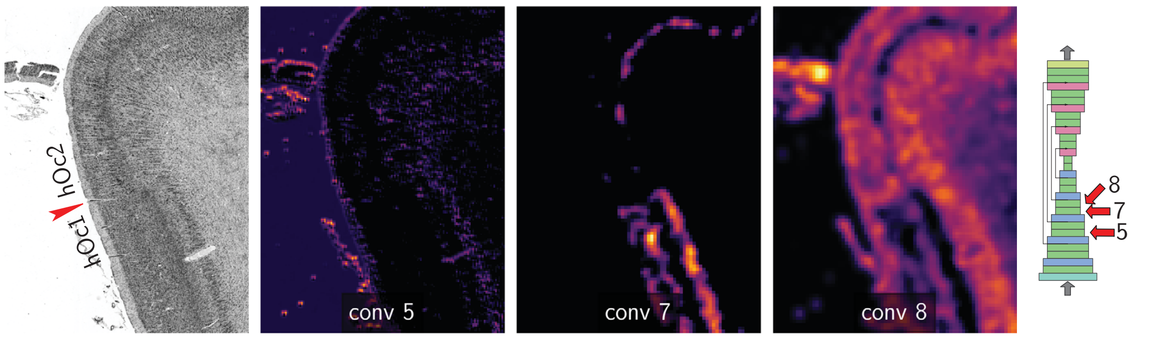
[2]Thanks to Kai Kiwitz (Heinrich-Heine-University Düsseldorf) for providing initial analyzes of the feature maps.



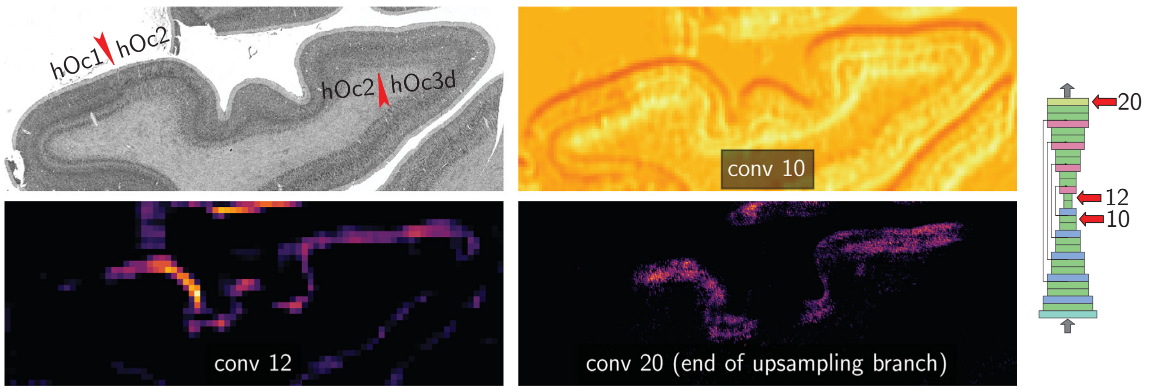
(a) Feature maps highlighting large cells and all cells, respectively.



(b) Feature maps highlighting cell dense and cell poor laminae, respectively.



(c) Feature maps specific to cells in lamina III and V/VI in non primary visual cortex, and later in the network, feature maps highlighting granular and cell poor laminae, respectively.



(d) Feature map for background/cortex edge, and feature map activating mostly in `hOc2`. This feature map is refined in the upsampling branch.

Figure 5.9: Feature maps at different depths of the `BaseNetwork` for one input patch showing areas `hOc1` and `hOc2`. The red marker denotes the border between the two areas. The number of the convolutional layer that produced the feature map is shown at the bottom of each map. The schematic drawing of the network architecture shows the location of the respective convolutional layers in the neural network.

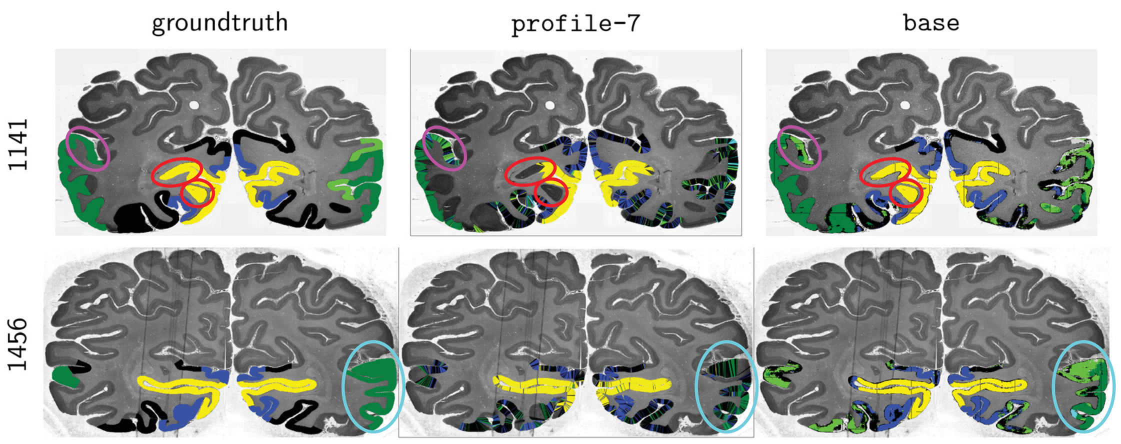


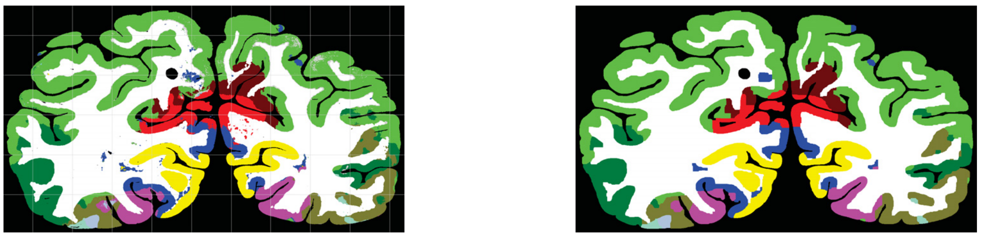
Figure 5.10: Comparison of area classification results from `profile-7`, a `ProfileNetwork` trained for profile classification, and `base`, a `BaseNetwork` trained for patch segmentation. Both models make similar mistakes in some regions (e.g., pink circle), but overall the `base` model performs better (e.g., turquoise circle) and can segment oblique patches of cortex (e.g., red circles).

5.3.4 Comparison with profile classification model

In order to allow for a fair comparison between the end-to-end segmentation and the profile classification, the `base` model is compared to the `profile-7` model. Since the `BaseNetwork` was trained on more labels than the `ProfileNetwork`, the predictions of labels `hOc3d`, `hOc3v`, `hOc4d`, `hOc4v`, `hOc5`, `FG1`, and `FG2` were grouped together in the class `all` and evaluations were calculated only on cortex showing one of the brain areas in the dataset, i.e., excluding all parts labeled with `bg`, `wm`, or `cor`. With this it is ensured that both models have seen the same data during training and that the evaluation is carried out on parts of the section that both models have predictions for.

Figure 5.10 qualitatively compares the results on two test sections. Visually, the patch segmentation model `base` performs better. The segmentation approach additionally allows to segment very oblique regions of the cortex where a profile is not defined (e.g. islands of cortex in white matter, red circles on Figure 5.10). Overall, `base` exhibits less noise than `profile-7` (turquoise circle), although interestingly both models make similar mistakes in some regions (pink circle).

To quantitatively compare the performance on the test set, the segmentation result is reduced to individual streamline classifications by assigning to each streamline in the test set $\mathcal{X}_{\mathrm{pr}}^{\mathrm{te}}$ the maximally predicted class under this streamline. With this preparation, the segmentation `BaseNetwork` has a Dice score of 0.63, and the `ProfileNetwork` a Dice score of 0.62. This shows that even in an evaluation procedure which is tailored towards profile classification, the segmentation model outper-



(a) Prediction of the CNN (b) Smoothed predictions with CRF

Figure 5.11: Applying a CRF that enforces label coherency to the output of the `atlas` model smooths the output.

forms the profile classification model. This increase in Dice score, together with the above mentioned advantages of the segmentation approach, make the patch segmentation model better suited for area recognition than the profile classification model.

5.4 Discussion

This chapter introduced an end-to-end CNN model for brain area segmentation. This model extracts features from the high-resolution histological sections and combines them with prior information given in form of probabilistic maps of the areas that should be predicted. The atlas prior gives relevant context to the model and improves the segmentation Dice score by 11 points. The model is invariant to rotations and gray value differences in the inputs. It outperforms the profile model from Chapter 4.

The model employs a pixel-wise classification where each pixel is classified independently from the neighboring pixels. Although the resulting segmentations are in general smooth and stable (Figure 5.7), they could be improved by explicitly modeling the spatial relationships between the pixels and enforcing label coherency by using for example a conditional random field (CRF). There are several examples in literature that apply a CRF on the CNN predictions (L.-C. Chen et al., 2018) or embed the CRF training in the CNN (Li and Ping, 2018; Zheng et al., 2015). Figure 5.11 demonstrates the effect of applying a CRF that penalizes the assignment of different labels to neighboring pixels to an example prediction of the `atlas` model. The resulting prediction is smoother than the raw output of the CNN. Additionally, a CRF could enable the direct modeling of the topology of areas in the brain. The investigation of such methods remains for future work.

The atlas data is essential for the good performance. However, including the atlas data includes an additional component in the workflow. In particular, a registration from the atlas to the individual sections is needed. Although only a linear registration is used to project the probability maps to the current section, it is obvious that with a low-quality registration the performance of the model will suffer, if it relies on the atlas data to select probable labels for each pixel. In Section 5.3.2 we have seen that if the atlas information is missing from the model, it performs worse than a model trained on only the high-resolution texture. This makes it necessary to check the output of the registration for each section that the model should be applied to, introducing a manual component in the otherwise fully automatic segmentation method.

Including atlas information is only one possible way of including prior information or contextual knowledge in the model. As argued in Section 5.1.1, a multi-scale approach for including context is not feasible as brain areas are in general not correlated with the local curvature of the cortex. One idea to allow the inclusion of larger scale context is to remove or unravel the curvature of the cortex. This is similar to working on the inflated surface of the brain (Fischl, Sereno, et al., 1999). However, cortical surface inflation only works for complete 3D brain volumes, not for individual histological sections, because on the latter the cortex does not form a closed ribbon. Defining a coordinate system along the cortical ribbon is thus only possible for parts of the section. In addition, unraveling the cortex means non-linearly deforming the cortex which introduces artifacts. Another way to include contextual information is to include views of the same position in the previous or next section. These views could be beneficial in cases when the original section has an artifact at the current position. Exploring this idea was out of the scope of this thesis and is left to future work.

Another option for including prior knowledge about the brain is pre-training the model on a task that teaches it about the general structure of the brain. Such a pre-training task additionally addresses the problem that training data for the brain area segmentation is limited, and that the variability of each brain area is large. Chapter 6 introduces a self-supervised pre-training method that leverages unlabeled data that is readily available for our task. This pre-training complements and improves the supervised area segmentation to a point where a pre-trained `BaseNetwork` (without the atlas prior) performs almost as well as the `atlas` model introduced in this chapter.

6 Self-supervised brain area analysis

Chapter 5 has shown that prior knowledge is important for good performance of the automatic brain area segmentation model (Section 5.3.2). One possible reason for this dependence on additional prior information is the limited number of training samples: For underrepresented classes, the samples do not contain enough variance to allow the model to learn robust features for this class. Several recent works propose *self-supervised* tasks that exploit structure present in the data (e.g., spatial relationships between patches sampled from one image) to train a deep model without requiring explicitly labeled training data (e.g., Doersch, Gupta, et al., 2015; Noroozi and Favaro, 2016; see Section 3.4.3). In these works, the resulting models are used to extract general-purpose features from images or fine-tuned on a supervised classification task. Following this line of research, this chapter introduces a self-supervised task that exploits unlabeled high-resolution cytoarchitectonic sections.

Leveraging the inherent 3D relationship between individual 2D sections of one brain volume, the spatial location of an image patch on the brain surface can be calculated. Given two input patches sampled from the same brain volume, a CNN is trained to predict the locations and the geodesic distance between the locations on the brain surface (see Figure 6.1). This self-supervised distance and location task is a suitable auxiliary task for area segmentation. By pre-training the area segmentation model on this task, the segmentation accuracy is significantly improved. Training models on the self-supervised task allows to learn an internal feature representation that encodes cytoarchitectonic properties. Without explicitly training the model on the concept of brain areas, it learns to identify plausible brain regions and several primary cytoarchitectonic areas.

The self-supervised auxiliary distance task and the applicability for pre-training semantic segmentation models was already reported in the article “Improving Cytoarchitectonic Segmentation of Human Brain Areas with Self-supervised Siamese Networks” (Spitzer, Kiwitz, et al., 2018). In a second contribution, “Compact feature representations for human brain cytoarchitecture using self-supervised learning” (Spitzer, Amunts, et al., 2018), the internal feature representation was analyzed. This chapter extends these two contributions.

In the remainder of this chapter, Section 6.1 introduces and motivates the self-supervised distance and location regression task and the neural network architecture in detail. The training procedure and training data are introduced in Section 6.2.

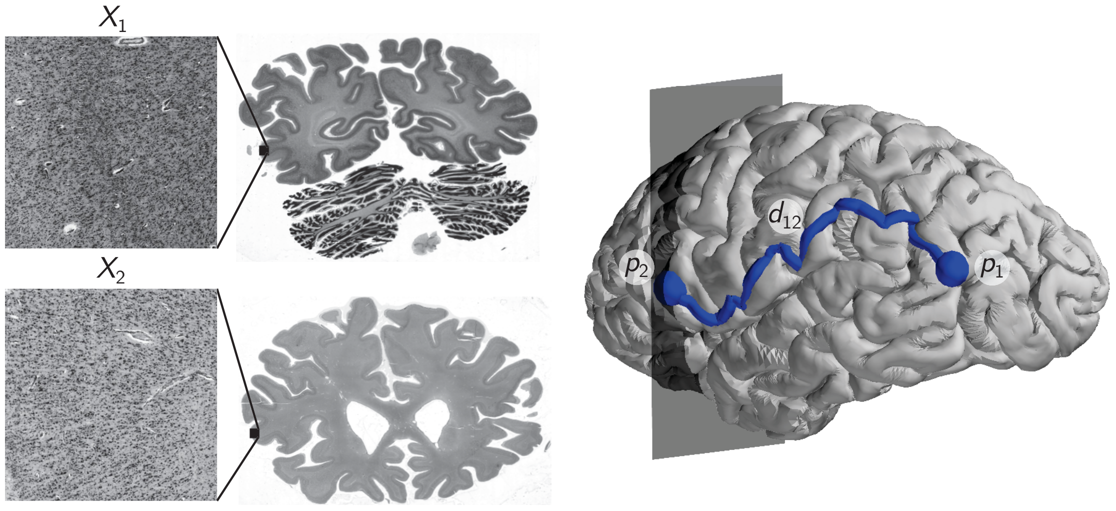


Figure 6.1: Using a 3D reconstruction, two patches X_1 and X_2 sampled from sections of the same brain volume are associated with their locations p_1 and p_2 on the reconstructed pial surface. The distance d_{12} is calculated as the shortest geodesic distance between p_1 and p_2 along the pial surface.

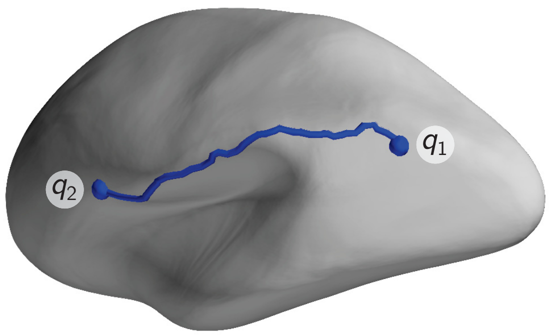
Evaluation will be done in two parts: First, Section 6.3 evaluates the applicability of the self-supervised model to improve supervised area segmentation. Second, Section 6.4 shows what knowledge the internal feature representations of the self-supervised model encode about the brain and how they can be used for unsupervised exploration of the brain.

6.1 Neural network model for the self-supervised task

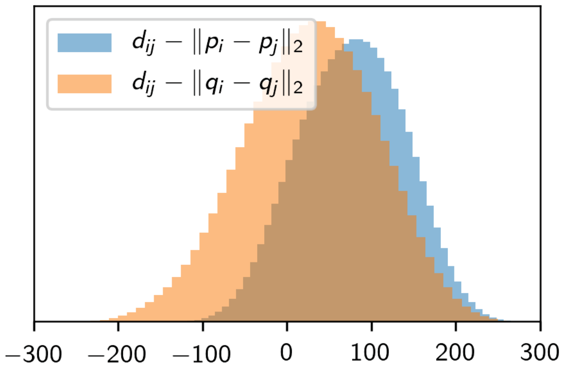
The aim of this chapter is to improve the supervised area segmentation from Chapter 5. Since classical cytoarchitectonic mapping is an extremely time-consuming expert task, we cannot easily overcome the problem of limited training data (Section 1.1). The idea is to leverage unlabeled brain sections from a 3D reconstructed whole-brain volume, of which much larger amounts can be acquired in reasonable time. Section 6.1.1 introduces the self-supervised task that uses the unlabeled brain sections and Section 6.1.2 describes the neural network architecture. The contents of this section were also described in Spitzer, Kiwitz, et al. (2018).

6.1.1 Distance and location regression task

The self-supervised distance and location regression task uses the inherent 3D relationship between individual sections in the same brain volume given by a 3D reconstruction (see Figure 6.1). Let $X_i \in \mathbb{R}^{1019\times1019}$ be a patch sampled from the



(a) Inflated left surface with locations q_1 and q_2 for inputs X_1 and X_2 from Figure 6.1.



(b) Comparison of differences between the geodesic distance d_{ij} and the Euclidean distance $\|p_i - p_j\|_2$ of locations on the pial surface and differences between d_{ij} and $\|q_i - q_j\|_2$ of locations on the inflated surface.

Figure 6.2: The Euclidean distance between locations q_i and q_j on the *inflated surface* approximates the geodesic distance d_{ij} more closely than the Euclidean distance between locations p_i and p_j on the pial surface.

cortex of one brain volume with $p_i \in \mathbb{R}^3$ its corresponding center coordinate on the reconstructed brain surface and X_j and p_j accordingly for a second patch defined in another location of the same brain sample. The geodesic distance d_{ij} is calculated by finding the shortest distance between coordinates p_i and p_j along the surface of the brain. The *distance task* is to determine d_{ij}, given only the content of patches X_i and X_j but not their location.

Why is this task useful? The aim of this chapter is to leverage unlabeled training data for automatic brain area parcellation. Training a model on the proposed distance task forces it to find features that are similar in spatially close patches and different in spatially distant patches. Cytoarchitecture differs throughout the cortex. It is similar for a small local region (a cytoarchitectonic area) and differs for examples outside this region. Thus, the hypothesis is that the model learns features that encode cytoarchitectonic properties in order to solve the distance regression task. The features, in turn, are highly relevant for supervised brain area segmentation.

The CNN trained on this task learns an embedding from patches of cortex X to a feature vector $g(X) \in \mathbb{R}^{32}$. The distance $\hat{d}_{ij}$, that is predicted by the network for inputs X_i and X_j, is the squared Euclidean distance between the feature representations $\|g(X_i) - g(X_j)\|_2^2 = \hat{d}_{ij}$. In this setup, the embedding g has to encode the location of the patch on a representation of the cortical manifold. It will place two semantically similar input images (e.g., with similar cytoarchitecture) close together, and semantically different input images far apart.

In order to guide the model to find a suitable embedding faster, the *location task* is defined: Let $q_i \in \mathbb{R}^3$ be the location of patch X_i on the inflated brain surface. Given a patch X with feature representation $g(X)$, predict its location on the inflated brain surface $\hat{q} = h(g(X))$ using a linear function h. The inflated brain surface is calculated from the white matter surface by blowing up each hemisphere and thus smoothing out all gyri and sulci (see Figure 6.2a). This results in a closer correlation between the Euclidean distance of locations on the inflated surface $\|q_i - q_j\|_2$ and geodesic distance d_{ij} than between the Euclidean distance of locations on the pial surface $\|p_i - p_j\|_2$ and d_{ij} (Figure 6.2b). Thus, the location task relates spatially close samples even though they do not necessarily appear together as a pair during training. Although the geodesic distance is only defined for intra-hemisphere pairs, the location is defined for both hemispheres. We can assume that the embedding of points X_i and X_j from different hemispheres but same brain area should be very similar. This is enforced if the location q is defined to be the corresponding location on the left inflated brain surface. This means that for points on the right surface the location q is calculated by registering the right surface to the left surface. This means that the precision of the location q depends on the quality of the registration. Since the hemispheres are not completely mirror-symmetric (Toga and Thompson, 2003; Zilles, Dabringhaus, et al., 1996), it is possible that right-hemisphere patches X from `area1` are mapped to a left-hemisphere location q in `area2`. Thus it is expected that the location task alone is not sufficient to make the model learn cytoarchitectonically relevant features since the location is not precise enough. However, it might guide the model towards a suitable embedding g.

Together, the location and distance tasks learn an embedding $g(X)$ of cortical patches X which is similar for patches that are in similar locations (on the same or on different hemispheres) and which is dissimilar for patches that are far apart. These tasks are called self-supervised, because they are defined on information which is inherently present in a 3D brain volume reconstruction and require no manual labels.

6.1.2 Siamese network architecture

The `SiameseNetwork` is a Siamese network that computes distance and location regression based on two input patches X_1 and X_2. First, the feature embeddings $g(X_1)$ and $g(X_2)$ are computed in two branches with identical architecture and shared weights (hence the name *Siamese* network). The architecture of the branches corresponds to the downsampling path of the `BaseNetwork` for supervised segmentation (Section 5.1.1) with a 32-channel fully connected layer added on top of the last convolutional layer. This design allows easy transfer of the weights for finetuning. From these feature embeddings, the network outputs the predicted geodesic distance between p_1 and p_2 as the squared Euclidean distance of $g(X_1)$ and $g(X_2)$:

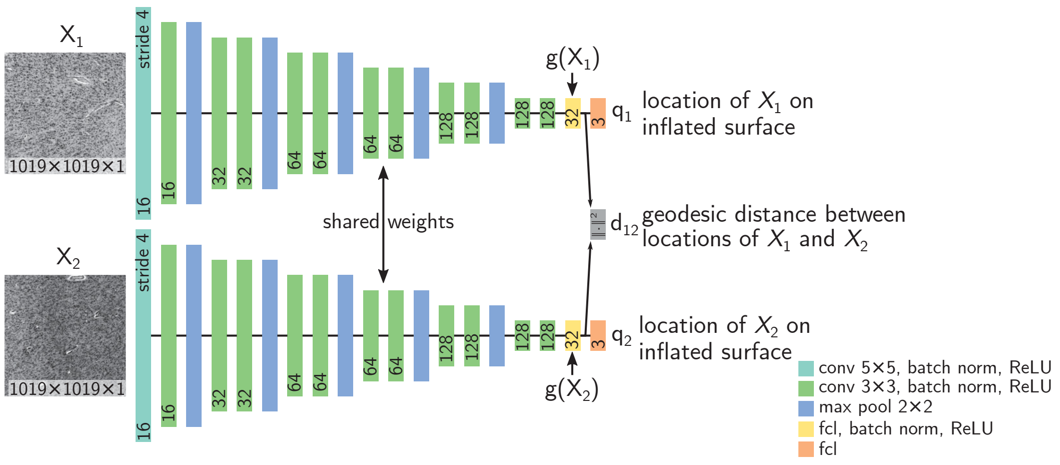

X₁
1019×1019×1
stride 4
16
16
32
32
64
64
64
64
128
128
128
128
32
3
g(X₁)
q₁
location of X₁ on inflated surface
shared weights
d₁₂
geodesic distance between locations of X₁ and X₂
X₂
1019×1019×1
q₂
location of X₂ on inflated surface
g(X₂)
conv 5×5, batch norm, ReLU
conv 3×3, batch norm, ReLU
max pool 2×2
fcl, batch norm, ReLU
fcl

Figure 6.3: `SiameseNetwork` for distance and location regression. The network takes two cortical patches X_1 and X_2 as input and predicts their locations q_1 and q_2 on the cortical surface and their geodesic distance d_{12}.

$\hat{d}_{12} = \|g(X_1) - g(X_2)\|_2^2$. The location of each input patch is predicted by adding a fully connected layer h with three output nodes to the top of each branch: $\hat{q}_i = h(g(X_i))$. Figure 6.3 shows the `SiameseNetwork` and its outputs. It takes two patches $X_1, X_2 \in \mathbb{R}^{1019\times1019}$ as inputs and predicts locations $\hat{q}_1, \hat{q}_2 \in \mathbb{R}^3$ and distance $\hat{d}_{12} \in \mathbb{R}$. The network has 7.4×10^5 parameters.

6.2 Self-supervised training and fine-tuning

The models trained on the self-supervised distance and location regression task are used in two different ways: First, they are used as starting point for fine-tuning a model for the brain area segmentation task (Section 6.3). Second, they are used as feature extractors to explore the automatically learned feature representation $g(X)$ (Section 6.4). The training data for these two use-cases is presented in Section 6.2.1. The training loss and hyper-parameters for training on the self-supervised task are introduced in Section 6.2.2. Section 6.2.3 describes the fine-tuning procedure for adapting the self-supervised models to supervised brain area segmentation. The pre-training and fine-tuning procedure was also described in Spitzer, Kiwitz, et al. (2018).

6.2.1 Data sampling

The self-supervised models are trained on datasets sampled from the `BigBrain Collection`. For this collection, pial surface and gray/white matter border meshes are available (L. Lewis et al., 2014). Additionally, we have access to transformation

Table 6.1: Training and evaluation datasets sampled from the `BigBrainCollection` for training self-supervised `SiameseNetworks`.

$\mathcal{X}$	$\lvert\mathcal{X}\rvert$	regions of the `BigBrainCollection` used for sampling
$\mathcal{X}^{\mathrm{tr}}_{\mathrm{vis}}$	200 000	sections 1-3000 in train/val split
$\mathcal{X}^{\mathrm{tr}}_{\mathrm{BB}}$	500 000	all sections in train/val split
$\mathcal{X}^{\mathrm{te}}_{\mathrm{BB}}$	2000	all sections in test split
$\mathcal{X}^{\mathrm{tr}}_{\mathrm{BBl}}$	200 000	left hemisphere of sections in train/val split
$\mathcal{X}^{\mathrm{te}}_{\mathrm{BBl}}$	150 000	left hemisphere of sections in test split
$\mathcal{X}^{\mathrm{tr}}_{\mathrm{BBr}}$	200 000	right hemisphere of sections in train/val split
$\mathcal{X}^{\mathrm{te}}_{\mathrm{BBl}}$	150 000	right hemisphere of sections in test split

files which allow the calculation of locations in the 3D reconstructed brain volume space from 2D pixel coordinates and section numbers. Pairs of patches (X_i, X_j) with $X \in \mathbb{R}^{1019\times1019}$ and resolution 2 µm are sampled from two randomly selected training sections of the `BigBrainCollection` in such a way that the center of the patch is located halfway between pial surface and gray/white matter border. Their geodesic distance d_{ij} and the locations q_i and q_j on the left inflated surface are calculated and used as groundtruth labels during training. Each patch is rotated by a random angle $\alpha \in [0, 2\pi]$, randomly flipped in left-right direction, and gamma augmented by a random value $\gamma \in [0.5, 2]$ (see Section 5.2.3). For clarity, we define $\Psi_{\mathrm{rot}}(\Psi_{\mathrm{flip}}(\Psi_{\mathrm{gamma}}(\mathcal{X}^{\mathrm{tr}}))) \equiv \mathcal{X}^{\mathrm{tr}}$ for the remainder of this chapter.

In Section 6.3 the model is trained on patches extracted from different sets of sections to investigate the impact of pre-training different self-supervised models for the segmentation task. In Section 6.4 several models are trained on patches from one hemisphere only. The datasets that these models are trained on are created as described above, but by restricting the sections/regions where the patches were sampled from. Table 6.1 summarizes the different training and evaluation datasets that are used in the following.

6.2.2 Self-supervised training

The model is trained on the L1 loss between predicted and groundtruth distances and locations

$$L_{\mathrm{dist}}(X_1, X_2) = |\hat{d}_{12} - d_{12}|\,, \tag{6.1}$$

$$L_{\mathrm{loc}}(X) = \|\hat{q} - q\|_1\,. \tag{6.2}$$

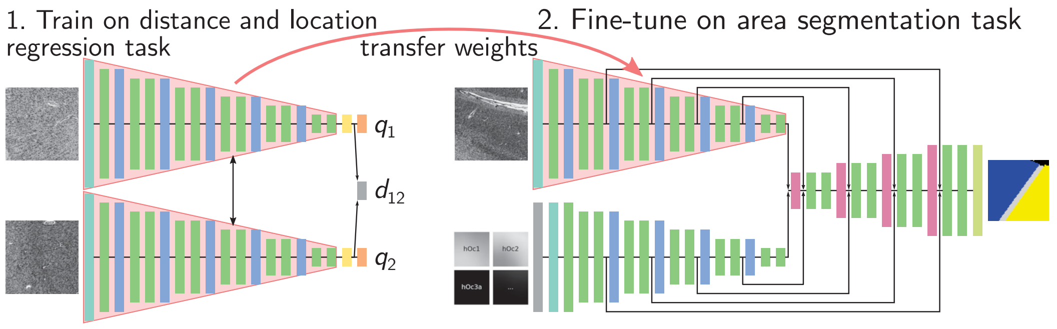

1. Train on distance and location regression task
transfer weights
2. Fine-tune on area segmentation task
q_1
d_{12}
q_2
hOc1
hOc2
hOc3a
...

Figure 6.4: Fine-tuning from self-supervised distance and location task improves supervised area segmentation. The texture downsampling path of the `AtlasAwareNetwork` is initialized by the learned weights from the `SiameseNetwork`.

The final training loss is a weighted combination of L_{dist} and L_{loc} with $\beta = 10$:

$$L(X_1, X_2) = L_{\text{dist}}(X_1, X_2) + \beta(L_{\text{loc}}(X_1) + L_{\text{loc}}(X_2)) \,. \tag{6.3}$$

L is regularized with a weight decay of all weights and biases, except those in the final dense layers to allow the regression of large values. The weight decay factor is set to $\lambda = 0.001$. Models were trained for 25 epochs (iterate over the entire training dataset 25 times) with a batch size of 32 using NAG with a momentum factor $\mu = 0.9$. The initial learning rate was $\eta = 0.001$. The learning rate was decreased every 5 epochs by factor 2. Inputs to the model were normalized by the mean and standard deviation of gray values in the training set.

6.2.3 Fine-tuning for area segmentation

After training a self-supervised model, its weights are transferred to the texture stream of the `AtlasAwareNetwork` (Figure 6.4). The segmentation network is trained as described in Section 5.2 with an iterative training procedure, but with a lower learning rate. For the first training phase (training the `BaseNetwork`), the learning rate was set to $\eta = 0.01$ and reduced every 2000 batches by factor 2. To enable faster adaption of the weights in the randomly initialized upsampling path, the learning rate was increased by factor 4 for these layers. For the second training phase (training the `AtlasAwareNetwork`), the learning rate was set to $\eta = 0.005$ and reduced every 3000 batches by factor 2. Again, the atlas-prior path had an increased learning rate by factor 8.

Although a 3D reconstruction is needed to train the Siamese network on the distance and location regression task, the fine-tuning can occur on any (unregistered) dataset of labeled histological sections. As described in Section 6.3.1, the model can be pre-trained and fine-tuned on two separate brain volumes. This allows us to utilize brain volumes with and without registration for the means of automatic brain area segmentation.

6.3 Improving supervised segmentation by pre-training

The principal motivation for designing the self-supervised distance and location regression task is to leverage unlabeled histological sections to pre-train the area segmentation model. This section shows that the self-supervised pre-training significantly increases the performance of the segmentation models and even allows the training of segmentation models on several brains. Section 6.3.1 evaluates the importance of the different components of the self-supervised loss function and how the training set of the self-supervised model influences the fine-tuned area segmentation performance. Then, Section 6.3.2 compares the pre-trained and randomly initialized segmentation models from Chapter 5 in detail. Finally, in Section 6.3.3 the generalizability of the resulting supervised segmentation models to unseen brain volumes is discussed. In the following the notation `model1[model2]` signifies that `model1` was trained (fine-tuned) on weights initialized from `model2`. The experiments presented in Sections 6.3.1 and 6.3.2 are an extension to the results presented in Spitzer, Kiwitz, et al. (2018).

6.3.1 Evaluation of model components

For pre-training neural networks, many auxiliary tasks can be used. The more similar the auxiliary task is to the original task (e.g., similar task, similar data), the better the fine-tuned network will perform on the original task (Azizpour et al., 2015; Mahajan et al., 2018; Yosinski et al., 2014). This section examines how different pre-training tasks influence the performance of the fine-tuned area segmentation network.

Training dataset for self-supervised model The brain area dataset contains visual areas which are located in the posterior part of the brain (occipital lobe). Thus, a self-supervised model that has been trained on sections 1-3000 can learn features that are specific to these areas of the brain. This should result in better fine-tuning performance.

To test this, two self-supervised models were trained and used to fine-tune the `BaseNetwork` on the `B01Collection` ($\mathcal{X}_{\mathrm{B01}}$): The first one was trained on $\mathcal{X}_{\mathrm{BB}}$ containing 500 000 patch pairs from the entire `BigBrainCollection` (`base[siam-BB]`), while the second one was trained on $\mathcal{X}_{\mathrm{vis}}$ containing 200 000 patch pairs from sections 1-3000 of the `BigBrainCollection` (`base[siam-vis]`). Table 6.2 shows that `base[siam-vis]` performs better than `base[siam-BB]` by three Dice score points.

Table 6.2: Comparison of supervised models trained in Chapters 5 and 6. The table compares `base` (no additional data) with models using the atlas prior (`only-atlas`, `atlas`), models pre-trained on different self-supervised tasks (`base[siam-vis]`, `base[siam-BB]`, `base[siam-vis-dist]`, `base[siam-vis-loc]`), and models trained on both (`atlas[siam-vis]`).

name	texture?	atlas?	L_{dist}?	L_{loc}?	Dice	err
`base`	✓	✗	✗	✗	0.59	0.293
`base[siam-vis]`	✓	✗	✓	✓	0.67	0.209
`base[siam-BB]`	✓	✗	✓	✓	0.64	0.236
`base[siam-vis-dist]`	✓	✗	✓	✗	0.62	0.255
`base[siam-vis-loc]`	✓	✗	✗	✓	0.67	0.210
`only-atlas`	✗	✓	✗	✗	0.65	0.251
`atlas`	✓	✓	✗	✗	0.71	0.211
`atlas[siam-vis]`	✓	✓	✓	✓	**0.74**	**0.174**

This shows that the self-supervised model learned features specific to visual areas that the supervised model was able to leverage. Note that since the self-supervised and the segmentation models are trained on two different brain volumes (BigBrain and B01), overfitting on B01 by the self-supervised model is not possible. Due to the better performance of `base[siam-vis]`, the following self-supervised models are trained on $\mathcal{X}_{\text{vis}}$.

Influence of loss function on distance and location regression The training loss for the self-supervised model consists of two components (Equation (6.3)): the distance loss (Equation (6.1)) and the location loss (Equation (6.2)). To evaluate the influence of each of these components on the self-supervised model, and subsequently on the fine-tuned segmentation model, three self-supervised models were trained: `siam-vis-dist` on the distance loss L_{dist}, `siam-vis-loc` on the location loss L_{loc} and `siam-vis` on the combined loss L.

Table 6.3 shows the performance of these models on the distance and location regression tasks. The measure err_{dist} is calculated as the mean absolute difference of predicted and groundtruth distance between all pairs (X_i, X_j) in the test set $\mathcal{X}_{\text{BB}}^{\text{te}}$ containing 2000 input pairs from both hemispheres:

$$\text{err}_{\text{dist}} = \frac{\sum_{i,j} |d_{ij} - \hat{d}_{ij}|}{|\mathcal{X}_{\text{BB}}^{\text{te}}|} . \tag{6.4}$$

The measure err_{loc} is calculated as the mean Euclidean distance between predicted

Table 6.3: Performance of self-supervised models on distance and location regression. `siam-vis` is trained on the combined loss from Equation (6.3), `siam-vis-dist` on the distance loss from Equation (6.1), and `siam-vis-loc` on the location loss from Equation (6.2).

	err_dist [mm]	err_loc [mm]
`siam-vis`	24.5	23.4
`siam-vis-dist`	31.5	-
`siam-vis-loc`	-	21.9

and groundtruth location on the left inflated hemisphere for all patches X_i:

$$\text{err}_\text{loc} = \frac{\sum_i \|q_i - \hat{q}_i\|_2}{2|\mathcal{X}_\text{BB}^\text{te}|}. \tag{6.5}$$

The self-supervised model has an average error of 24.5 mm when predicting geodesic distances, and an average error of 23.4 mm when predicting locations on the inflated cortical surface. To put these numbers in relation with the dimensions of the test dataset, err_dist is 12% of the maximal geodesic distance in test dataset, and err_loc is 15% of the maximal Euclidean distance between two locations in the test dataset. A location error of 23.4 mm is in the order of brain area size. Such an error must be expected when assuming that the model learns features encoding cytoarchitectonic properties, as the cytoarchitecture does not change within an area and thus more precise localization based on cytoarchitectonic features is not possible.

The `siam-vis` model has a lower distance error than `siam-vis-dist`. This shows that the location loss L_loc could help the model to find a suitable feature representation $g(X)$. Compared to `siam-vis-loc` which was trained only on L_loc, the combined model `siam-vis` has a slightly higher location error. This is probably due to the fact that the locations on the inflated surface and the geodesic distances between these locations are not perfectly correlated due to local stretching and compression of the cortical surface during the inflation. If the model learns an embedding that can be used to calculate geodesic distances, the location on the inflated surface can only be approximately represented with this embedding.

Influence of loss function on area segmentation Next, the effect of the different loss functions on the fine-tuned segmentation performance is evaluated. Weights from the `siam-vis`, `siam-vis-dist` and `siam-vis-loc` models were used to fine-tune the `BaseNetwork` on the `B01Collection`. Here, `base[siam-vis]` refers to the model fine-tuned from `siam-vis` (trained on the combined loss L), `base[siam-vis-dist]` to the model fine-tuned from `siam-vis-dist` (trained on the distance loss L_dist), and base[siam-vis-loc] to the models fine-tuned from siam-vis-loc (trained on the location loss L_{loc}). The expectation is that fine-tuning from models that are good at localizing the input patches results in good area segmentation models.

The base[siam-vis] model performs better than base[siam-vis-dist], showing that including the location loss L_{loc} improves the applicability of the model for area segmentation (Table 6.2). This is consistent with the expectation, as siam-vis performs better than siam-vis-dist. Interestingly, the model trained only on the location loss base[siam-vis-loc] has the same Dice score as base[siam-vis], but base[siam-vis] has a slightly lower topological misclassification error. Although the difference between base[siam-vis] and base[siam-vis-loc] is not significant in this setting, Section 6.4.3 shows that with the distance loss the model learns more realistic features that encode cytoarchitectonic properties. Thus, the combined distance and location loss is used in the remainder of this section.

6.3.2 Comparison to training from scratch

Section 6.3.1 has shown that the choice of pre-training task is important for the performance of the fine-tuned model. But how much does the fine-tuned model differ from the model trained from scratch? Regarding the quantitative evaluation from Table 6.2, we can see that the pre-trained BaseNetwork (base[siam-vis]) performs 8 Dice score points better than the base model from Section 5.1.1. The base[siam-vis] model additionally outperforms the model trained only on the atlas data (only-atlas) and almost reaches the Dice score of the AtlasAwareNetwork trained from scratch (atlas). It has a lower weighted misclassification error than the two other models. The self-supervised pre-training allows the model to recognize the location of the input patch from the texture in this patch, where the non pre-trained atlas-aware models need to rely on the atlas prior inputs. The low topological misclassification error shows that the model has learned to respect the topology of areas from only the texture input.

Including the atlas prior in the pre-trained model improves the Dice score to 0.74 and misclassification error to 0.174. This is an improvement of 3 Dice score points compared to the atlas model and shows that even though the pre-training allowed the model to learn about the structure of the cortex in different regions of the brain, the atlas prior introduces more information that helps to further increase the performance. Figure 6.5 qualitatively compares the segmentation results of atlas, base[siam-vis], and atlas[siam-vis] on three example sections. The atlas[siam-vis] model produces smoother segmentations with less noise and accurate borders (red markers in Figure 6.5). It is able to identify areas FG1, FG2, FG3, and FG4 on section 1861, and to segment several parts of unlabeled cortex correctly in the cor class (black arrows). As noted in Section 5.3.2, the models including the

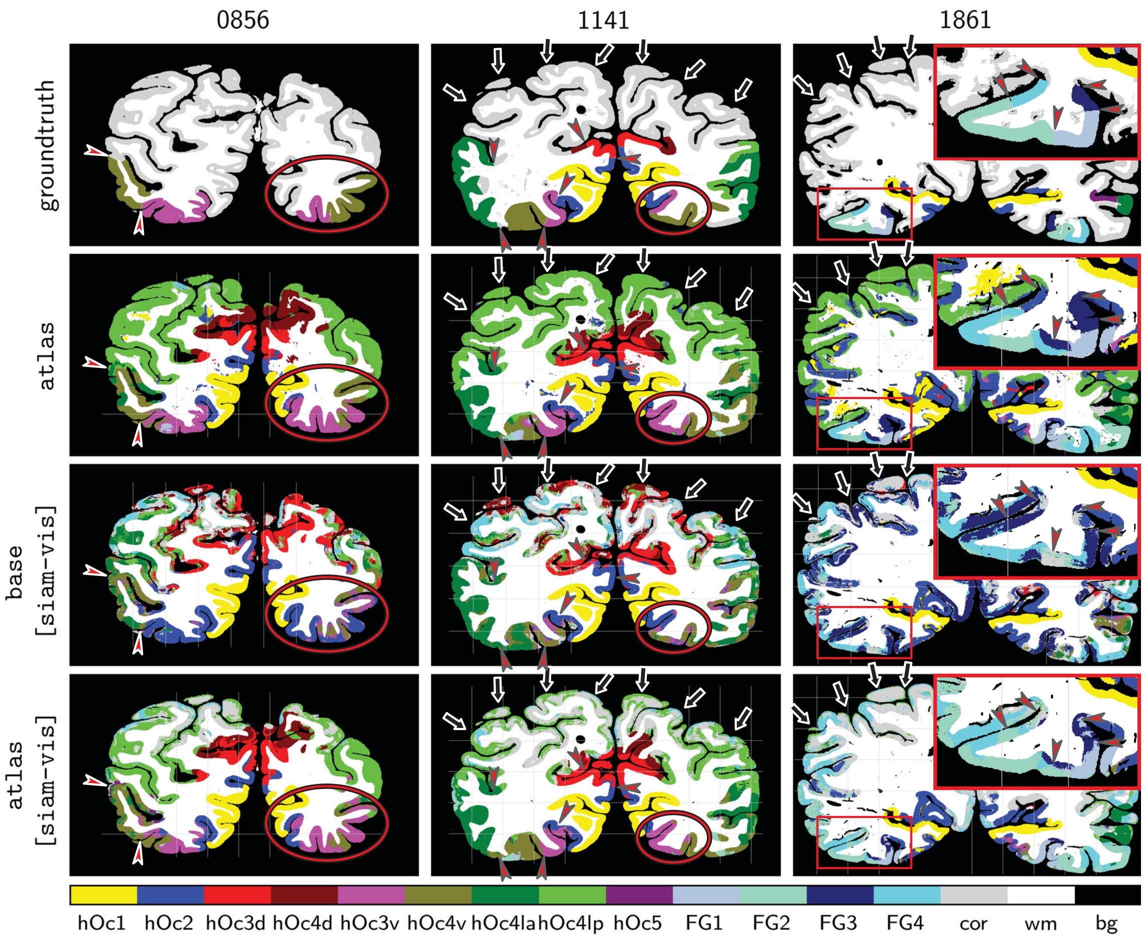


Figure 6.5: Qualitative comparison of models with and without fine-tuning. The atlas[siam-vis] model outperforms the others by producing less noisy segmentations and identifying several borders more precisely (red markers). It correctly predicts the cor class for several regions (black arrows). Models that include the atlas prior are able to identify area hOc3v (red circles).

atlas pior are able to segment area hOc3v (red circles in Figure 6.5), however they are prone to oversegmenting the area.

For more details, Figure 6.6 shows per-label Dice scores of the models mentioned above. With this visualization, subtle differences between the models can be appreciated. Regarding the base and base[siam-vis] models, we can see that the pre-trained BaseNetwork, base[siam-vis], has a higher Dice score than base on labels hOc3d, hOc3v, hOc4v, hOc4la, FG2, and FG3. Areas hOc5 and FG1 could only be predicted by the models using the atlas prior. Those are the two smallest areas in the B01Collection which indicates that there is not enough data to allow the model to learn representations for these areas.

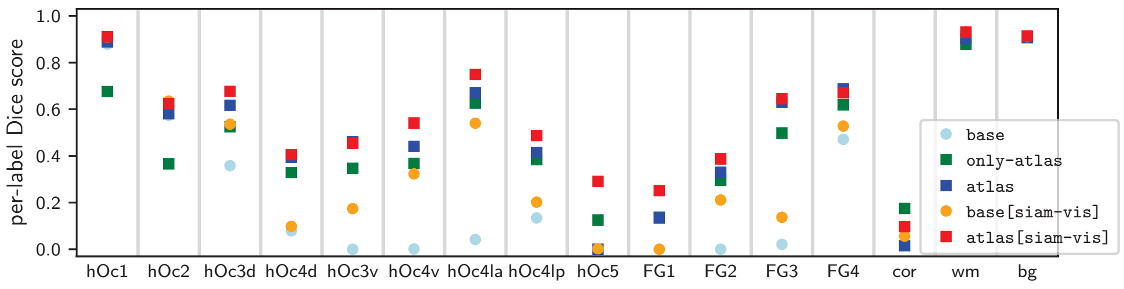


Figure 6.6: Per-label Dice score for segmentation models from Chapters 5 and 6

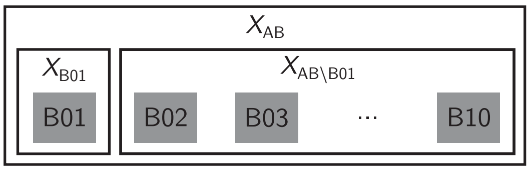


Figure 6.7: Composition of the training datasets used in Section 6.3.3. $\mathcal{X}_{\mathrm{B01}}$ is sampled from B01 (the `B01Collection`), while $\mathcal{X}_{\mathrm{AB}\setminus\mathrm{B01}}$ is sampled from all brain samples in the `AtlasBrainsCollection`, except the B01. $\mathcal{X}_{\mathrm{AB}}$ contains both $\mathcal{X}_{\mathrm{B01}}$ and $\mathcal{X}_{\mathrm{AB}\setminus\mathrm{B01}}$.

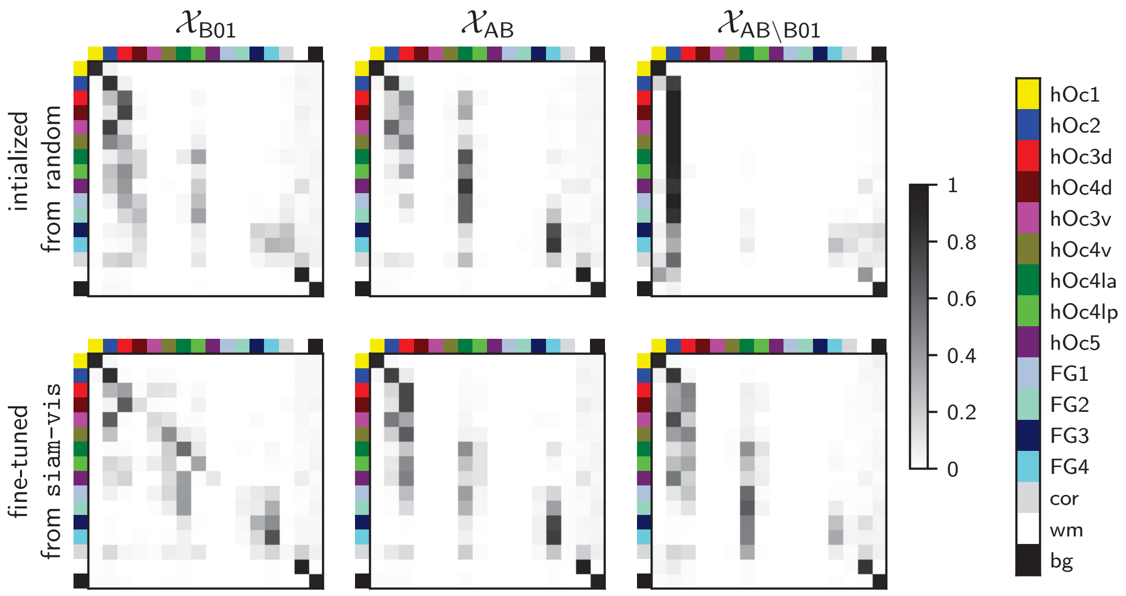
6.3.3 Training on multiple brain volumes

Until now, all segmentation models were trained on one brain volume, B01. Although including more than one brain volume in training the models looks at first glance like an increase in training data, it also means an increase in diversity and complexity of the data. There is a certain set of complex features that characterize each brain area. However, every brain exhibits these features differently due to the large inter-individual variability and individual differences in cortical folding. Additionally, the brains comprising the `AtlasBrainsCollection` have a different visual appearance due to differences in contrast between white matter and cortex and overall gray values (see Section 2.3.1). Thus, training with more brains does not necessarily mean better performance.

This section evaluates the effect of training with multiple brain volumes and shows that pre-training the `BaseNetwork` allows to make predictions on a previously unseen brain volume. For this evaluation, networks are trained on training datasets sampled from the `B01Collection` ($\mathcal{X}_{\mathrm{B01}}$), the `AtlasBrainsCollection` ($\mathcal{X}_{\mathrm{AB}}$), and the `AtlasBrainsCollection` without the `B01Collection` ($\mathcal{X}_{\mathrm{AB}\setminus\mathrm{B01}}$). The datasets $\mathcal{X}_{\mathrm{AB}}$ and $\mathcal{X}_{\mathrm{AB}\setminus\mathrm{B01}}$ are created in a similar fashion as $\mathcal{X}_{\mathrm{B01}}$ (see Section 5.2.2) by sampling from their respective set of sections. Figure 6.7 visualizes the composition of the different datasets used in this section. A randomly initialized `BaseNetwork` and a `BaseNetwork` pre-trained on `siam-vis` are trained on the different datasets and

(a) Dice scores.

	$\mathcal{X}_{\mathrm{B01}}$	$\mathcal{X}_{\mathrm{AB}}$	$\mathcal{X}_{\mathrm{AB}\setminus\mathrm{B01}}$
initialized from random	0.59	0.60	0.40
fine-tuned from `siam-vis`	0.67	0.61	0.56



(b) Confusion matrices (see Section 2.2.5) showing the predicted labels (x-axis) vs the groundtruth labels (y-axis).

Figure 6.8: Comparison of `BaseNetworks` initialized from random weights or fine-tuned from `siam-vis`, and trained on different datasets including multiple brain volumes. The evaluation is done on $\mathcal{X}^{\mathrm{te}}_{\mathrm{B01}}$.

compared on B01, using $\mathcal{X}^{\mathrm{te}}_{\mathrm{B01}}$. For predicting sections of B01, these datasets pose increasingly harder tasks, from training on B01, over training on all brains, to training on all brains except B01. Figure 6.8a shows the Dice score on $\mathcal{X}^{\mathrm{te}}_{\mathrm{B01}}$ of the models trained on the different datasets. Confusion matrices are shown in Figure 6.8b.

First, consider the performance of the randomly initialized models: as expected, including multiple brain volumes in the training (training on $\mathcal{X}_{\mathrm{AB}}$ instead of $\mathcal{X}_{\mathrm{B01}}$) only slightly increases the Dice score due to the increased diversity of the training data. Additionally, we can see that without pre-training the models are not generalizable to previously unseen brain volumes. The model trained on $\mathcal{X}_{\mathrm{AB}\setminus\mathrm{B01}}$ not including the B01 can only predict area `hOc1` when evaluated on the B01.

Second, we look at the performance of the pre-trained networks. When including more brains in the training, the performance of the pre-trained models on the B01 test split drops compared to training only on the B01. This shows again that the

AtlasBrainsCollection contains a diverse mixture of different brains and areas and too few examples per brain and area. Although the self supervised pre-training provides good initial features for the model, the model cannot leverage this data to learn better representations for the B01. However, self-supervised pre-training does allow to train models that are transferable to previously unseen brains: training a pre-trained network on $\mathcal{X}_{\mathrm{AB}\setminus\mathrm{B01}}$ results in acceptable performance on the B01. Although still worse than a not pre-trained network trained on $\mathcal{X}_{\mathrm{B01}}$, this model can predict hOc1, hOc2, hOc4la and FG3.

Thus, pre-training on the distance and location task enables to learn models that are partially transferable to previously unseen brain volumes. Overall, training on several brain volumes does not improve results due to the large inter-individual differences that overshadow the differences between different brain areas. It is therefore doubtful, whether it is possible to learn a general model that provides brain area segmentations for all brain volumes. Rather, it seems like specialized CNN models for each brain are necessary in order to get good results for the segmentation. Chapter 7 takes one step further and presents an approach for training specialized models on a pair of sections and one brain area to enable very precise segmentation of one brain area in several spatially close sections.

6.4 Analysis of the self-supervised model

In Section 6.3, we have seen that pre-training on the self-supervised distance and location regression task improves the performance on the area segmentation task. This shows that the self-supervised model learns suitable internal representations that can be leveraged during fine-tuning. But what exactly do these features represent? Do they encode cytoarchitectonic properties? This section investigates these questions by analyzing the learned embedding of the high-resolution input patches X to a feature vector $g(X)$.

To evaluate the learned feature representations on the entire brain, a self-supervised SiameseNetwork is trained on all sections of the BigBrainCollection. To get a more independent test set, training is done on either the left or the right hemisphere and the evaluation is done on the other hemisphere. In Section 6.4.1 we first confirm that the model can generalize to another hemisphere. Then, Section 6.4.2 shows how the features $g(X)$ correlate with measures such as cortical thickness and average gray value to get an indication of the information that is present in $g(X)$. The learned features $g(X)$ are qualitatively analyzed in Section 6.4.3 by visualizing them on the brain surface. These visualizations indicate that the feature vectors encode cytoarchitectonic properties and that it is possible to create plausible parcellations based on the vectors. To gather further evidence for this hypothesis, Section 6.4.4 ana-

Table 6.4: Mean error $\mathrm{err}_{\mathrm{loc}}$ in millimeters between predicted location $\hat{q}$ and true location q of models trained and evaluated on different hemispheres. Errors are lowest for models trained and evaluated on the same hemisphere, and higher if evaluating on the opposite hemisphere.

		evaluate on	
		left	right
train on	left	21.64	26.79
	right	26.42	22.07
	both	25.26	25.54

lyzes the features on 2D sections and shows their clustering respects borders between cytoarchitectonic areas. The evaluations presented in this section were inspired by the analysis of the feature vectors published in Spitzer, Amunts, et al. (2018). Section 6.4.3 contains a new version of the hierarchical clustering presented in Spitzer, Amunts, et al. (2018).

6.4.1 Inter-hemisphere generalizability

To draw general conclusions from an analysis of the internal representations of the self-supervised model, an independent test set is needed. In absence of a second dense brain volume, models were trained on one hemisphere of the BigBrain and tested on the other. This section confirms that features learned on one hemisphere are applicable to the second hemisphere. Table 6.4 shows the error $\mathrm{err}_{\mathrm{loc}}$ (Equation (6.5)) in millimeters between predicted location $\hat{q}$ and the true location q of the input patch X on the inflated surface for models trained on the left, right, and both hemispheres. Models were trained with 200 000 random patch pairs from the corresponding hemisphere(s) (i.e., on $\mathcal{X}^{\mathrm{tr}}_{\mathrm{BBl}}$ and $\mathcal{X}^{\mathrm{tr}}_{\mathrm{BBr}}$). Evaluation was done on 150 000 patches from each hemisphere (i.e., on $\mathcal{X}^{\mathrm{te}}_{\mathrm{BBl}}$ and $\mathcal{X}^{\mathrm{te}}_{\mathrm{BBr}}$).

As expected, the error is lowest for models that were trained and evaluated on the same hemisphere ($\mathrm{err}_{\mathrm{loc}}$ is 21.64 and 22.07 for left and right hemispheres, respectively). Models trained on both hemispheres perform a little worse with $\mathrm{err}_{\mathrm{loc}} = 25.26$ on the left and $\mathrm{err}_{\mathrm{loc}} = 25.54$ on the right hemisphere. Compared to this, the models trained on one hemisphere and evaluated on the other hemisphere perform only marginally worse ($\mathrm{err}_{\mathrm{loc}}$ is 26.79 and 26.42 for models trained on left and evaluated on right and vice-versa). There is no significant difference between the hemispheres. This shows that features are not specific to one hemisphere, but that learning location on one hemisphere enables the model to accurately predict location on the second hemisphere.

6.4.2 Relationship between learned features and cortical properties

In order to learn the distance and location regression task, the Siamese network model learns to calculate features $g(X)$ that are similar for input patches that are located close together, and different for input patches that are located far apart. The hypothesis is that the model uses cytoarchitectonic properties of the input patches X to calculate $g(X)$. However, the model may detect other signals to solve the distance and regression task, such as 1. cortical thickness, 2. average gray value of the patch, 3. absolute angle between sectioning plane and cortical surface (called obliqueness in the following), and 4. local curvature of the cortex. Cortical thickness is a morphological measure that is systematically related to cytoarchitecture (Wagstyl, Ronan, et al., 2015). In addition, the average gray value of cell-body stained tissue is a substitute measure for cell density. In contrast, obliqueness and local curvature are not related to cytoarchitecture. The aim of this section is to verify whether cytoarchitectonic properties are indeed the main signal that is encoded in $g(X)$, and thereby confirm that the auxiliary task is appropriate for the purpose of pre-training an area segmentation model. For this aim this section assesses in the following, whether the feature vector $g(X)$ can be used to predict the above signals by calculating a linear regression from $g(X)$ to each scalar value. Following the hypothesis, cortical thickness and average gray value should be predictable by $g(X)$, and obliqueness and local curvature not.

The cortical thickness s_{thick} was calculated by following the gradient of the Laplace potential field from the gray/white matter border to the pial surface and measuring the distance while accounting for local effects of folding using the model proposed by Bok (1929).[1] The gray value s_{gray} was calculated as the mean gray value of the high-resolution input patch X. The mean curvature s_{curv} was calculated for every point on the midline between gray/white matter border and pial surface as the amount of deviation of the curve from the tangent line (Goldman, 2005, Equation 4.2). The obliqueness s_{oblique} was calculated as the absolute angle between the sectioning plane and the mean of the normal of the pial surface. Figure 6.9 shows the calculated measures on the brain surface.

To evaluate whether the learned feature vector contains these signals, a linear regression from the feature vector to the signal value is calculated resulting in a predicted signal value $\hat{s} = w^T g(X) + b$, with $w \in \mathbb{R}^{32}$, and $b \in \mathbb{R}$. The goodness-of-fit was measured as the ratio between the prediction error and the deviation of the

[1] Thanks to Konrad Wagstyl (University of Cambridge) for calculating the cortical thickness.

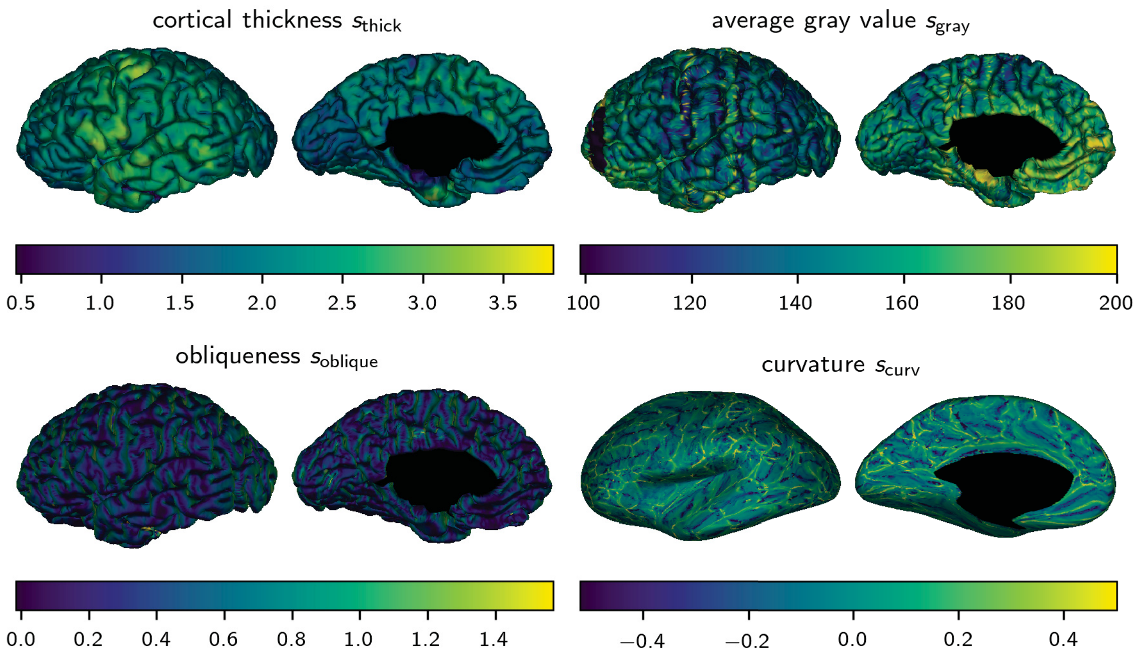

cortical thickness s_{thick}
0.5
1.0
1.5
2.0
2.5
3.0
3.5
average gray value s_{gray}
100
120
140
160
180
200
obliqueness s_{oblique}
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
curvature s_{curv}
−0.4
−0.2
0.0
0.2
0.4

Figure 6.9: Different signals plotted on the brain surface. The cortical thickness is highest in the motor cortex and is reduced in the primary sensory areas. The average gray value shows staining differences between the individual sections in anterior-posterior direction. The obliqueness, i.e., the angle between the sectioning plane and the cortical surface, changes depending on the normal of the cortical surface. The curvature is correlated with the outward and inward folds of the cortex. It is plotted on the inflated white matter surface to show the areas of high negative curvature in the bottom of the gyri.

groundtruth values s from their mean $\bar{s}$:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(s_i - \hat{s}_i)^2}{\sum_{i=1}^{n}(s_i - \bar{s})^2}$$

with $n = |\mathcal{X}^{\text{tr}}|$ the size of the test dataset. A model was trained on patches from the right hemisphere ($\mathcal{X}^{\text{tr}}_{\text{BBr}}$) and evaluated on patches from the left hemisphere ($\mathcal{X}^{\text{te}}_{\text{BBl}}$). Figure 6.10 shows scatter plots of the predicted signal values $\hat{s}$ and the true signal values s.

Cortical thickness is systematically related to cytoarchitecture (Wagstyl, Ronan, et al., 2015). Thus, we can expect that if the model learned to extract cytoarchitectonic properties from the cortical patches, it should be possible to predict cortical thickness from the feature vectors. The linear regression confirms this. 21% of the variance in cortical thickness can be explained by $g(X)$.

Average gray values of the input patches X can be seen as a measure of cell density, which is a cytoarchitectonic property. However, in the `BigBrainCollection` the

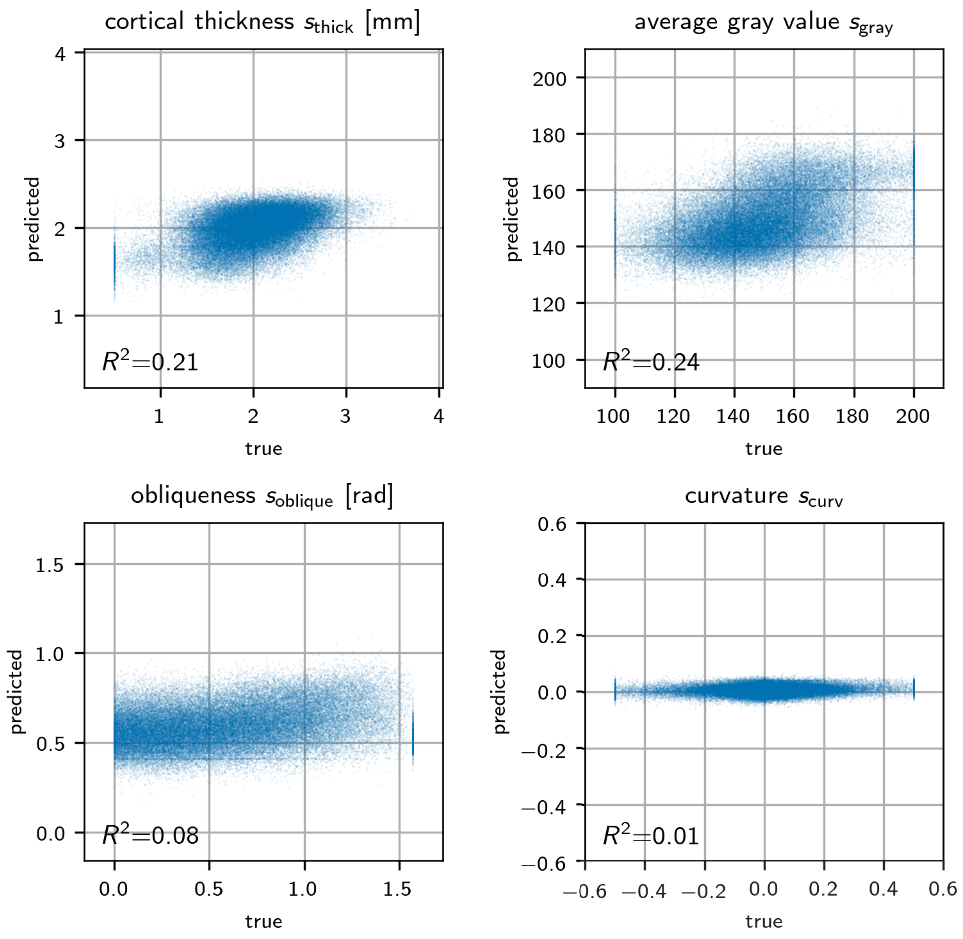


Figure 6.10: Results of a linear regression from the learned feature representation $g(X)$ of cortical patches X to different signals. Cortical thickness and average gray value can be predicted from the features, obliqueness and curvature not.

average gray value of the input patches is highly dependent on the individual section due to the differences in staining intensity (cf. Section 2.1.1) as can be seen in the vertical stripes visible in Figure 6.9. Besides this dependence on the section, the average gray values form a smooth gradient in the medial-lateral direction, with more medial patches being lighter. Due to this correlation, it is clear that the average gray value will be encoded by the feature vector, because the self-supervised model was successfully trained to predict the medial-lateral coordinate. The linear regression shows that gray values can be predicted from the feature vectors with $R^2 = 0.24$.

Obliqueness, i.e., the absolute angle between the sectioning plane and the cortical surface, has an influence on the cell densities of the cortical laminae. Therefore, following the hypothesis that the neural network model learns to represent cytoarchitectonic properties, the model needs to consider the obliqueness of the input patch

X in the early layers of the network. However, in the later layers of the network, the model should abstract from the concrete obliqueness and produce similar feature vectors for inputs from the same area but with different sectioning angles. As can be seen from the evaluation, the obliqueness can not be predicted from the feature vectors ($R^2 = 0.08$)

In principle, *curvature* is not correlated with cytoarchitectonic properties. Curvature has an impact on the widths of cortical laminae, but borders between areas usually do not coincide with such anatomic landmarks (Amunts, Lepage, et al., 2013). Interestingly, the linear regression could not learn to predict curvature from the feature vector ($R^2 = 0.01$).

These evaluations show that a linear combination of the learned features can be used to predict cortical thickness and average gray values. In contrast, the obliqueness and the curvature, although prominently visible in the inputs, are not predictable by the feature representation. These findings are consistent with the hypothesis that the model learned to represent cytoarchitecture in the feature vectors.

6.4.3 3D analysis of feature vectors

Feature vectors of 7300 evenly distributed patches from the left hemisphere were extracted using the self-supervised model trained on $\mathcal{X}_{\mathrm{BBr}}$. In order to further strengthen the claim that the learned feature vectors represent cytoarchitecture, we will analyze their principal components and possible clusterings.

Principal Component Analysis The six largest principal components of the feature vectors were calculated using principal component analysis (PCA) (Abdi and Williams, 2010) and plotted on the inflated white matter brain surface (Figure 6.11).

The first three components seem to encode mostly posterior-anterior, ventral-dorsal, and medial-lateral gradients respectively. It is expected that these gradients are strongly represented in the feature vector, because the model was trained with a combination of distance and location loss and thus has to represent the spatial coordinates in the feature vector. The next three components are especially interesting, because they highlight the motor cortex on one extreme, and the primary areas (somatosensory, visual, auditory) on the other. This is consistent with the cytoarchitectonic gradients mentioned in (Huntenburg et al., 2017) and shows that the model learned more than just to represent the location of the input patch.

Components such as those last three, which represent cytoarchitectonic gradients, only occur in models trained with the distance loss. For comparison, Figure 6.12 shows the first 6 components of a model trained only with the location loss. The first three components represent the three coordinate axes and explain all the variance. The remaining components only contain noise. This confirms the hypothesis

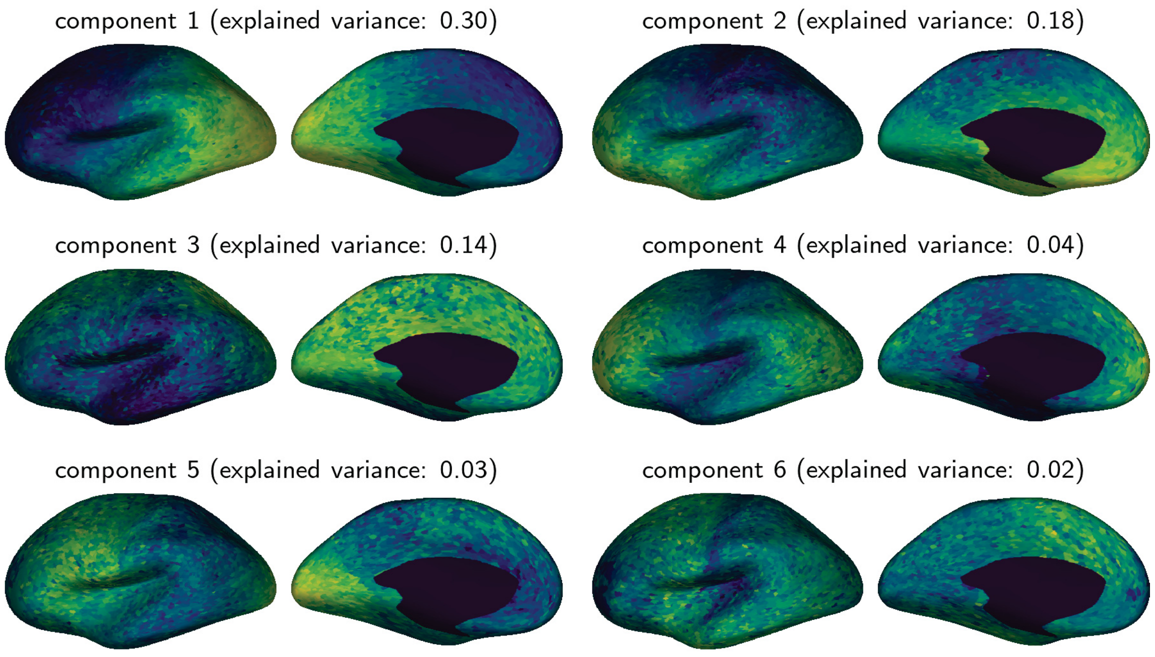

component 1 (explained variance: 0.30)
component 2 (explained variance: 0.18)
component 3 (explained variance: 0.14)
component 4 (explained variance: 0.04)
component 5 (explained variance: 0.03)
component 6 (explained variance: 0.02)

Figure 6.11: The six largest principal components of the learned feature representation plotted on the inflated left brain surface. The first components represent the spatial directions. The last three components represent other signals that are similar to cytoarchitectonic gradients.

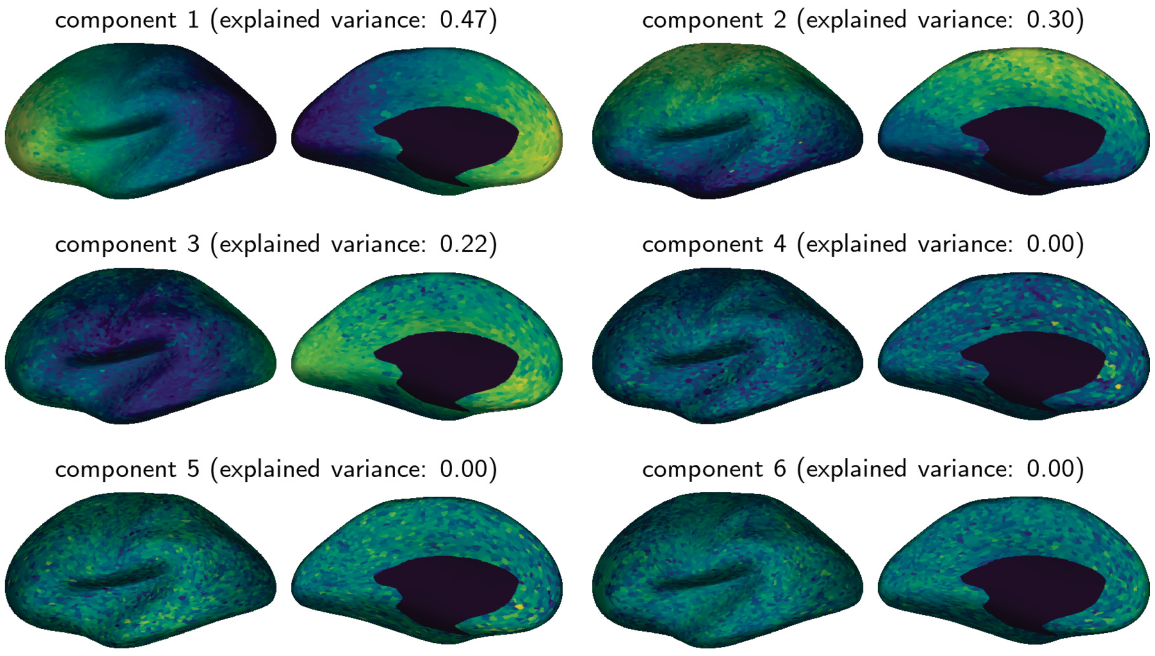

component 1 (explained variance: 0.47)
component 2 (explained variance: 0.30)
component 3 (explained variance: 0.22)
component 4 (explained variance: 0.00)
component 5 (explained variance: 0.00)
component 6 (explained variance: 0.00)

Figure 6.12: For comparison to Figure 6.11, the six largest principal components of the feature representation learned by a model trained only on the location loss L_{loc}. There are no components representing cytoarchitectonic gradients.

discussed in Section 6.1.1 that the location task alone is not precise enough to let the model learn cytoarchitectonically relevant features. The distance loss is vital to learn features that represent fine-grained differences between cortical patches.

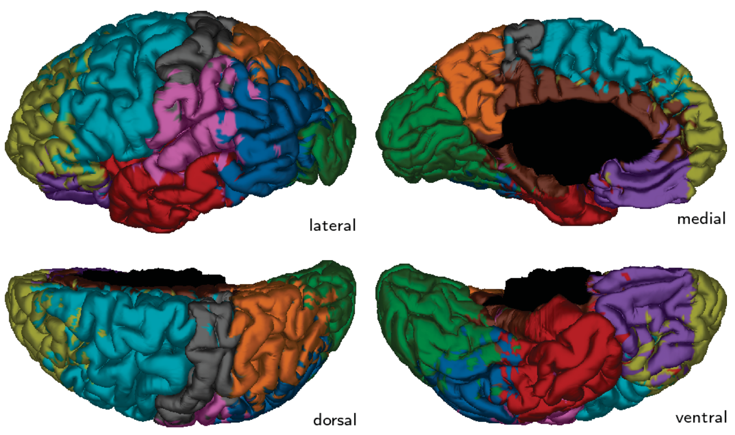
Hierarchical clustering The feature vectors were clustered using Ward's method, minimizing within-cluster variance using the squared Euclidean distance function as metric (Ward Jr, 1963) and thresholded at 10 clusters. The resulting clusters are slightly different than the ones reported in Spitzer, Amunts, et al. (2018), because they were calculated on features extracted using a model that was trained on a slightly different dataset. However, the results and conclusions presented here are consistent.

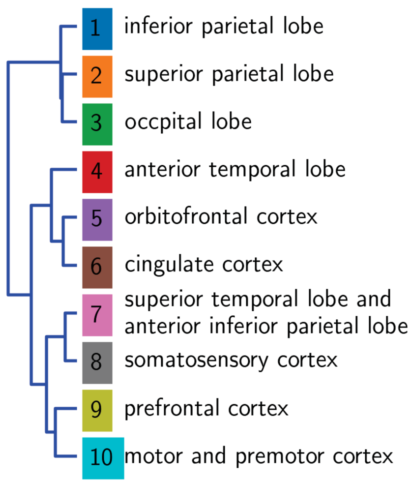
The clustering is purely based on the feature vectors $g(X)$ of the trained network. Most importantly, the locations of the feature vectors $g(X)$ were not included into the clustering. Figure 6.13a shows that the clusters form smooth regions on the brain surface. The clustered regions roughly correspond to different anatomical regions of the brain (Figure 6.13b). For comparison, Figure 6.14 shows selected probabilistic maps from the JuBrain atlas.

For example, clusters 8 and 10 have their boundary along the central sulcus. This central landmark separates the motor cortex (column E) from the somatosensory cortex (column C) which have very different cytoarchitecture. The motor cortex is agranular and contains giant pyramidal cells, while the somatosensory is granular with smaller cells and a more distinct laminar pattern. Other clusters also correspond to anatomical regions: the visual cortex (column A) corresponds well to cluster 3, the orbitofrontal cortex (column B) has the same extent as cluster 5, and the prefrontal cortex (column D) is represented by cluster 9. This shows that feature vectors form clusters based on cytoarchitectonic similarity and not only based on location. This is strong evidence that the feature vectors contain cytoarchitectonic attributes.

6.4.4 2D analysis of feature vectors

In Section 6.4.3 we gained an overview over the differences in feature vectors at a whole brain scale. This section analyzes individual cytoarchitectonic borders and shows that several cluster borders correspond with cytoarchitectonic borders when calculated on a per-section basis. In detail, the borders between between primary visual cortex (`hOc1`) and secondary visual cortex (`hOc2`) and the borders between motor, premotor and somatosensory cortex are analyzed. The cytoarchitecture of these cortical areas is very different from each other. In order to assess the difference of feature vectors over these borders, feature vectors were extracted from the cortex of each section. A hierarchical clustering was calculated on a per-hemisphere and per-section basis. Figure 6.15 shows the clustering results evaluated at 10 clusters,





(a)

(b)

Figure 6.13: (a) Hierarchical clustering of feature vectors extracted from the left hemisphere using a model trained on the right hemisphere. (b) The clusters roughly correspond to different cortical regions of the brain (compare to JuBrain areas in Figure 6.14).

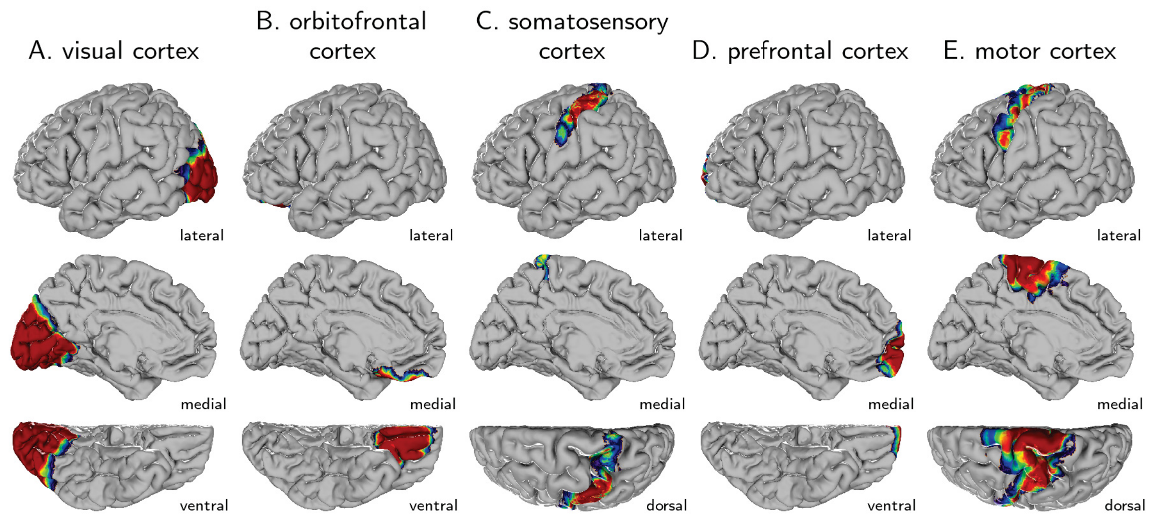
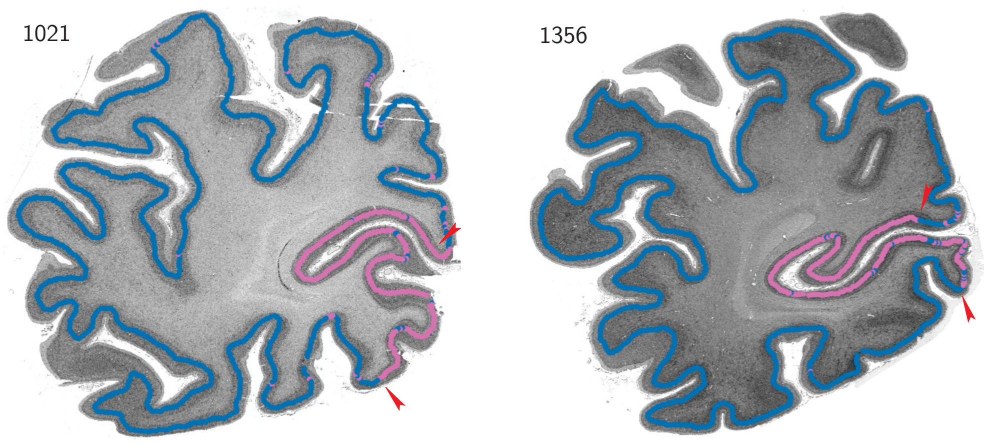
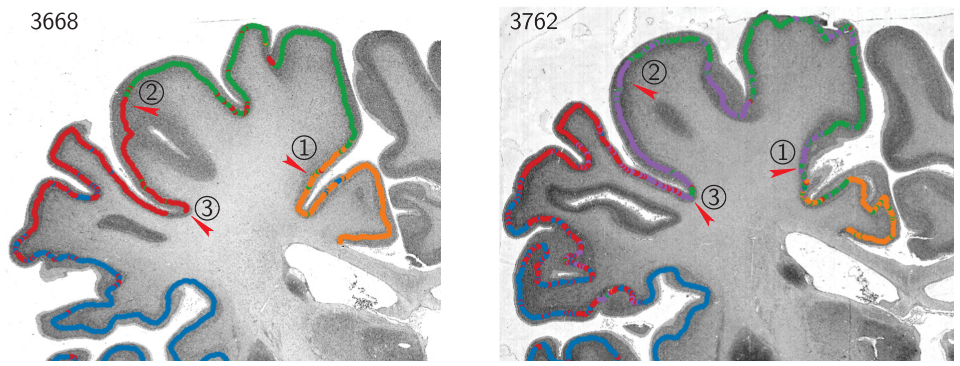


Figure 6.14: Probabilistic maps of selected brain areas from the JuBrain atlas shown on the left hemisphere. The images are taken from the JuBrain Cytoarchitectonic Atlas Viewer (*JuBrain Atlas Viewer* 2014–2019). Column A shows areas of the visual cortex: `hOc1`, `hOc2`, `hOc3v`, `hOc4v`, `hOc3d`, `hOc4d`, `hOc5`. Column B shows areas of the orbitofrontal cortex: `Fo2`, `Fo3`. Column C shows areas of the somatosensory cortex: `1`, `3a`, `3b`. Column D shows areas of the prefrontal cortex: `Fp1`, `Fp2`. Column E shows areas of the premotor and motor cortex: `6`, `4a`, `4p`.



(a) Sections of the visual system. The `hOc1`/`hOc2` borders are marked.



(b) Sections of the motor and somatosensory cortex. Border 1 separates the cingulate cortex from the premotor cortex. Border 2 separates premotor and motor cortex. Finally, border 3 separates the motor cortex from the primary somatosensory cortex.

Figure 6.15: Results of clustering feature vectors on a per-section basis. As a reference, borders between cytoarchitectonic areas marked by a neuroscientific expert are denoted by the red arrows.[2] Each feature vector is plotted as a dot on its location on the cortex and colored according to its cluster value at cluster depth 8. The blue points represent several clusters.

on the original section.

Again, clusters correspond to cytoarchitectonic areas. In both sections showing the visual cortex, the primary visual cortex forms its own cluster (pink cluster in Figure 6.15a). The actual cytoarchitectonic borders (red markers) correspond well with the cluster borders. On the sections from the motor cortex (Figure 6.15b) we can see that the border between cingulate areas (orange cluster) and the premotor cortex (green cluster) corresponds well with the border between the orange and green

[2]Thanks to Kai Kiwitz (Heinrich-Heine-University Düsseldorf) for delineating the borders between the cytoarchitectonic areas and doing initial analyzes of the clusters.

clusters. At the border between the premotor and the primary motor cortex the green cluster ends in the two example sections. However, the border between primary motor cortex and somatosensory cortex is only represented in the clustering on section 3762 (purple and red clusters). These results show that without explicitly training on the task of area segmentation, the model has learned an internal feature representation that allows to distinguish primary visual cortex and motor cortex from surrounding areas. This is a strong indicator that training on the proposed auxiliary task indeed drives the model to learn to encode cytoarchitectonic properties.

6.5 Conclusion

The self-supervised distance and location regression task allows to leverage large quantities of unlabeled high-resolution histological sections of one human brain volume to learn feature representations that are useful for characterizing cytoarchitecture. The evaluations have shown this by 1. applying the learned representations to the area segmentation task and showing superior results compared to the base network, and by 2. analyzing the representations directly and showing that they represent cytoarchitectonic properties.

Pre-training models on the distance and location regression task significantly improves the results of the area segmentation (Section 6.3.2). Although the pre-training requires a brain volume with an existing 3D reconstruction, the fine-tuning for area segmentation does not require a registration. This allows to apply the knowledge gained from the spatial information contained in the 3D reconstruction to datasets that are not 3D reconstructed. With this pre-training it is possible to train models on one set of brain volumes, and apply the trained model to a previously unseen brain volume (Section 6.3.3). The performance of the fine-tuned models depends on the similarity of the pre-training task to the areas that should be segmented (Section 6.3.1).

The distance and location regression task allows to learn models on one hemisphere of the brain and apply them to the other (Section 6.4.1). A clustering of the feature representations $g(X)$ shows that clusters roughly represent different anatomically and functionally relevant regions of the brain. A 2D analysis of the clusters shows that several cluster borders directly correspond to areal borders. Principal component analysis shows that the features encode cytoarchitectonic gradients described in literature. To conclude, the self-supervised task is suitable for learning relevant cytoarchitectonic characteristics in the high-resolution input patches and it can be used to improve the area segmentation described in Chapter 5.

From these results, several avenues for future work open up. In a recent contribution, Beul et al. (2017) found that cytoarchitectonic similarity represented by neuron density is a good predictor for structural connectivity. The feature representation $g(X)$ could be used as a better estimate of cytoarchitecture to research the relationship between cytoarchitecture and connectivity. In general, the distance and location regression task is applicable to other types of brain volumes and modalities. Future work will evaluate if this task can be used a a general-purpose feature learning algorithm that extracts cytoarchitectonic features from lower resolution histological or MRI brain volumes. Instead of extracting features based on cell densities (cytoarchitecture) it could also be applied to diffusion MRI or polarized light imaging data to extract features based on fiber tracts.

7 Application: Deep label propagation between sections

The previous chapters focused on learning a general-purpose model for segmenting a set of cortical areas in one or more brains. To solve this challenging task, a model that combines high-resolution texture information with an atlas prior was developed. The results are very encouraging. Nevertheless, the reliability of the model does not yet permit to perform fully automated brain mapping with sufficiently robust results in all brain areas. Getting this precision would require more training data and more time to develop specialized learning algorithms and architectures.

Despite these remaining challenges, the current results have a direct practical relevance as will be shown in this chapter. The main idea is to use the presented CNN models to support the neuroanatomists in the brain mapping workflow in an interactive setup. Instead of trying to solve the general brain area segmentation problem, this chapter presents a specialized multi-scale CNN model that is trained with two spatially close sections and annotations of one cortical area. The model can precisely segment this cortical area on the sections in between the training sections. Essentially, the model thus propagates the labels from one section to neighboring sections. In the following, this task is referred to as *deep label propagation* (Figure 7.1).

The deep label propagation complements the current semi-automatic brain area mapping workflow as summarized in Section 2.1.3. For mapping a new brain area, neuroanatomists currently locate the area in approximately every 60th section, and validate its borders using the method described by Schleicher, Amunts, et al. (1999). The proposed multi-scale CNN model provides high-resolution segmentations between annotated sections that support and speed up the mapping process. They are consistent over different sections and allow to create novel, 20 µm resolution 3D reconstructions of individual brain areas. The deep label propagation does not require a registration between the individual sections; on the contrary, the resulting predictions could be used in the future to drive the registration.

This chapter results from work done by Christian Schiffer during his masters studies under the supervision of Hannah Spitzer. The development of the data sampling and the hyper-parameter tuning for the multi-scale CNN model was done by Christian Schiffer. Hannah Spitzer calculated the 3D reconstructions of `hOc1` and `hOc2` described in Section 7.4.

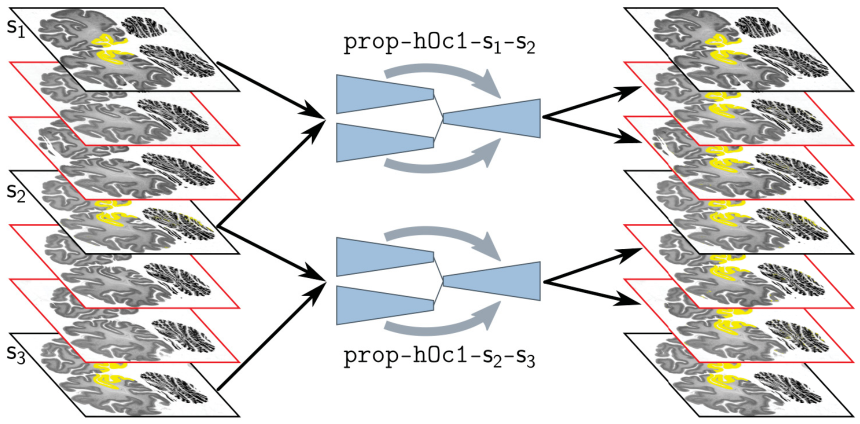


Figure 7.1: Deep label propagation workflow. A brain area is mapped in several sections (black border). For each pair of neighboring sections s_1 and s_2 with annotations of area a, a neural network model prop-a-s_1-s_2 is trained on s_1 and s_2 to “fill the gap" (red border) between the annotated sections.

Section 7.1 introduces the deep label propagation task and workflow to create dense segmentations of one brain area in one brain volume and discusses related work. The multi-scale network architecture and the training procedure used for the deep label propagation are described in Section 7.2. Section 7.3 shows that the low-resolution context input is beneficial for the performance and that the resulting segmentations are precise and spatially consistent. Then, 3D reconstructions of hOc1 and hOc2 that were calculated based on the propagated labels are shown in Section 7.4. Finally, Section 7.5 discusses the impact and the limitations of the deep label propagation task.

7.1 Deep label propagation workflow

The subject of this chapter is the creation of dense, precise, and high-resolution segmentations of one brain area in one brain. With this aim, the brain area is annotated on every n-th section of one brain with n in order of hundred sections. Using patches samples from a pair of annotated sections s_1 and s_2, with $s_1 < s_2$, a deep label propagation model prop-a-s_1-s_2 is trained to identify the brain area a in sections $s = s_1 + 1, ..., s_2 - 1$ in between the training sections (Figure 7.1; Section 7.2 provides details about the model and the training procedure). This allows prop-a-s_1-s_2 to leverage the similarity between the training and evaluation sections. To create a dense segmentation of area a, for a set of 1200 sections with annotations on every 120th section, 10 models (prop-a-1-121, prop-a-121-241, ..., prop-a-1001-1201) have to be trained on all pairs of consecutive annotated sections.

Related work Differing from atlas registration approaches for label propagation (Rohlfing et al., 2005), this approach does not depend on the quality of the registration and takes into account the tissue values for the segmentation. In fact, the deep label propagation workflow will work without any slice to slice registration, relying only on the section numbers to identify gaps and ordering. Recently, Atzeni et al. (2018) combined multi atlas fusion with a CNN to label histological sections at a structure level. They implemented a generalized expectation-maximization (EM) algorithm that is executed for every test section s using labels on neighboring training sections s_1 and s_2, $s_1 < s < s_2$. First, the segmentation on the test section s is estimated by combining the predictions of the CNN with the registered labels from the nearest labeled sections s_1 and s_2 using multi-atlas fusion (E step). Then, the CNN is fine-tuned using the labels on s_1 and s_2 and the estimated labels on s from the E step (M step). Their approach yields promising results on the prediction of large scale structure labels such as “cortex", although the difference between their proposed EM algorithm and a simple product between the registered labels and the predictions of the CNN is small.

However, it has several drawbacks: Each M-step of the EM algorithm signifies training a CNN. In the case of high-resolution area segmentation this is too computationally expensive. In addition, if for every test section a separate model is trained, this workflow is intractable to propagate labels to a large amount of test sections. In contrast to the here presented deep label propagation, for the approach of Atzeni et al. (2018) a slice-to-slice registration is mandatory. Nonetheless, including the test sections in the training with an estimated label has the advantage that artifacts or other variation specific to this test section is seen by the network during training which allows it to adapt to it. For the deep label propagation this could mean to first train a model on only the test sections (like the models presented in the following) and then train a second model using the predictions of the first model as groundtruth. Such a cascaded training procedure seems like a promising premise for future work on deep label propagation.

7.2 Multi-scale model for label propagation

For the deep label propagation the general area segmentation task is reduced to the segmentation of one area in a spatially constrained set of sections from one brain. The main difference between this task and the general area segmentation task is the low variance of training and testing data in the deep label propagation task. Due to the closeness of the training and testing sections, the main source of variance are per-section artifacts and staining differences. In contrast to the training data for the general area segmentation task, the morphology of the surface changes only

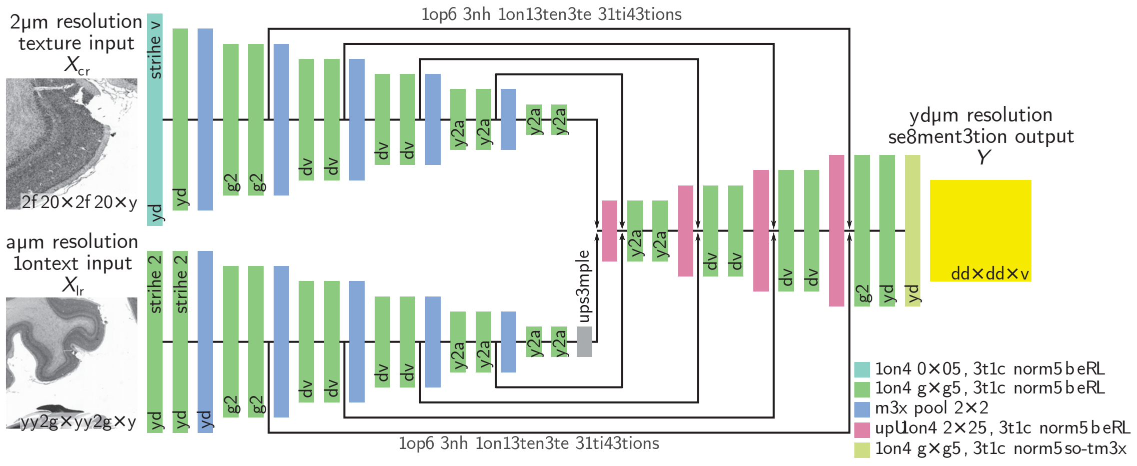


Figure 7.2: `MultiScaleNetwork` for deep label propagation. The model gets a 2 µm high-resolution texture input and a 8 µm context input. It is trained on and applied to spatially close sections. Context information is used to localize cortical areas based on anatomical landmarks.

slightly between training and testing sections. Curvature and obliqueness are almost identical between spatially close sections. This allows us to use the local curvature of the cortex to localize a brain area in a test section. Note that for the general segmentation task, this is not possible, because the curvature of the cortex is highly variable when considering a larger set of training sections.

The model introduced in this section uses the closeness of training and testing data by leveraging low-resolution context information in combination with high-resolution texture information. To deal with the small amount of training data, it is fine-tuned from the general area segmentation model from Section 6.3. Section 7.2.1 presents the multi-scale network architecture. Then, the generation of the training datasets for a training on two sections and one brain area is described in Section 7.2.2. Finally, Section 7.2.3 gives details about the training procedure for training a multi-scale model on deep label propagation.

7.2.1 Multi-scale network architecture

Similar to the `AtlasAwareNetwork` from Section 5.1.2, the `MultiScaleNetwork` uses a U-Net architecture with two input streams to process the high-resolution texture input and the low-resolution context information. To allow transfer learning from the area segmentation models, the high-resolution texture input is processed in a downsampling path which has the same architecture as the downsampling path of the `BaseNetwork` (Section 5.1.1). The context input is processed in a second downsampling path whose architecture is similar to the atlas prior stream of the

AtlasAwareNetwork. The two downsampling streams are concatenated before the upsampling patch. For this, the activations of the context stream are upsampled and cropped to fit to the resolution of the high-resolution texture stream. Figure 7.2 shows the network architecture.

The network takes a 2 µm resolution texture input $X_{\mathrm{hr}} \in \mathbb{R}^{2025\times 2025}$ and a 16 µm resolution context input $X_{\mathrm{lr}} \in \mathbb{R}^{1123\times 1123}$ which corresponds to a 8984×8984 pixel region at 2 µm resolution. Thus, the context input shows a region that is 16 times larger than the region that the texture input shows. For these inputs, the model outputs corresponding segmentations $Y \in \mathbb{R}^{68\times 68}$ with resolution 16 µm. It has 2.44×10^6 trainable parameters. In the following, this architecture is called MultiScaleNetwork.

7.2.2 Dataset generation

The MultiScaleNetwork models presented in the following are trained on sections from the BigBrainCollection, as this collection contains a dense set of histological sections from one brain volume. Annotations of areas hOc1 and hOc2 were made on every 60th section and models were trained on sections with distance 120, leaving an annotated validation section in between the training sections. As motivated in Section 5.2.1, an inhomogeneous background class is not desirable. Thus, the annotated sections are enhanced with the gray/white matter segmentation described in Section 2.3.3. Consequently, the MultiScaleNetworks are trained to segment input patches in the four classes “brain area", “cortex", “white matter", and “background". In the evaluation presented in the following, models were either trained on labels hOc1, cor, wm, bg or on labels hOc2, cor, wm, bg.

Let $\mathcal{X}_{\mathtt{a},s_1,s_2}$ be the training dataset for model prop-a-s_1-s_2 with elements $X = (X_{\mathrm{hr}} \in \mathbb{R}^{2025\times 2025}, X_{\mathrm{lr}} \in \mathbb{R}^{1123\times 1123}) \in \mathcal{X}_{\mathtt{a},s_1,s_2}$. It is created by randomly sampling high-resolution texture and context patches from the two training sections s_1 and s_2 of the BigBrainCollection. Every location (s, i, j) on section s was given a sampling probability $p_{s,i,j}$ and training patches were sampled according to this probability (see Figure 7.3).

To define the sampling probabilities, first a sampling area $\mathcal{S}$ consisting of those pixels with a nonzero sampling probability was selected. Let $\mathcal{G}_{\mathtt{k}}(s)$ be the set of coordinates on section s that are annotated with area k, with $\mathtt{k} \in \{\mathtt{a}, \mathtt{bg}, \mathtt{cor}, \mathtt{wm}\}$. For a radius r and section s let the sampling area $\mathcal{S}_r(s)$ consist of those pixels whose shortest Euclidean distance to a pixel annotated with the brain area a is smaller than r, i.e.,

$$\mathcal{S}_r(s) = \{(i,j) |\, \min \operatorname{dist}\big((i,j)\,, \mathcal{G}_{\mathtt{a}}(s)\big) < r\}\,. \tag{7.1}$$

Then, the probability of sampling a location inside the sampling area is set to the

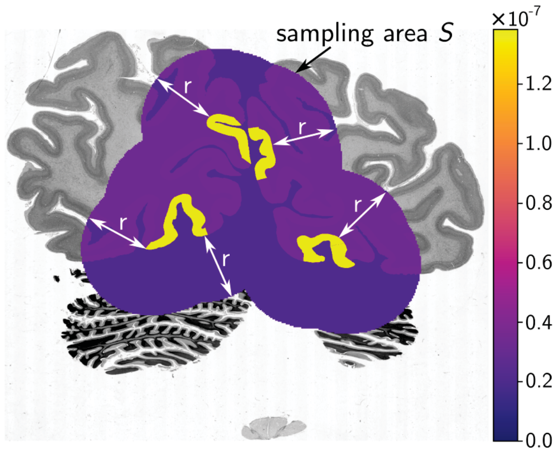


Figure 7.3: Sampling area $\mathcal{S}_r(1021)$ for `hOc2` on section 1021 of the BigBrain. The radius r is selected such that the sampling area includes `hOc2` and cortex pixels at a ratio of 1:4. Inside the sampling area, a sampling probability $p_{1021,i,j}$ is assigned to each pixel such that each class has equal probability of being sampled.

inverse of the frequency of its label in the sampling area with

$$p_{s,i,j} = \begin{cases} 0, & (i,j) \notin \mathcal{S}_r(s) \\ 1/|\mathcal{G}_{\mathtt{k}}|, & (i,j) \in \mathcal{S}_r(s)\,, \end{cases} \tag{7.2}$$

where `k` is the class of pixel (i,j): $(i,j) \in \mathcal{G}_{\mathtt{k}}$. Thus, the sampling area determines how much of the section the model can possibly see during training, and the subsequent weighting ensures that the model sees the same amount of pixels from each class.

The radius r of the sampling area is calculated for every section s as the radius at which the sampling area contains annotated brain area pixels and cortex pixels at a ratio of 1:τ: $|\mathcal{S}_r(s) \cap \mathcal{G}_{\mathtt{cor}}(s)| = \tau \cdot |\mathcal{G}_{\mathtt{a}}(s)|$. This ratio was empirically chosen to allow models to correctly predict the `cor` class. For `hOc1` the ratio was 1:1, and for `hOc2` 1:4. In the latter case, a smaller ratio had the effect that, due to the small size of `hOc2` on many sections, the sampling area was very small and there were too few samples for training. Figure 7.3 shows the sampling area for `hOc2` on section 1021 of the BigBrain. For each pixel in the sampling area, its normalized sampling probability $p_{s,i,j}/\sum_{i,j} p_{s,i,j}$ is shown.

7.2.3 Training procedure

To provide a good starting point for training, the high-resolution texture stream of the network is initialized with weights learned by the general area segmentation model `base[siam-vis]` (Section 6.3). The low-resolution context stream is initialized with random weights. It would be possible to use a higher learning rate for the randomly initialized weights as done in Section 6.2.3. However, even without this modification, convergence of the `prop-a-`s_1`-`s_2 models seems normal. The models are trained with an unweighted cross-entropy loss (Equation (2.18)) for 4000 batches with a batch size of 14. The loss included a small weight decay with weight decay factor $\lambda = 0.0001$.

Training was done with NAG with a momentum of $\mu = 0.9$ and a learning rate $\eta = 0.005$. Models were evaluated with the Dice score of the brain area on the validation section in between the training sections.

Since for dense label propagation several models need to be trained, it is important that the training can be done in a reasonable amount of time. Therefore, the code was optimized using an in-house adaption of Horovod (Sergeev and Balso, 2018) to enable training on several GPUs. With these optimizations, training one model takes 45 minutes on one JURECA node utilizing all 4 GPUs (Jülich Supercomputing Centre, 2018). For the evaluations presented in the following, areas `hOc1` and `hOc2` were densely predicted. This required the training of 18 `prop-hOc1-`s_1`-`s_2 models and 18 `prop-hOc2-`s_1`-`s_2 models.

7.3 Results

This section shows that the `MultiScaleNetwork` can be used to provide dense, high-resolution segmentation of one brain area. Section 7.3.1 analyzes the impact of the low-resolution context input for models trained on area `hOc2`. Then, the spatial coherence of the resulting predictions is shown Section 7.3.2.

7.3.1 Influence of context input

The motivation for including context information in the deep label propagation models is that it can be used to localize the brain area with the help of the local curvature of the cortex. To show the impact of the low-resolution context, `MultiScaleNetwork` models (`prop-a-`s_1`-`s_2) are compared to single-scale `BaseNetwork` models (`base-a-`s_1`-`s_2). Then, a `MultiScaleNetwork` model is analyzed to show that the model still needs high-resolution texture information to distinguish the brain area and surrounding cortex.

Comparison of multi-scale and single-scale model Label propagation models `prop-hOc2-`s_1`-`s_2 (uses context) and `base-hOc2-`s_1`-`s_2 (uses only high-resolution texture) are trained to propagate area `hOc2`. This area is harder to recognize and correctly segment than the primary visual cortex, area `hOc1`. To quantitatively compare the `prop-hOc2-`s_1`-`s_2 and `base-hOc2-`s_1`-`s_2 models, the Dice score for area `hOc2` is calculated for the sampling areas on the validation section of each of the 18 models. Figure 7.4 shows the Dice score of the two sets of models. In addition, Figure 7.5 shows predictions for several `prop-hOc2-`s_1`-`s_2 and `base-hOc2-`s_1`-`s_2 models. Each image shows the validation section s of a model trained on sections $s_1 = s - 60$ and $s_2 = s + 60$. Overall, the multi-scale models `prop-hOc2-`s_1`-`s_2 perform better and

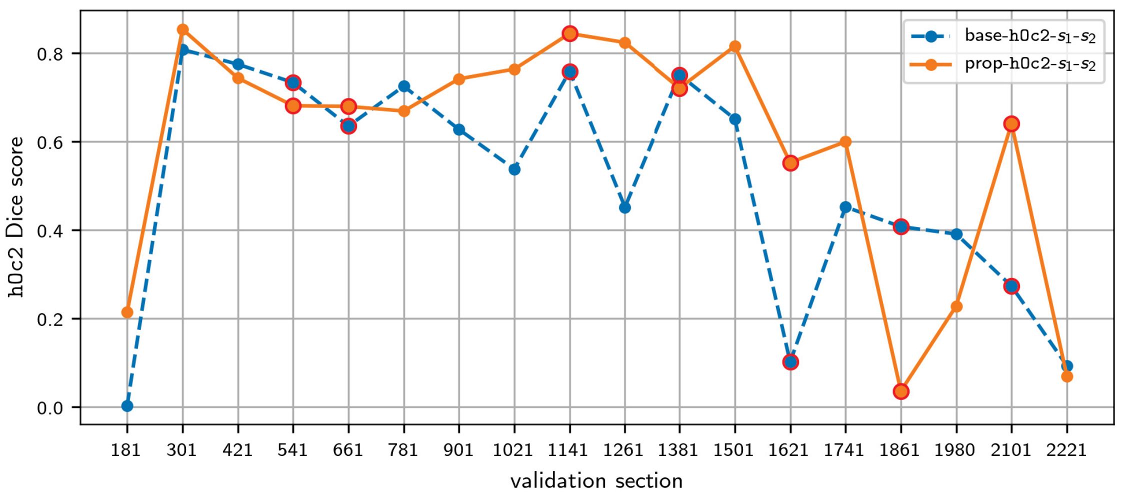


Figure 7.4: Deep label propagation models trained with context (prop-hOc2-s_1-s_2) and without context (base-hOc2-s_1-s_2). Shown are the Dice scores of hOc2 on the sampling area of the validation sections. Each point represents one model trained on two sections. For several models, the prediction on the validation section is additionally shown in Figure 7.5 (red circles).

are more consistent in their performance than the single-scale models base-hOc2-s_1-s_2.

However, as the size of hOc2 on the sections decreases (with increasing section numbers), predictions of both models are noisier (e.g., sections 1861, 2101 in Figure 7.5). Models trained on larger section numbers see only a small part of the tissue where hOc2 is located (visualized by the spotlight in Figure 7.5). Note that the models do not see the validation section shown in the figure, but a comparable sampling region on the two training sections. This focus on a spatially small sampling region has the positive effect that the model can specialize to recognize the brain area in spatially close sections, but has the downside that it might erroneously recognize the brain area in other parts of the validation section as well, because it has never seen these particular parts of the cortex before. However, as described in Section 7.4, an automatic post-processing step can remove the noise in the cortex. After this post-processing, only activations in the cerebellum (e.g., in section 1861) remain. Although it has not been done for this thesis, the cerebellum can be easily masked out based on its extremely dark appearance due to the high density of neurons. Thus, the noise outside the sampling region can be easily removed and further evaluation focuses on the performance of the models inside the sampling region.

Close to the occipital pole of the brain (low section numbers), models prop-hOc2-s_1-s_2 and base-hOc2-s_1-s_2 have similar Dice scores. However, the prop-hOc2-s_1-s_2 models are slightly better at localizing area hOc2 (e.g., sections 541, 661). This better

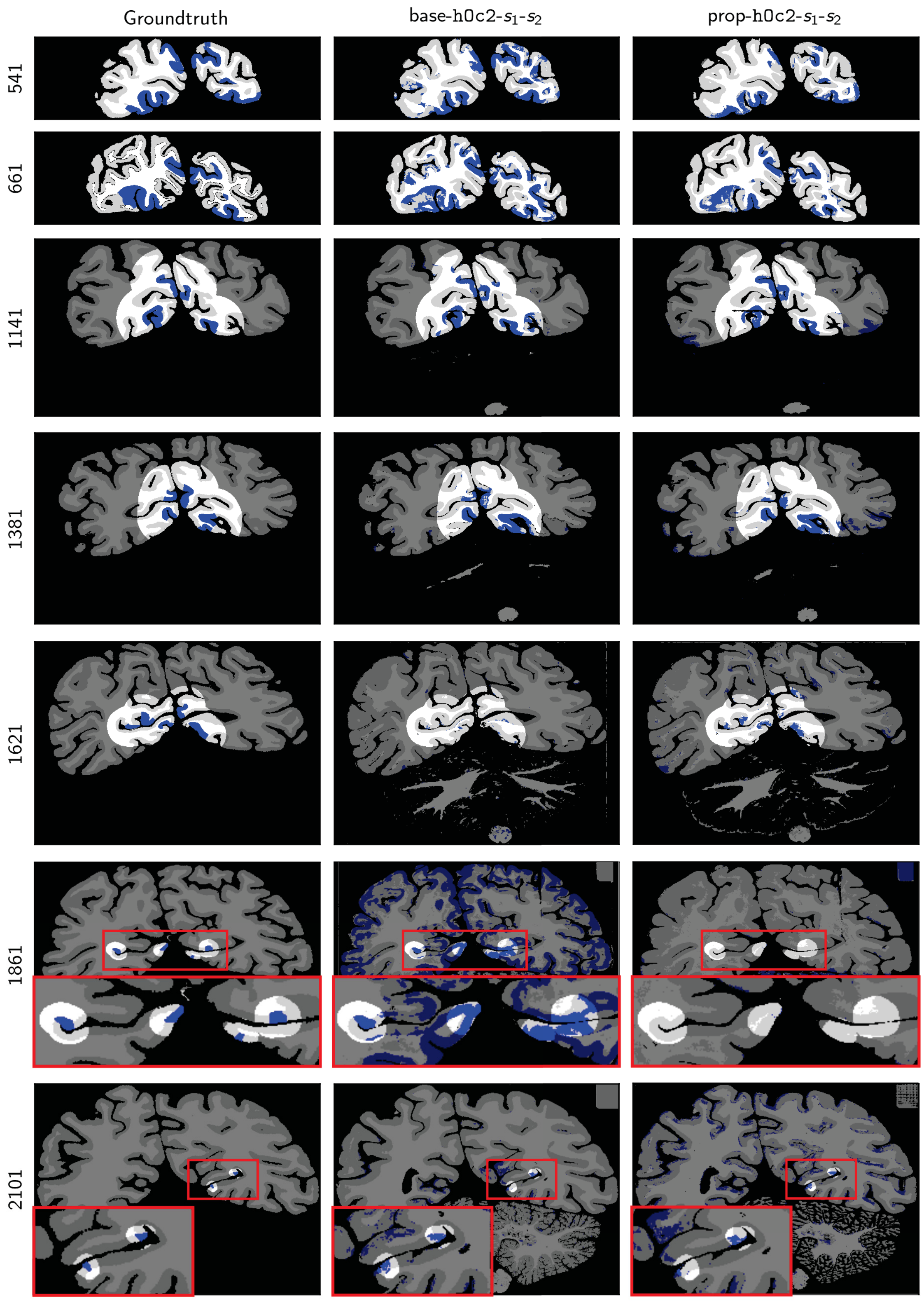
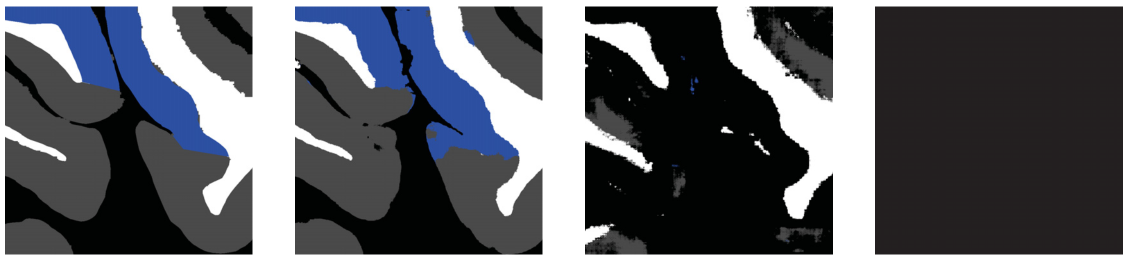


Figure 7.5: Predictions of models trained with context (`prop-h0c2-`s_1`-`s_2) and without context (`base-h0c2-`s_1`-`s_2). The pixels outside the sampling area $\mathcal{S}$ of each section are darkened – an automatic post-processing step can remove these responses (Section 7.4.1). For sections 1861 and 2101, $\mathcal{S}$ is displayed larger to show details (red boxes). The results are discussed in Section 7.3.1.



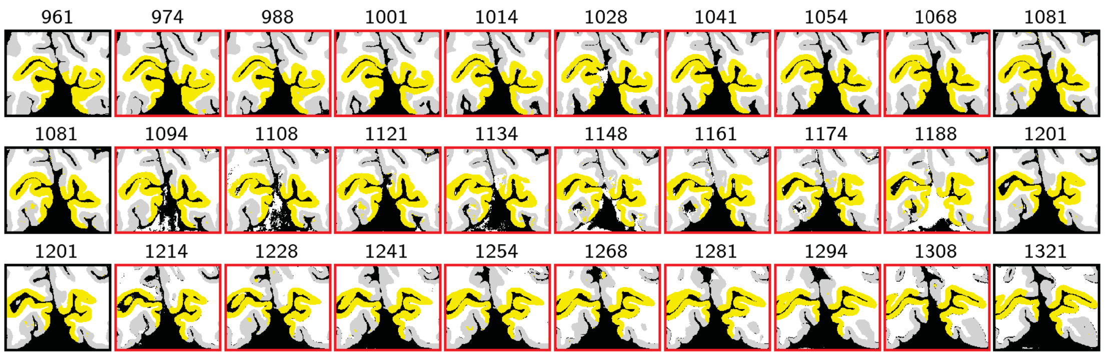
(a) Groundtruth **(b)** Prediction with texture and context **(c)** Prediction with context only **(d)** Prediction with texture only

Figure 7.6: Predictions of a multi-scale mode while feeding zeros for one input type. The model requires both low and high-resolution inputs to make correct predictions.

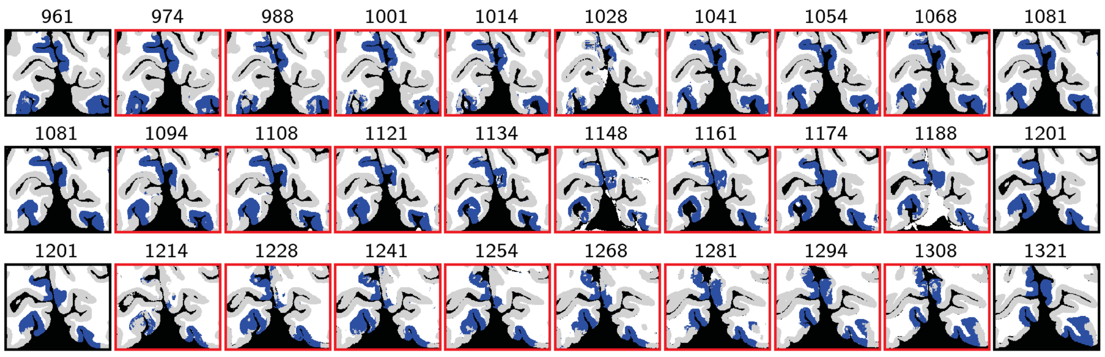
localization becomes more pronounced, as the sections become larger (e.g., sections 1141, 1621). The `hOc2` annotations after section 1741 are very small, making it difficult for both models to localize the area in the section even with context information. On section 1861, `base-area-1801-1921` recognizes `hOc2` in in many locations on the cortex, including a correctly segmented piece of `hOc2` on the left hemisphere. In contrast, `prop-hOc2-1801-1921` does not recognize `hOc2` at all. For this range of sections, the sampling region is so small that both models could not learn good features. On section 2101, the models mostly manage to localize the area.

In conclusion, these evaluations show that the performance of the `MultiScaleNetwork` models is more consistent than the performance of the `BaseNetwork` models. However, performance is limited when the size of the area on the training sections is very small.

What is the context used for? In order to better understand what features the model extracts from which input, a `MultiScaleNetwork` model is used to predict its validation section in the following way: first, by inputting zeros instead of the context information, and second by inputting zeros instead of the high-resolution texture information. The results in Figure 7.6 show that without the context information, the model predicts the label `bg` for the entire section (Figure 7.6d). In contrast, without the high-resolution input, the model predicts `bg` for background and cortex, and `wm` for white matter (Figure 7.6c). This shows that a model trained with both context and high-resolution information uses both inputs. The context input is sufficient for background and white matter segmentation, however the high-resolution input is needed in order to predict `hOc2` and `cor`. These results indicate that the `MultiScaleNetwork` uses the context information to roughly localize the brain area, and do the cortex and background segmentation, and the high-resolution texture information to precisely segment the brain area and surrounding cortex. This is in accordance with our expectation and similar to the reasoning of a human expert.



(a) `hOc1`. Row 1 shows predictions of `prop-hOc1-0961-1081`. Row 2 shows predictions of `prop-hOc1-1081-1201`. Row 3 shows predictions of `prop-hOc1-1201-1321`.



(b) `hOc2`. Row 1 shows predictions of `prop-hOc2-0961-1081`. Row 2 shows predictions of `prop-hOc2-1081-1201`. Row 3 shows predictions of `prop-hOc2-1201-1321`.

Figure 7.7: Predictions of `hOc1` and `hOc2` models on a set of consecutive sections. The predictions are compatible with each other and spatially coherent. Training sections are marked in black, previously unlabeled sections in red.

7.3.2 Spatial coherence of predictions

The resulting segmentations are spatially coherent and the predictions for different areas are compatible. To show this, three `MultiScaleNetworks` were trained on sections (961,1081), (1081,1201), (1201,1321) for propagating areas `hOc1` and `hOc2`. Each model was evaluated on a set of 10 sections from the `BigBrainCollection` which were located in between the two training sections.

The overlap of the `hOc1` and `hOc2` prediction on these sections is 0.1% of the pixels predicted as `hOc1` and `hOc2`. This shows that even though the models are only trained on one brain area, they are compatible with models predicting other areas. This is important if several brain areas should be propagated to the same section.

Figure 7.7 shows the predicted segmentation on registered sections. This allows to appreciate the evolution of the area border throughout the sections. Although the predictions were done on the unregistered, non-aligned sections, the predictions are very coherent. Predictions for area `hOc1` are less noisy than predictions for area `hOc2`.

7.4 High-resolution 3D reconstruction of brain areas

The dense predictions of brain areas can be used to form a high-resolution 3D mesh of the brain area. For this step, a 3D registration between the individual sections is required. On the one hand, this demonstrates the quality of the automatic annotations. On the other hand, the resulting 3D mesh of the brain areas can be integrated in high-resolution brain templates and atlases. In this section the 3D reconstruction of the BigBrain was used to create 3D meshes of areas `hOc1` and `hOc2`. Section 7.4.1 describes the workflow for creating the 3D meshes and Section 7.4.2 shows the resulting reconstructions.

7.4.1 Generation of reconstruction

As described in Section 7.2.3, 18 `prop-hOc1-`s_1`-`s_2 and `prop-hOc2-`s_1`-`s_2 models were trained on consecutive pairs of sections s_1 and s_2 and used to predict area `hOc1` and `hOc2` on sections in between s_1 and s_2. From these predictions, masks for `hOc1` and `hOc2` were calculated on every section. These predictions were then used to construct a high-resolution 3D volume of `hOc1` and `hOc2` (Figure 7.8).

Using the available transformation files for the BigBrain (calculated at 20 µm precision; Section 2.3.3), the masks were transformed to the BigBrain volume. Since some sections were missing in the `BigBrainCollection` due to large tissue artifacts, there were gaps in the masks. To form a dense volume of the area, missing masks on section s were interpolated from their neighboring sections s_i and s_j, with $s_i < s < s_j$. For this interpolation, the potential field between the overlap of the masks on sections s_i and s_j and the background was calculated with Laplace's equation. The mask on section s was sampled at the equipotential surface at value $(s - s_i)/(s_j - s_i)$.

As mentioned in Section 7.3.1, parts of the tissue that are far away from the location of the brain area on the section are likely to have noisy predictions. Fortunately, this noise is not systematic and only occurs in different regions for consecutive sections, whereas the prediction for the brain area is always at the same place. Thus, the brain area should form a large connected component on the resulting volume. Connected components were calculated in the volume and every component with a size of at least 20% of the size of the largest component was included in the final volume. Visual inspection resulted that one of the four components for area `hOc1` and two of the eight components for area `hOc2` belonged to the cerebellum and were discarded. All the other label noise present in the original masks was filtered out by calculating

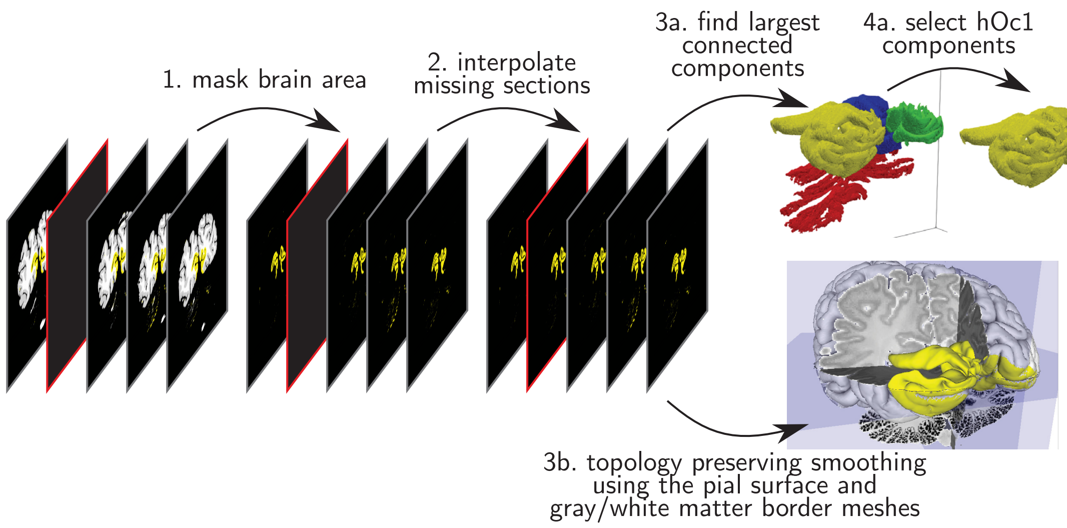

1. mask brain area
2. interpolate missing sections
3a. find largest connected components
4a. select hOc1 components
3b. topology preserving smoothing using the pial surface and gray/white matter border meshes

Figure 7.8: 3D reconstruction workflow. Steps 3a and 4a result in a non-smoothed, noise-free volume that shows the precision of the individual predictions. Step 3b results in a smoothed, topologically correct volume that can be integrated in the multilevel human brain atlas (*HBP Human Brain Atlas* 2018).

the largest connected components. The resulting meshes candidly show the quality of the predictions in 3D with minimal post-processing applied.

For the integration of the predictions in the multilevel human brain atlas, more post-processing was applied to create smoothed, topologically correct area meshes. The brain area volume was smoothed using morphological filtering by geodesic alternate sequential filters (Couprie and Bertrand, 2004).[1] This method works by smoothing the brain area mask while respecting the topology of the mask and geometrical constraints imposed by a mask of the cortical ribbon. This results in a smoothed mask that lies inside the cortical ribbon and has the same topology as the original mask.

7.4.2 Results: hOc1 and hOc2 meshes

Figure 7.9 shows several views of the `hOc1` and `hOc2` meshes created by steps 3a and 4a. This visualization shows the extent and connection between the two areas. These meshes were not smoothed. This can be seen in some places where `hOc2` extends over the imagined smooth contour of the `hOc2` mesh. This slight noise might be due to artifacts on the individual section resulting in misclassifications. The 3D view on the predicted areas shows that they fit well together. This shows that the `hOc1`/`hOc2` border has been correctly recognized by models learning to predict `hOc1`

[1]Thanks to Dr. Yann Leprince (NeuroSpin) for calculating this post-processing and for integrating the resulting meshes in the multilevel human brain atlas.

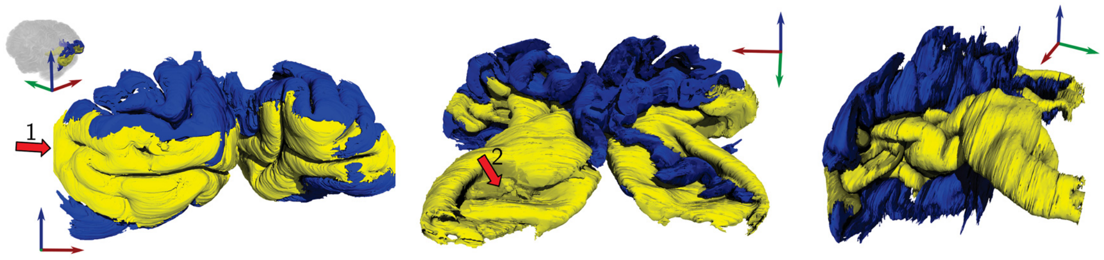


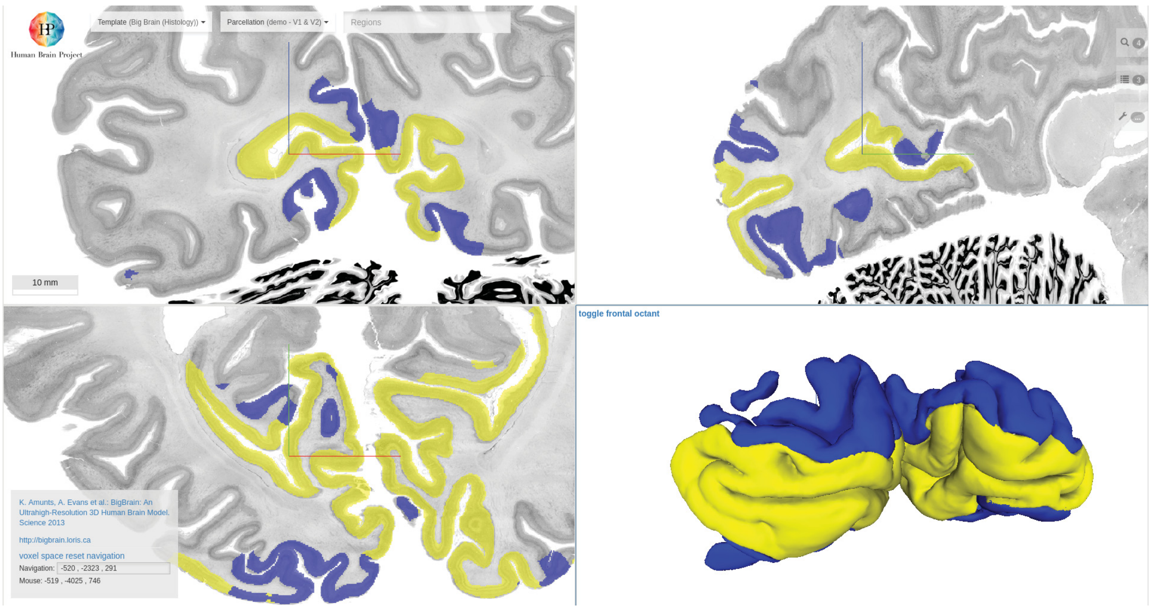
Figure 7.9: Views of the reconstructed `hOc1` and `hOc2` volumes that were created from a few annotations using deep label propagation. Red arrows show locations where `hOc2` predictions are missing.

and by models learning to predict `hOc2`. However, some inconsistencies remain: The extent of `hOc2` is too small in several parts (marked with red arrows in Figure 7.9). Several of the missing predictions could be traced to a bad training section with tissue artifacts. This training section negatively influenced the performance of the two models that were trained on this section. Exchanging this training section improved the results at arrow 1. This shows that although the label propagation process is automatic, careful selection of good training sections (i.e., no artifacts) is necessary. The remaining missing `hOc2` predictions are due to very small annotations on the training section.

The meshes created by additional topology-preserving smoothing (step 3b) can be integrated in the multilevel human brain atlas (*HBP Human Brain Atlas* 2018). Figure 7.10 shows a screenshot of the atlas viewer where this integration was demonstrated. This software allows users to examine the `hOc1` and `hOc2` predictions overlayed on the reconstructed 20 µm BigBrain volume. After further evaluation of the correctness of the predictions, these areas will be made available publicly available as shown in Figure 7.10. Previously, 3D meshes of single brain areas were not available at such a resolution and precision. In the context of the Human Brain Project (HBP), this workflow will be implemented into a toolchain, striving to map all of the BigBrain at microscopic resolution.

7.5 Discussion

This chapter introduced the deep label propagation task which has the aim to create dense segmentations of one brain area in one brain volume given a few manual expert annotation of this brain area. A `MultiScaleNetwork` is trained on two spatially close sections to predict one brain area on all sections in between the training sections. Like the models introduced in Chapters 5 and 6, the `prop-a-`s_1`-`s_2 models use high-resolution texture information to localize the brain area on the input. Since the



Template (Big Brain (Histology))
Parcellation (demo - V1 & V2)

K. Amunts, A. Evans et al.: BigBrain: An Ultrahigh-Resolution 3D Human Brain Model. Science 2013

http://bigbrain.loris.ca

Navigation: -520 , -2323 , 291
Mouse: -519 , -4025 , 746

Figure 7.10: Screenshot of the multilevel human brain atlas viewer (*HBP Human Brain Atlas* 2018) showing the BigBrain with the integrated `hOc1` and `hOc2` meshes.

training and testing data for the deep label propagation task is very similar, the `prop-a-`s_1`-`s_2 models combine the high-resolution information with low-resolution context information that helps to localize the brain area using anatomical landmarks.

The resulting brain area segmentations are precise and spatially coherent. Predictions for several areas are compatible with each other, with very little overlap between the labels. From these predictions it is possible to calculate a 3D mesh of each brain area at a previously unseen resolution. However, when the brain area is very small, the whole section predictions can be noisy. Using predictions of several consecutive sections, this noise can be filtered out by calculating connected components.

One remaining challenge is the performance of the label propagation of very small brain areas. The models are trained on a sampling area $\mathcal{S}$ surrounding the brain area. For small brain areas this sampling area does provide enough information to train robust label propagation models. A larger context could be useful to allow localization of these small patches.

Currently, no data augmentation is used when training the label propagation models. The rationale is that the training data is already similar to the evaluation data and thus a generalizable area segmentation model is not needed. However, the training and evaluation sections are not registered. Thus, they might be rotated relative to each other. Therefore, augmenting the data with random rotations by a small angle might be helpful to further increase the precision.

The deep label propagation was executed for areas `hOc1` and `hOc2`. Prediction of area `hOc1` is quite straight-forward and serves here as a proof-of-concept. Prediction of area hOc2 is much more difficult, especially the correct localization of the border between hOc2 and neighboring areas hOc3d and hOc3v. The general-purpose segmentation models based on the AtlasAwareNetwork do not always manage to localize this border. The label propagation models based on the MultiScaleNetwork, however, correctly localize this border in many sections. Since this difficult border is localized, it is expected that other brain areas can also be propagated using the deep label propagation task. However, future work should evaluate the deep label propagation for more areas.

The ability to localize arbitrary brain areas comes at the cost of needing several annotations of the brain area in sections that are close to the section that should be predicted. The deep label propagation therefore complements the current manual mapping process and allows the calculation of densely labeled brain area volumes (if a registration is present) at no additional cost to the neuroanatomist. In contrast to related approaches (Atzeni et al., 2018), the deep label propagation itself does not require a registration between the sections. In contrast, in the future the predicted brain area masks could be used to drive the registration process. Another application is to use the predictions of the label propagation model as training data for the general-purpose area segmentation network. This way, more diverse training data could be included in the brain area segmentation training and in particular increase the variability of underrepresented, small brain areas.

8 Conclusion

This work has shown that deep learning models are suitable for automatically identifying cytoarchitectonic brain areas in high-resolution digital scans of histological human brain sections. To reach this conclusion, four different approaches using different network architectures and input data were developed and compared. Each approach has different strengths and limitations which are summarized in Table 8.1 and are discussed in the following. Figure 1.2 on Page 7 shows a visual summary of the different neural network architectures developed for this thesis.

Section 8.1 summarizes the different models and their results. Section 8.2 lists the main contributions of this work. Then, the limitations of the models and general recommendations for automatic brain area segmentation using deep learning are discussed in Section 8.3. The chapter ends with identifying several directions for future work arising from the methods and models developed in this thesis (Section 8.4).

8.1 Summary

Inspired by the current observer-independent statistical method for confirming areal borders, a workflow to automatically extract gray-value intensity profiles from the cortex was developed. With these profiles, a 1D convolutional neural network (CNN) was trained on the area classification task (Chapter 4). A saliency analysis of this model gives insight into which parts of a profile, and thus which cortical laminae, are most descriptive for identifying a cortical area. However, the classification performance is inferior to a segmentation model based on high-resolution 2D input patches.

The segmentation model was improved by informing it with prior information from a probabilistic atlas (Chapter 5). Although an analysis of intermediate feature maps of the model indicates that it derives its results based on reasonable cytoarchitectonic features, its potential is restricted by the limited amount of training data. To further increase performance of the supervised segmentation, a self-supervised learning task based on the 3D relationship between individual sections of the same brain volume was developed to leverage unlabeled sections of one human brain volume (Chapter 6). Pre-training the supervised segmentation models on this self-supervised task significantly increases the segmentation performance and enables the prediction of several brain areas on previously unseen brain volumes.

Table 8.1: Characteristics, strengths and limitations of the different approaches for brain area segmentation. Models are trained on gray-value profiles or high-resolution image patches. As additional information, an atlas prior or context information is used. Optionally, the models are pre-trained on an auxiliary task. Concepts that may be added to a task in the future and that are discussed in the following are marked with an asterisk *.

Task	Used concepts					Main advantages and limitations
	profiles	texture	atlas prior	context	pre-training	
profile classification	✓	✗	✗*	✓	✗*	+ models easy to interpret + generalizes naturally to the 3D case - outperformed by patch segmentation
patch segmentation	✗	✓	✓	✗	✓	+ leverages prior knowledge and pre-training - does not perform equally well for each area
label propagation	✗	✓	✗	✓	✓	+ precise for arbitrary areas - needs annotations in spatially close sections
distance and location regression	✗*	✓	✗	✗	✗	+ learns cytoarchitectonic features - task biases spatially close patches towards similar features

The self-supervised task allows the model to learn a compact feature representation of the input by exploiting the structure of the input data. Evaluations of the self-supervised feature representations show that they encode cytoarchitectonic properties of the input without explicitly being trained on area labels.

To show the applicability of these automatic models for current brain area mapping projects, a specialized multi-scale CNN model for predicting one brain area in a set of adjacent sections was introduced. This model complements the current manual mapping process of labeling every 60th section by creating dense, high-resolution, and precise labels of one brain area. These predictions allow the creation of high-resolution 3D reconstructions of a single brain area at previously unseen resolutions.

8.2 Contributions

This work shows that it is possible to apply deep learning techniques to the expert task of brain area segmentation in high-resolution histological sections. Three main contributions with high relevance to the research field of brain area parcellation can be identified:

1. **A CNN model for automatic brain area segmentation.** This model combines prior knowledge with high-resolution histological input patches. It can predict several brain areas at reasonable precision in previously unseen sections and outperforms a 1D CNN model trained on gray-value profiles. This model demonstrates that automatic brain area segmentation is possible with deep learning methods.

2. **A specialized CNN model for label propagation between spatially close sections.** Using both high-resolution texture information and low-resolution context, this model allows to segment one brain area given an annotation of this area in a spatially close section. This shows the practical relevance of this work in a concrete application. This workflow complements the manual brain area mapping workflow and allows the high-resolution 3D reconstruction of brain areas without adding to the manual parcellation workload of the neuroanatomist. These 3D reconstructions will be made publically available in the multilevel human brain atlas (*HBP Human Brain Atlas* 2018). This workflow has the potential to support mapping projects for new brain volumes or brain areas.

3. **A self-supervised method that learns to encode cytoarchitectonic properties without requiring prior labels and assumptions about brain areas.** This task leverages the 3D relationship between unlabeled sections of one brain volume. It allows the data-driven analysis of cytoarchitectonic patterns in histological sections and could be used to research the relationship between cytoarchitecture and connectivity (Beul et al., 2017). This self-supervised task is easy to extend to different resolutions or modalities.

Addressing the challenges of histological brain area analysis The proposed CNN models overcome several of the challenges posed by automatic brain area parcellation (Section 1.1). The difficult *localization* of brain areas is handled with two different approaches: The first approach is the segmentation model from Chapter 5 where the key idea is to combine high-resolution texture information with prior knowledge about probable areas at the given input. The second approach is to train specialized label propagation models that focus on predicting one brain area in spatially close sections utilizing low-resolution texture information providing additional context together with the high-resolution texture information (Chapter 7). Thus, some form of additional knowledge (atlas prior, or context information) was necessary for successful localization of brain areas. The fixed *topology* of brain areas is indirectly modeled by the atlas prior. The topological correctness of trained models was evaluated using a weighted misclassification error. However, the segmentation model has no direct incentive to respect the topology of areas. As motivated in Section 5.4, in future work a conditional random field (CRF) might be used to directly model the topology of areas. A strength of the patch-based CNN models is their relative robustness to *artifacts* in the data. The gamma augmentation (for the segmentation models) and the gray-value normalization (for the profile classification model) worked well to alleviate the staining differences between individual sections. The key to overcoming the *limited labeled training data* was to pre-train the models on an auxiliary self-supervised task leveraging unlabeled histological sections (Chapter 6). The pre-trained models learned features from histological sections that could be efficiently adapted during fine-tuning for brain area segmentation. Due to the *large inputs* required for brain area segmentation, a memory-efficient segmentation model based on downsampling and subsequent upsampling of the inputs was used. Combined with the use of several GPUs on the JURECA supercomputer (Jülich Supercomputing Centre, 2018) per training process, training was possible in reasonable time with batch sizes of around 40 (Section 5.2.5).

8.3 Limitations and general recommendations

Status of automatic brain area segmentation Several models for different tasks related to brain area segmentation were introduced in the last chapters. Do they solve automatic brain area segmentation? The answer depends on how general the task of brain area segmentation is posed and what data is available to train the neural network models. In the following, different instances of the brain area segmentation task are discussed, and recommendations for researchers that are faced with such a task are given.

Can we solve the task of brain area segmentation with ...

- **... annotations available in spatially close sections of the same brain volume?** *Yes.* The deep label propagation workflow solves this task by training specialized multi-scale models for the propagation of one brain area to spatially close sections. In this case, the model learns topological properties of the individual subject to identify an area, which is not desired in the general setting. Chapter 7 demonstrates that this workflow results in high precision segmentations of areas `hOc1` and `hOc2` by leveraging the similarity of training and evaluation data. The deep label propagation should work for arbitrary areas, but currently requires annotations on every 120th section.
- **... annotations available in sections of the same brain volume?** *Yes, but sufficient training data or prior knowledge is needed.* The two stream `AtlasAwareNetwork` combining high-resolution histological sections and an atlas prior performed adequately on this task (Chapter 5). However, the performance depends on how difficult an area is to recognize and how much annotated

data is available for this area. For example no model was able to learn area hOc5 because only 14 annotations (amounting to only 3.5% of the annotated hOc1 pixels) were available. In contrast, hOc1 was learned by every model, because this area is easy to recognize due to the characteristic stria of Gennari. If not enough labeled training data is available, this model can be pre-trained with a self-supervised task as demonstrated in Chapter 6. If a 3D isotropic brain volume like the BigBrain volume at 20 µm resolution is available, using profile features could be considered. 3D data allows to extract profiles which are always perpendicular to the brain surface (which is not possible using 2D sections) and thus compress relevant information of the cortical laminae. Compared to a 3D convolutional approach, this is a much more compact representation. Like the area segmentation model, such a profile model could be pre-trained with the self-supervised distance task and be extended to leverage atlas information.

- **... annotations available in other brain volumes?** *Partly, and it depends on the brain area.* Fine-tuning a pre-trained AtlasAwareNetwork on annotations in several brain volumes and applying it to a previously unseen brain volume resulted in a model that was predicting area hOc1 correctly (Section 6.3.3). Again, for good performance large amounts of diverse training data are required. Transferring models to other brains is difficult due to inter-individual differences. Although a simple gray-value augmentation technique was used, more sophisticated normalization or domain adaptation techniques should be investigated in the future to get better performance.
- **... no annotations available?** *No, but anatomically meaningful regions can be found.* The self-supervised distance and location regression task allows to learn a feature representation for each high-resolution input patch that encodes cytoarchitectonic properties (Chapter 6). These features cluster in anatomically meaningful regions that respect cytoarchitectonic differences. Clusters for hOc1 and primary motor cortex were found, but usually the clusters are quite large and encompass several brain areas. The features additionally allow to assess similarity of cytoarchitecture and can be computed in every point in the cortex. Currently, the calculation of such features requires one brain volume with registrations between each section and a 3D cortical mesh. The transferability of the features to a new brain volume remains to be seen.

Lessons learned about training convolutional neural networks The work presented in this thesis allows to draw several conclusions concerning the training of CNNs in different scenarios. Several models were partly fine-tuned from weights learned on auxiliary tasks (e.g., initialization of the downsampling path by the self-supervised model). In such a case, it was useful to increase the learning rate of the remaining randomly initialized weights and reduce the learning rate for the pre-trained weights. This was especially necessary when only the first layers of a model (e.g., only the downsampling path) were pre-trained and the later layers (e.g., the upsampling path) were not. The segmentation models include an atlas prior input. Such inputs may be useful for many medical applications where training data is limited but atlases are present. When training with an input that provides an estimate of the output labels, it is necessary to discourage the CNN from learning the input too quickly. In this thesis this was achieved by randomly dropping out 20 % of the atlas prior input and using an iterative training procedure: First, the model was trained to convergence without the atlas prior, and then it was fine-tuned with the atlas prior.

8.4 Future work

In addition to the discussions of future work in the experiment Chapters 4 to 7, this section highlights a few promising directions for future research based on the results presented in this thesis. The segmentation models include an atlas prior which acts as a soft constraint for the topology of areas. However, for future work it might be advantageous to include a hard constraint for the topology of areas by for example integrating a CRF on top of the segmentations of the CNN model as in L.-C. Chen et al. (2018). Such a model could enforce spatially coherent and topological correct segmentations without loosing the advantages of the deep learning approach.

The performance of brain area segmentation is limited by the small amount of training data. This was partly alleviated by pre-training on the self-supervised task. However, more training data for the original task would be advantageous. Rather than manually labeling more training sections, the deep label propagation workflow could be used to provide more training data. With this workflow, brain area labels can be precisely propagated to spatially close sections. These newly labeled sections provide valuable additional groundtruth for the multi-area segmentation models. In particular, underrepresented labels could be boosted with this technique.

The features learned by the self-supervised distance and regression task are very promising. They provide a way to compress cytoarchitectonic information from a high-resolution patch to a 32-long vector and provide insights in the objectively visible feature in the data. In the future, this feature representation could be used to study the connection between connectivity and cytoarchitectonic similarity, similar to previous work on less precise measurements of cytoarchitecture (e.g., Beul et al., 2017; Wei et al., 2018). Additionally, the self-supervised distance and regression task can be used to extract features from other imaging modalities (e.g., magnetic resonance imaging (MRI) or polarized light imaging data) and used to validate other hypotheses about brain structure. Furthermore, the self-supervised task does not have to be trained on 2D patches for cortex; it could also be applied to intensity profiles, extracted, e.g., from the 3D BigBrain volume.

Finally, it may be worthwhile to design and evaluate other unsupervised feature learning architectures and algorithms, and compare the learned feature representations with each other. The distance and location task has the limitation that due to the design of the task, the resulting feature representations will be biased in such a way that spatially close patches get similar representations. This may be desirable when considering brain areas and cytoarchitecture (as cytoarchitectonic similarity is correlated with location), but it bears the danger of under-emphasizing long-range similarities in the brain.

To conclude, this work has shown that CNNs can be a powerful tool for brain area segmentation even with limited training data. Including prior knowledge in the training process and using self-supervised pre-training has proven to be especially useful. Limiting the task to predict areas in spatially close sections allows to achieve very precise segmentations for arbitrary brain areas. The self-supervised distance and location regression task allows to learn features that encode cytoarchitectonic properties which can be leveraged to analyze the brain in new ways and may provide new insights in the future.

Appendix

A.1 Author contribution to publications presented in this thesis

The work described in this thesis was already partially published in three publications with me, Hannah Spitzer, as the first author. For all of the papers I was the main responsible for designing and implementing the neural network architectures, training the neural networks, evaluating the results, and writing the manuscript. The distance and location regression task presented in Spitzer, Kiwitz, et al. (2018) was conceptualized in discussion with Dr. Timo Dickscheid. The co-authors have agreed to the use of these papers in this thesis. In the following, the publications and the extent to which they are used in this thesis are shortly described.

H. Spitzer, K. Amunts, et al. (2017). "Parcellation of visual cortex on high-resolution histological brain sections using convolutional neural networks". In: *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*. IEEE, pp. 920–923 Authors: Spitzer, Amunts, Harmeling, and Dickscheid. This paper describes the two stream CNN model which combines prior information with high-resolution texture input. Chapter 5 extends this paper. The neural network architecture (Section 5.1) and the training procedure (Sections 5.2.1 to 5.2.4) described in this thesis are the same as in Spitzer, Amunts, et al. (2017). The evaluation of the influence of the atlas prior described in Section 5.3.2 was done in a similar manner in Spitzer, Amunts, et al. (2017). Since the models presented in this thesis were retrained on a larger dataset and with a slightly different learning rate and label weighting scheme, the scores reported in Spitzer, Amunts, et al. (2017) and Section 5.3.2 differ.

H. Spitzer, K. Kiwitz, et al. (2018). "Improving cytoarchitectonic segmentation of human brain areas with self-supervised siamese networks". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*. Springer, pp. 663–671 Authors: Spitzer, Kiwitz, Amunts, Harmeling, and Dickscheid. This paper describes the self-supervised distance and location regression task and how the self-supervised pre-training improves the supervised area segmentation. The

Siamese network model (Section 6.1) and the self-supervised training and fine-tuning (Section 6.2) are described in Spitzer, Kiwitz, et al. (2018). The ablation study from Section 6.3.1 and the comparison to the training without fine-tuning (Section 6.3.2) were done in a similar manner in Spitzer, Kiwitz, et al. (2018). Since the self-supervised models presented in this thesis were trained on a larger dataset than the models in Spitzer, Kiwitz, et al., 2018, the reported scores differ.

H. Spitzer, K. Amunts, et al. (2018). "Compact feature representations for human brain cytoarchitecture using self-supervised learning". In: *1st International Conference on Medical Imaging in Deep Learning*. URL: `https://openreview.net/forum?id=HyjJ812of` Authors: Spitzer, Amunts, Harmeling, and Dickscheid. This contribution analyzes the self-supervised features using a hierarchical clustering. A similar clustering (which results in the same conclusions) is presented in Section 6.4.3. The remaining subsections in Section 6.4 provide further analyzes of the self-supervised features.

A.2 Author contribution to collaborative work presented in this thesis

The ideas and results described in Chapters 4 and 7 are the result of collaborative work. Currently, no publications are available, but for each of the chapters, a publication is in preparation. The collaborators have agreed to the use of the results for this thesis. In the following, the collaborations and my contribution to them is described.

Area classification using profile features (Chapter 4) This chapter results from a collaboration with Konrad Wagstyl, University of Cambridge, where Konrad Wagstyl contributed expertise in extracting and using profile features and I expertise in deep learning for cortical areas. The main part of the work was done during a stay at the ACELab under the direction of Prof. Alan Evans at the Montréal Neurological Institute (MNI), McGill University, Montréal. We developed the profile extraction pipeline (Section 4.1) and the neural network architecture (Section 4.2) together and I trained the models discussed in Sections 4.3 and 4.4.

Application: Deep label propagation between sections (Chapter 7) This chapter results from work done by Christian Schiffer during his masters studies under my supervision. The development of the data sampling and the hyper-parameter tuning for the multi-scale CNN model was done by Christian Schiffer (Sections 7.1 and 7.2).

Basing on the dense predictions, I calculated the 3D reconstructions of `hOc1` and `hOc2` described in Section 7.4.

Glossary

Abbreviations

batch norm	batch normalization
CNN	convolutional neural network
conv	convolutional layer
CRF	conditional random field
EM	expectation-maximization
fcl	fully connected layer
FCN	fully convolutional network
fMRI	functional magnetic resonance imaging
GLI	gray-level index
HBP	Human Brain Project
HPC	high performance computing
INM-1	Institute of Neuroscience and Medicine
max pool	max pooling layer
MNI	Montréal Neurological Institute
MRI	magnetic resonance imaging
NAG	Nesterov accelerated gradient
OCT	optical coherence tomography
PCA	principal component analysis
ReLU	rectified linear unit
RNN	recurrent neural network
SGD	stochastic gradient descent
VAE	variational autoencoder

Nomenclature

Mathematical notation

$*$	convolution operator
$U(a, b)$	uniform distribution between a and b

$X^{(i,j)}$	element at row i and column j of matrix $X \in \mathbb{R}^{n\times m}$; i or j may be replaced by a colon, with $X^{(i,:)}$ denoting row i of X and $X^{(:,j)}$ denoting column j of X; can be extended to higher-dimensional tensors
$X^{\odot a}$	Hadamard power; takes each element of matrix X to the a-th power
∇_θ	nabla operator; calculates partial derivatives with respect to parameter vector θ
$x \sim \mathcal{A}$	x has the distribution $\mathcal{A}$
$x^{(i)}$	element at index i of vector $x \in \mathbb{R}^n$ with elements $x^{(1)}, ..., x^{(n)}$

Symbols

X	input to neural network/layer (tensor with rank 1 or higher) 20
Y	groundtruth output (tensor with rank 1 or higher) 29
Z	output of neural network/layer (tensor with rank 1 or higher) 20
η	learning rate 29
λ	weight decay factor 31
$\mathcal{G}_k$	set of all indices of pixels labeled as class k for groundtruth labels $\mathcal{Y}$ 31
$\mathcal{P}_k$	set of all indices of pixels predicted as class k 31
$\mathcal{X}^{\text{te}}$	test split of dataset $\mathcal{X}$ (used for evaluation) 36
$\mathcal{X}^{\text{tr}}$	train split of dataset $\mathcal{X}$ (used for training) 36
$\mathcal{X}^{\text{val}}$	validation split of dataset $\mathcal{X}$ (used for hyper-parameter tuning) 36
$\mathcal{X}$	set of neural network inputs X 29
$\mathcal{Y}$	set of groundtruth labels for inputs $\mathcal{X}$ 29
Dice	Dice score 32
acc	accuracy 32
err	weighted misclassification error 33
μ	momentum factor 29
θ	set of parameters of a neural network 20
d_{ij}	geodesic distance between location p_i and p_j following along the cortical surface 93
p	sampling location of input image X on the pial surface 93
q	sampling location of input image X on the inflated gray/white matter border mesh 93

Functions

$L(\mathcal{X}, \mathcal{Y}; \theta)$	loss function evaluating parameters θ of neural network with respect to inputs $\mathcal{X}$ with labels $\mathcal{Y}$ 29

$\Psi(\mathcal{X})$	data augmentation function applied to the input dataset $\mathcal{X}$ 53
$\Psi_{\text{flip}}(X)$	flips input image X with a chance of 50% 78
$\Psi_{\text{gamma}}(X, \gamma)$	gamma augmentation; brightens or darkens input image X by taking it to power $\gamma \sim U(0.5, 2)$ 78
$\Psi_{\text{rot}}(X, \alpha)$	rotates input image X by random angle $\alpha \sim U(0, 2\pi)$ 78
$\text{BN}(X; \gamma_{\text{BN}}, \beta_{\text{BN}})$	batch normalizing transform; normalizes feature map X by mean and variance and introduces trainable scaling and shifting parameters γ_{BN} and β_{BN} 24
$\text{FM}_i(X)$	outputs (feature maps) of i-th layer for input image X 34
$\text{UP}(X)$	upsampling function 23
$\varphi(x)$	non-linear activation function 24
$f(X; \theta)$	neural network with input X and parameters θ (may be omitted for clarity) 20
$l_i(X; \theta)$	i-th layer of neural network with parameters θ (may be omitted for clarity) 21

Collections and datasets

`AtlasBrainsCollection`

This collection contains 450 sections from the visual system of 7 brains labeled with 13 visual brain areas. 37

`B01Collection`

Subset of the `AtlasBrainsCollection`, comprising 62 sections from one brain. 37

`BigBrainCollection`

This collection contains 6000 sections of the BigBrain brain volume, including a 3D reconstruction between the sections. No labels of the visual system available. 39

$\mathcal{X}_{\text{AB}}$

Dataset for area segmentation containing randomly sampled patches X_{texture} from the `AtlasBrainsCollection` and X_{atlas} from the JuBrain atlas. 103

$\mathcal{X}_{\text{AB}\setminus\text{B01}}$

Dataset for area segmentation containing randomly sampled patches X_{texture} from the `AtlasBrainsCollection` without the `B01Collection` and X_{atlas} from the JuBrain atlas. 103

$\mathcal{X}_{\mathtt{a},s_1,s_2}$

Dataset for deep label propagation of area $\mathtt{a}$ between sections s_1 and s_2 of the `BigBrainCollection` containing randomly sampled 2 µm texture patches X_{hr} (high resolution) and 16 µm context patches X_{lr} (low resolution). 121

$\mathcal{X}_{\mathrm{B01}}$

Dataset for area segmentation containing randomly sampled patches X_{texture} from the `B01Collection` and X_{atlas} from the JuBrain atlas. 77

$\mathcal{X}_{\mathrm{B01-0}}$

Dataset for area segmentation containing randomly sampled patches X_{texture} from the `B01Collection` and atlas prior patches X_{atlas} containing zeros. Used for evaluating the influence of the atlas prior. 85

$\mathcal{X}_{\mathrm{BB}}$

Dataset for location and distance regression containing pairs of patches X_1 and X_2 sampled from the `BigBrainCollection`. $\mathcal{X}^{\mathrm{tr}}_{\mathrm{BB}}$ contains 500 000 patch pairs from the train/val split, and $\mathcal{X}^{\mathrm{te}}_{\mathrm{BB}}$ contains 2000 patch pairs from the test split. 96

$\mathcal{X}_{\mathrm{BBl}}$

Dataset for location and distance regression containing pairs of patches X_1 and X_2 sampled from the left hemisphere of the `BigBrainCollection`. $\mathcal{X}^{\mathrm{tr}}_{\mathrm{BBl}}$ contains 200 000 patch pairs from the train/val split, and $\mathcal{X}^{\mathrm{te}}_{\mathrm{BBl}}$ contains 150 000 patch pairs from the test split. 96

$\mathcal{X}_{\mathrm{BBr}}$

Dataset for location and distance regression containing pairs of patches X_1 and X_2 sampled from the right hemisphere of the `BigBrainCollection`. $\mathcal{X}^{\mathrm{tr}}_{\mathrm{BBr}}$ contains 200 000 patch pairs from the train/val split, and $\mathcal{X}^{\mathrm{te}}_{\mathrm{BBr}}$ contains 150 000 patch pairs from the test split. 96

$\mathcal{X}_{\mathrm{pr}}$

Dataset for profile classification containing 960 000 raw and smooth profiles p_{raw} and p_{smooth} extracted from the `B01Collection`. 64

$\mathcal{X}_{\mathrm{vis}}$

Dataset for location and distance regression containing pairs of patches X_1 and X_2 sampled from sections 0 to 3000 of the `BigBrainCollection`. $\mathcal{X}^{\mathrm{tr}}_{\mathrm{vis}}$ contains 200 000 patch pairs from the train/val split. 96

Neural network architectures and models

`AtlasAwareNetwork`

Two stream segmentation network combining texture and atlas prior input. 74

`BaseNetwork`

Basic segmentation network based on the U-net. Segments high-resolution image patches. 72

`MultiScaleNetwork`

Two stream segmentation network combining high- and low-resolution texture input to capture extended spatial context. 120

`ProfileNetwork`

1D CNN for classification of gray-value intensity profiles. 64

`SiameseNetwork`

Siamese network architecture for training self-supervised models on the distance and location regression task. 95

`atlas`

`AtlasAwareNetwork` with only the atlas prior path trained on rotation and gamma augmented dataset $\Psi_{\text{rot}}(\Psi_{\text{flip}}(\Psi_{\text{gamma}}(\mathcal{X}_{\text{B01}})))$ for area segmentation. 83

`atlas[siam-vis]`

`AtlasAwareNetwork` trained on $\mathcal{X}_{\text{B01}}$. Initialized from `SiameseNetwork` trained on $\mathcal{X}_{\text{vis}}$. 101

`base`

`BaseNetwork` trained on rotation and gamma augmented dataset $\Psi_{\text{rot}}(\Psi_{\text{flip}}(\Psi_{\text{gamma}}(\mathcal{X}_{\text{B01}})))$ for area segmentation. 82

`base-a-`s_1`-`s_2

`BaseNetwork` trained on patches sampled from sections s_1 and s_2 from the `BigBrainCollection` for the propagation of area `a` to sections $s_1+1, ..., s_2-1$. 123

`base-nogamma`

`BaseNetwork` trained on rotation augmented (but not gamma augmented) dataset $\Psi_{\text{rot}}(\Psi_{\text{flip}}(\mathcal{X}_{\text{B01}}))$ for area segmentation. 82

`base-norot`

: `BaseNetwork` trained on gamma augmented (but not rotation augmented) dataset $\Psi_{\text{gamma}}(\mathcal{X}_{\text{B01}})$ for area segmentation. 82

`base[siam-BB]`

: `BaseNetwork` trained on $\mathcal{X}_{\text{B01}}$. Initialized from a `SiameseNetwork` trained on $\mathcal{X}_{\text{BB}}$ inclucing all sections of the `BigBrainCollection`. 98

`base[siam-vis-dist]`

: `BaseNetwork` trained on $\mathcal{X}_{\text{B01}}$. Initialized from `siam-vis-dist`. 100

`base[siam-vis-loc]`

: `BaseNetwork` trained on $\mathcal{X}_{\text{B01}}$. Initialized from `siam-vis-loc`. 100

`base[siam-vis]`

: `BaseNetwork` trained on $\mathcal{X}_{\text{B01}}$. Initialized from a `siam-vis`. 98

`only-atlas`

: `AtlasAwareNetwork` with only the atlas prior path trained on rotation and gamma augmented dataset $\Psi_{\text{rot}}(\Psi_{\text{flip}}(\Psi_{\text{gamma}}(\mathcal{X}_{\text{B01}})))$ for area segmentation. 83

`profile-`w

: `ProfileNetwork` trained on $\mathcal{X}_{\text{pr}}$ with block size w for profile classification. 65

`prop-a-`s_1`-`s_2

: `MultiScaleNetwork` trained on patches sampled from sections s_1 and s_2 from the `BigBrainCollection` for the propagation of area `a` to sections $s_1+1,...,s_2-1$. 118

`siam-vis`

: `SiameseNetwork` trained on $\mathcal{X}_{\text{vis}}$. 99

`siam-vis-dist`

: `SiameseNetwork` trained on $\mathcal{X}_{\text{vis}}$ using only the distance loss L_{dist}. 99

`siam-vis-loc`

: `SiameseNetwork` trained on $\mathcal{X}_{\text{vis}}$ using only the location loss L_{coord}. 99

Bibliography

Abadi, M., A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, Chris Olah, Mike Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng (2015). *TensorFlow: large-scale machine learning on heterogeneous systems.* Software available from tensorflow.org. URL: `https://www.tensorflow.org/` (cit. on p. 81).

Abdi, H. and L. J. Williams (2010). "Principal component analysis". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 2.4, pp. 433–459 (cit. on p. 110).

Adamson, C., L. Johnston, T. Inder, S. Rees, I. Mareels, and G. Egan (2005). "A tracking approach to parcellation of the cerebral cortex". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2005.* Springer, pp. 294–301 (cit. on p. 43).

Agrawal, P., J. Carreira, and J. Malik (2015). "Learning to see by moving". In: *2015 IEEE International Conference on Computer Vision (ICCV).* IEEE, pp. 37–45 (cit. on p. 56).

Akkus, Z., A. Galimzianova, A. Hoogi, D. L. Rubin, and B. J. Erickson (2017). "Deep learning for brain MRI segmentation: state of the art and future directions". In: *Journal of Digital Imaging* 30.4, pp. 449–459 (cit. on p. 46).

Amunts, K., C. Lepage, L. Borgeat, H. Mohlberg, T. Dickscheid, M.-É. Rousseau, S. Bludau, P.-L. Bazin, L. B. Lewis, A.-M. Oros-Peusquens, et al. (2013). "BigBrain: an ultrahigh-resolution 3D human brain model". In: *Science* 340.6139, pp. 1472–1475 (cit. on pp. 2, 12, 39, 44, 61, 73, 110).

Amunts, K., A. Malikovic, H. Mohlberg, T. Schormann, and K. Zilles (2000). "Brodmann's areas 17 and 18 brought into stereotaxic space – where and how variable?" In: *NeuroImage* 11.1, pp. 66–84 (cit. on pp. 11, 18, 19, 68).

Amunts, K., A. Schleicher, and K. Zilles (May 2002). "Architectonic mapping of the human cerebral cortex". In: *Cortical Areas: Unity and Diversity* (cit. on p. 10).

– (2007). "Cytoarchitecture of the cerebral cortex – more than localization". In: *NeuroImage* 37.4, pp. 1061–1065 (cit. on pp. 1, 11, 18).

Amunts, K. and K. Zilles (2015). "Architectonic mapping of the human brain beyond Brodmann". In: *Neuron* 88.6, pp. 1086–1107 (cit. on pp. 1, 2, 11, 13, 14, 18, 45).

Atzeni, A., M. Jansen, S. Ourselin, and J. E. Iglesias (2018). "A probabilistic model combining deep learning and multi-atlas segmentation for semi-automated labelling of histology". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018.* Springer, pp. 219–227 (cit. on pp. 119, 132).

Azevedo, F. A., L. R. Carvalho, L. T. Grinberg, J. M. Farfel, R. E. Ferretti, R. E. Leite, W. J. Filho, R. Lent, and S. Herculano-Houzel (2009). “Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain”. In: *Journal of Comparative Neurology* 513.5, pp. 532–541 (cit. on p. 9).

Azizpour, H., A. Sharif Razavian, J. Sullivan, A. Maki, and S. Carlsson (2015). “From generic to specific deep representations for visual recognition”. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 36–45 (cit. on pp. 54, 98).

Barbas, H. (1986). “Pattern in the laminar origin of corticocortical connections”. In: *Journal of Comparative Neurology* 252.3, pp. 415–422 (cit. on p. 15).

Bayramoglu, N. and J. Heikkilä (2016). “Transfer learning for cell nuclei classification in histopathology images”. In: *Computer Vision – ECCV 2016*. Springer, pp. 532–539 (cit. on p. 54).

Bengio, Y. (2009). “Learning deep architectures for AI”. In: *Foundations and Trends in Machine Learning* 2.1, pp. 1–127 (cit. on p. 20).

Bengio, Y., A. Courville, and P. Vincent (2013). “Representation learning: a review and new perspectives”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8, pp. 1798–1828 (cit. on p. 55).

Bentaieb, A. and G. Hamarneh (2018). “Adversarial stain transfer for histopathology image analysis”. In: *IEEE Transactions on Medical Imaging* 37.3, pp. 792–802 (cit. on p. 50).

Beul, S. F., H. Barbas, and C. C. Hilgetag (2017). “A predictive structural model of the primate connectome”. In: *Scientific Reports* 7, p. 43176 (cit. on pp. 115, 135, 138).

Bludau, S. (2011). “Cytoarchitectonic mapping of the human frontal pole”. Dissertation. RWTH Aachen. URL: `http://juser.fz-juelich.de/record/15133` (cit. on p. 16).

Bok, S. (1929). “Der Einfluß der in den Furchen und Windungen auftretenden Krümmungen der Großhirnrinde auf die Rindenarchitektur”. In: *Zeitschrift für die gesamte Neurologie und Psychiatrie* 121, pp. 682–750 (cit. on pp. 62, 69, 107).

De Brébisson, A. and G. Montana (2015). “Deep neural networks for anatomical brain segmentation”. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 20–28 (cit. on pp. 46, 47).

Brodmann, K. (1909). *Vergleichende Lokalisationslehre der Grosshirnrinde in ihren Prinzipien dargestellt auf Grund des Zellenbaues*. Barth (cit. on p. 14).

Carneiro, G., J. Nascimento, and A. P. Bradley (2015). “Unregistered multiview mammogram analysis with pre-trained deep learning models”. In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2015*. Springer, pp. 652–660 (cit. on p. 54).

Caspers, J., K. Zilles, S. B. Eickhoff, A. Schleicher, H. Mohlberg, and K. Amunts (2013). “Cytoarchitectonical analysis and probabilistic mapping of two extrastriate areas of the human posterior fusiform gyrus”. In: *Brain Structure and Function* 218.2, pp. 511–526 (cit. on p. 19).

Caspers, S., S. B. Eickhoff, K. Zilles, and K. Amunts (2013). “Microstructural grey matter parcellation and its relevance for connectome analyses”. In: *NeuroImage* 80, pp. 18–26 (cit. on p. 18).

Chen, L.-C., G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille (2018). “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, pp. 834–848 (cit. on pp. 27, 89, 138).

Collins, C. E., D. C. Airey, N. A. Young, D. B. Leitch, and J. H. Kaas (2010). “Neuron densities vary across and within cortical areas in primates”. In: *Proceedings of the National Academy of Sciences* 107.36, pp. 15927–15932 (cit. on p. 15).

Couprie, M. and G. Bertrand (2004). “Topology preserving alternating sequential filter for smoothing two-dimensional and three-dimensional objects”. In: *Journal of Electronic Imaging* 13.4, pp. 720–731 (cit. on p. 129).

Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei (2009). “ImageNet: A Large-Scale Hierarchical Image Database”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 248–255 (cit. on pp. 19, 54).

Desikan, R. S., F. Ségonne, B. Fischl, B. T. Quinn, B. C. Dickerson, D. Blacker, R. L. Buckner, A. M. Dale, R. P. Maguire, B. T. Hyman, et al. (2006). “An automated labeling system for subdividing the human cerebral cortex on MRI scans into gyral based regions of interest”. In: *NeuroImage* 31.3, pp. 968–980 (cit. on p. 44).

Doersch, C., A. Gupta, and A. A. Efros (2015). “Unsupervised visual representation learning by context prediction”. In: *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, pp. 1422–1430 (cit. on pp. 1, 56, 57, 91).

Doersch, C. and A. Zisserman (2017). “Multi-task self-supervised visual learning”. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, pp. 2070–2079 (cit. on p. 57).

Dosovitskiy, A., P. Fischer, J. T. Springenberg, M. Riedmiller, and T. Brox (2016). “Discriminative unsupervised feature learning with exemplar convolutional neural networks”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.9, pp. 1734–1747 (cit. on p. 56).

Duyn, J. H. (2012). “The future of ultra-high field MRI and fMRI for study of the human brain”. In: *NeuroImage* 62.2, pp. 1241–1248 (cit. on p. 11).

Von Economo, C. F. and G. N. Koskinas (1925). *Die Cytoarchitektonik der Hirnrinde des erwachsenen Menschen*. J. Springer (cit. on p. 14).

Eickhoff, S. B., S. Heim, K. Zilles, and K. Amunts (2006). “Testing anatomically specified hypotheses in functional imaging using cytoarchitectonic maps”. In: *NeuroImage* 32.2, pp. 570–582 (cit. on p. 18).

Eickhoff, S. B., T. Paus, S. Caspers, M.-H. Grosbras, A. C. Evans, K. Zilles, and K. Amunts (2007). “Assignment of functional activations to probabilistic cytoarchitectonic areas revisited”. In: *NeuroImage* 36.3, pp. 511–521 (cit. on pp. 14, 18).

Van Essen, D. C. and D. L. Dierker (2007). “Surface-based and probabilistic atlases of primate cerebral cortex”. In: *Neuron* 56.2, pp. 209–225 (cit. on p. 73).

Esteva, A., B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun (2017). “Dermatologist-level classification of skin cancer with deep neural networks”. In: *Nature* 542.7639, p. 115 (cit. on pp. 1, 20, 54).

Everingham, M., L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman (2010). “The pascal visual object classes (VOC) challenge”. In: *International Journal of Computer Vision* 88.2, pp. 303–338 (cit. on p. 57).

Fischl, B., N. Rajendran, E. Busa, J. Augustinack, O. Hinds, B. T. Yeo, H. Mohlberg, K. Amunts, and K. Zilles (2007). “Cortical folding patterns and predicting cytoarchitecture”. In: *Cerebral Cortex* 18.8, pp. 1973–1980 (cit. on pp. 2, 73).

Fischl, B., D. H. Salat, E. Busa, M. Albert, M. Dieterich, C. Haselgrove, A. Van Der Kouwe, R. Killiany, D. Kennedy, S. Klaveness, et al. (2002). “Whole brain segmentation: automated labeling of neuroanatomical structures in the human brain”. In: *Neuron* 33.3, pp. 341–355 (cit. on p. 44).

Fischl, B., M. I. Sereno, and A. M. Dale (1999). “Cortical surface-based analysis: II: inflation, flattening, and a surface-based coordinate system”. In: *NeuroImage* 9.2, pp. 195–207 (cit. on p. 90).

Fukushima, K. and S. Miyake (1982). “Neocognitron: a self-organizing neural network model for a mechanism of visual pattern recognition”. In: *Competition and Cooperation in Neural Nets*. Springer, pp. 267–285 (cit. on p. 19).

Gao, Y., O. Beijbom, N. Zhang, and T. Darrell (2016). “Compact bilinear pooling”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 317–326 (cit. on p. 52).

Gidaris, S., P. Singh, and N. Komodakis (2018). “Unsupervised representation learning by predicting image rotations”. In: *International Conference on Learning Representations*. URL: `https://openreview.net/forum?id=S1v4N2l0-` (cit. on p. 56).

Glasser, M. F., T. S. Coalson, E. C. Robinson, C. D. Hacker, J. Harwell, E. Yacoub, K. Ugurbil, J. Andersson, C. F. Beckmann, M. Jenkinson, et al. (2016). “A multimodal parcellation of human cerebral cortex”. In: *Nature* 536.7615, pp. 171–178 (cit. on pp. 15, 47, 48).

Glorot, X., A. Bordes, and Y. Bengio (2011). “Deep sparse rectifier neural networks”. In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. Vol. 15, pp. 315–323 (cit. on pp. 24, 35).

Goldman, R. (2005). “Curvature formulas for implicit curves and surfaces”. In: *Computer Aided Geometric Design* 22.7, pp. 632–658 (cit. on p. 107).

Gonzalez, R. C. and R. E. Woods (Jan. 15, 2002). *Digital Image Processing, 2nd Edition*. 2nd Edition. Prentice Hall (cit. on p. 50).

Goodfellow, I., Y. Bengio, A. Courville, and Y. Bengio (2016). *Deep Learning*. Vol. 1. MIT press Cambridge (cit. on pp. 9, 23).

HBP Human Brain Atlas (2018). URL: `https://www.humanbrainproject.eu/en/explore-the-brain/atlases/` (cit. on pp. 6, 12, 129–131, 135).

He, K., X. Zhang, S. Ren, and J. Sun (2016). “Deep residual learning for image recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 770–778 (cit. on p. 1).

Heckemann, R. A., J. V. Hajnal, P. Aljabar, D. Rueckert, and A. Hammers (2006). "Automatic anatomical brain MRI segmentation combining label propagation and decision fusion". In: *NeuroImage* 33.1, pp. 115–126 (cit. on p. 44).

Hinton, G. E. (1986). "Learning distributed representations of concepts". In: *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*. Amherst, MA, pp. 1–12 (cit. on pp. 19, 21).

Hinton, G. E., S. Osindero, and Y.-W. Teh (2006). "A fast learning algorithm for deep belief nets". In: *Neural Computation* 18.7, pp. 1527–1554 (cit. on p. 55).

Holmes, C. J., R. Hoge, L. Collins, R. Woods, A. W. Toga, and A. C. Evans (1998). "Enhancement of MR images using registration for signal averaging". In: *Journal of Computer Assisted Tomography* 22.2, pp. 324–333 (cit. on p. 18).

Hornik, K. (1991). "Approximation capabilities of multilayer feedforward networks". In: *Neural Networks* 4.2, pp. 251–257 (cit. on p. 20).

Huntenburg, J. M., P.-L. Bazin, and D. S. Margulies (2017). "Large-scale gradients in human cortical organization". In: *Trends in Cognitive Sciences* (cit. on pp. 15, 110).

Ioffe, S. and C. Szegedy (July 2015). "Batch normalization: accelerating deep network training by reducing internal covariate shift". In: *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*. Vol. 37. PMLR, pp. 448–456 (cit. on pp. 23, 24).

Jayaraman, D. and K. Grauman (2015). "Learning image representations tied to egomotion". In: *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1413–1421 (cit. on p. 56).

Johnson, M., M. Schuster, Q. V. Le, M. Krikun, Y. Wu, Z. Chen, N. Thorat, F. Viégas, M. Wattenberg, G. Corrado, et al. (2017). "Google's multilingual neural machine translation system: enabling zero-shot translation". In: *Transactions of the Association of Computational Linguistics* 5.1, pp. 339–351 (cit. on p. 20).

Jones, S. E., B. R. Buchbinder, and I. Aharon (2000). "Three-dimensional mapping of cortical thickness using laplace's equation". In: *Human Brain Mapping* 11.1, pp. 12–32 (cit. on pp. 15, 60).

JuBrain Atlas Viewer (2014–2019). URL: `https://www.jubrain.fz-juelich.de` (cit. on pp. 18, 113).

Jülich Supercomputing Centre (2018). "JURECA: Modular supercomputer at Jülich Supercomputing Centre". In: *Journal of large-scale research facilities* 4.A132 (cit. on pp. 81, 123, 136).

JURON (2018). URL: `https://hbp-hpc-platform.fz-juelich.de/?page_id=1073` (cit. on pp. viii, 81).

Kainz, P., M. Pfeiffer, and M. Urschler (2017). "Segmentation and classification of colon glands with deep convolutional neural networks and total variation regularization". In: *PeerJ* 5, e3874 (cit. on p. 50).

Kamnitsas, K., C. Ledig, V. F. Newcombe, J. P. Simpson, A. D. Kane, D. K. Menon, D. Rueckert, and B. Glocker (2017). "Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation". In: *Medical Image Analysis* 36, pp. 61–78 (cit. on p. 53).

Kimia, B. B. and K. Siddiqi (1996). "Geometric heat equation and nonlinear diffusion of shapes and images". In: *Computer Vision and Image Understanding* 64.3, pp. 305–322 (cit. on p. 61).

Klein, A., B. Mensh, S. Ghosh, J. Tourville, and J. Hirsch (2005). "Mindboggle: automated brain labeling with multiple atlases". In: *BMC Medical Imaging* 5.1, p. 7 (cit. on p. 44).

Komura, D. and S. Ishikawa (2018). "Machine learning methods for histopathological image analysis". In: *Computational and Structural Biotechnology Journal* 16, pp. 34–42 (cit. on pp. 48, 49).

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). "ImageNet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, pp. 1097–1105 (cit. on pp. 19, 54).

Krogh, A. and J. A. Hertz (1992). "A simple weight decay can improve generalization". In: *Advances in Neural Information Processing Systems 4 (NIPS 1991)*, pp. 950–957 (cit. on p. 31).

Kujovic, M., K. Zilles, A. Malikovic, A. Schleicher, H. Mohlberg, C. Rottschy, S. B. Eickhoff, and K. Amunts (2013). "Cytoarchitectonic mapping of the human dorsal extrastriate cortex". In: *Brain Structure and Function* 218.1, pp. 157–172 (cit. on p. 19).

Lafferty, J. D., A. McCallum, and F. C. N. Pereira (2001). "Conditional random fields: probabilistic models for segmenting and labeling sequence data". In: *Proceedings of the 18th International Conference on Machine Learning (ICML-01)*. Morgan Kaufmann Publishers Inc., pp. 282–289 (cit. on p. 47).

Larsson, G., M. Maire, and G. Shakhnarovich (2016). "Learning representations for automatic colorization". In: *Computer Vision – ECCV 2016*. Springer, pp. 577–593 (cit. on p. 56).

LeCun, Y., Y. Bengio, and G. Hinton (2015). "Deep learning". In: *Nature* 521.7553, p. 436 (cit. on p. 20).

Lee, H.-Y., J.-B. Huang, M. Singh, and M.-H. Yang (2017). "Unsupervised representation learning by sorting sequences". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, pp. 667–676 (cit. on p. 56).

Lewis, L., C. Lepage, M. Fournier, K. Zilles, K. Amunts, and A. Evans (2014). "BigBrain: initial tissue classification and surface extraction". In: *20th Annual Meeting of the Organization for Human Brain Mapping (OHBM)*. URL: `https://f1000research.com/posters/1096350` (cit. on pp. 39, 95).

Li, Y. and W. Ping (2018). "Cancer metastasis detection with neural conditional random field". In: *1st International Conference on Medical Imaging in Deep Learning*. URL: `https://openreview.net/forum?id=S1aY66iiM` (cit. on p. 89).

Lin, T.-Y., A. RoyChowdhury, and S. Maji (2015). "Bilinear CNN models for fine-grained visual recognition". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE, pp. 1449–1457 (cit. on p. 52).

Liu, L., J. Chen, P. Fieguth, G. Zhao, R. Chellappa, and M. Pietikainen (2018). "A survey of recent advances in texture representation". In: *arXiv preprint arXiv:1801.10324* (cit. on p. 52).

Liu, Y., K. Gadepalli, M. Norouzi, G. E. Dahl, T. Kohlberger, A. Boyko, S. Venugopalan, A. Timofeev, P. Q. Nelson, G. S. Corrado, et al. (2017). “Detecting cancer metastases on gigapixel pathology images”. In: *arXiv preprint arXiv:1703.02442* (cit. on pp. 50, 51, 53).

Long, J., E. Shelhamer, and T. Darrell (2015). “Fully convolutional networks for semantic segmentation”. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 3431–3440 (cit. on pp. 27, 54).

Lorenz, S., K. S. Weiner, J. Caspers, H. Mohlberg, A. Schleicher, S. Bludau, S. B. Eickhoff, K. Grill-Spector, K. Zilles, and K. Amunts (2015). “Two new cytoarchitectonic areas on the human mid-fusiform gyrus”. In: *Cerebral Cortex* 27.1, pp. 373–385 (cit. on p. 19).

Madabhushi, A. and G. Lee (2016). “Image analysis and machine learning in digital pathology: challenges and opportunities”. In: *Medical Image Analysis* 33, pp. 170–175 (cit. on p. 49).

Magnain, C., J. C. Augustinack, M. Reuter, C. Wachinger, M. P. Frosch, T. Ragan, T. Akkin, V. J. Wedeen, D. A. Boas, and B. Fischl (2014). “Blockface histology with optical coherence tomography: a comparison with Nissl staining”. In: *NeuroImage* 84, pp. 524–533 (cit. on p. 12).

Mahajan, D., R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten (2018). “Exploring the limits of weakly supervised pretraining”. In: *Computer Vision – ECCV 2018*. Springer, pp. 185–201 (cit. on pp. 54, 98).

Mahalanobis, P. C. (1936). “On the generalized distance in statistics”. In: *Proceedings of the National Institute of Science of India*. Vol. 12, pp. 49–55 (cit. on p. 15).

Malandain, G. and E. Bardinet (2003). “Intensity compensation within series of images”. In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2003*. Springer, pp. 41–49 (cit. on pp. 50, 51).

Malikovic, A., K. Amunts, A. Schleicher, H. Mohlberg, S. B. Eickhoff, M. Wilms, N. Palomero-Gallagher, E. Armstrong, and K. Zilles (2006). “Cytoarchitectonic analysis of the human extrastriate cortex in the region of V5/MT+: a probabilistic, stereotaxic map of area hOc5”. In: *Cerebral Cortex* 17.3, pp. 562–574 (cit. on p. 19).

Malikovic, A., K. Amunts, A. Schleicher, H. Mohlberg, M. Kujovic, N. Palomero-Gallagher, S. B. Eickhoff, and K. Zilles (2016). “Cytoarchitecture of the human lateral occipital cortex: mapping of two extrastriate areas hOc4la and hOc4lp”. In: *Brain Structure and Function* 221.4, pp. 1877–1897 (cit. on p. 19).

Mangin, J.-F., J. Lebenberg, S. Lefranc, N. Labra, G. Auzias, M. Labit, M. Guevara, H. Mohlberg, P. Roca, P. Guevara, et al. (2016). “Spatial normalization of brain images and beyond”. In: *Medical Image Analysis* 33, pp. 127–133 (cit. on p. 45).

Mangin, J.-F., D. Riviere, A. Cachia, E. Duchesnay, Y. Cointepas, D. Papadopoulos-Orfanos, P. Scifo, T. Ochiai, F. Brunelle, and J. Regis (2004). “A framework to study the cortical folding patterns”. In: *NeuroImage* 23, S129–S138 (cit. on p. 2).

Merker, B. (1983). “Silver staining of cell bodies by means of physical development”. In: *Journal of Neuroscience Methods* 9.3, pp. 235–241 (cit. on pp. 11, 50).

Moeskops, P., M. A. Viergever, A. M. Mendrik, L. S. de Vries, M. J. Benders, and I. Išgum (2016). “Automatic segmentation of MR brain images with a convolutional neural network”. In: *IEEE Transactions on Medical Imaging* 35.5, pp. 1252–1261 (cit. on pp. 46, 47).

Mormont, R., P. Geurts, and R. Marée (2018). “Comparison of deep transfer learning strategies for digital pathology”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE (cit. on p. 54).

Nair, V. and G. E. Hinton (2010). “Rectified linear units improve restricted boltzmann machines”. In: *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*. Omnipress, pp. 807–814 (cit. on p. 20).

Nesterov, Y. E. (1983). “A method for solving the convex programming problem with convergence rate o(1/k$\hat{2}$)”. In: *Soviet Mathematics Doklady*. Vol. 27, pp. 372–376 (cit. on p. 29).

Noroozi, M. and P. Favaro (2016). “Unsupervised learning of visual representations by solving jigsaw puzzles”. In: *Computer Vision – ECCV 2016*. Springer, pp. 69–84 (cit. on pp. 1, 56, 91).

Oden, L., C. Schiffer, H. Spitzer, T. Dickscheid, and D. Pleiter (2019). “Io challenges for human brain atlasing using deep learning methods - an in-depth analysis”. In: *The 27th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, 13-15 February 2019 in Pavia, Italy.* (Cit. on p. 81).

Olshausen, B. A. and D. J. Field (1996). “Emergence of simple-cell receptive field properties by learning a sparse code for natural images”. In: *Nature* 381.6583, p. 607 (cit. on p. 55).

Owens, A., J. Wu, J. H. McDermott, W. T. Freeman, and A. Torralba (2016). “Ambient sound provides supervision for visual learning”. In: *Computer Vision – ECCV 2016*. Springer, pp. 801–816 (cit. on p. 56).

Pathak, D., P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros (2016). “Context encoders: feature learning by inpainting”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 2536–2544 (cit. on p. 56).

Pereira, S., A. Pinto, V. Alves, and C. A. Silva (2016). “Brain tumor segmentation using convolutional neural networks in MRI images”. In: *IEEE Transactions on Medical Imaging* 35.5, pp. 1240–1251 (cit. on p. 53).

Pichat, J., J. E. Iglesias, T. Yousry, S. Ourselin, and M. Modat (2018). “A survey of methods for 3D histology reconstruction”. In: *Medical Image Analysis* 46, pp. 73–105 (cit. on pp. 2, 17, 50).

Rohlfing, T., R. Brandt, R. Menzel, D. B. Russakoff, and C. R. Maurer (2005). “Quo vadis, atlas-based segmentation?” In: *Handbook of Biomedical Image Analysis*. Springer, pp. 435–486 (cit. on pp. 44, 119).

Ronneberger, O., P. Fischer, and T. Brox (2015). “U-net: convolutional networks for biomedical image segmentation”. In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2015*. Springer, pp. 234–241 (cit. on pp. 23, 27, 28, 53, 72).

Rosenblatt, F. (1961). *Principles of neurodynamics. Perceptrons and the theory of brain mechanisms.* Tech. rep. Cornell Aeronautical Laboratory (cit. on p. 19).

Rottschy, C., S. B. Eickhoff, A. Schleicher, H. Mohlberg, M. Kujovic, K. Zilles, and K. Amunts (2007). “Ventral visual cortex in humans: cytoarchitectonic mapping of two extrastriate areas”. In: *Human Brain Mapping* 28.10, pp. 1045–1059 (cit. on p. 19).

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). “Learning representations by back-propagating errors”. In: *Nature* 323.6088, p. 533 (cit. on p. 30).

Schiffer, C. (2018). “Anwendungsmöglichkeiten von Unsupervised Domain Adaptation mit Convolutional Neural Networks zur automatischen Kartierung menschlicher Hirnareale in Mikroskopaufnahmen histologischer Gewebeschnitte”. MA thesis. FH Jülich. URL: `http://juser.fz-juelich.de/record/857238` (cit. on p. 40).

Schleicher, A., K. Amunts, S. Geyer, P. Morosan, and K. Zilles (1999). “Observer-independent method for microstructural parcellation of cerebral cortex: a quantitative approach to cytoarchitectonics”. In: *NeuroImage* 9.1, pp. 165–177 (cit. on pp. 1, 5, 14–16, 18, 37, 43, 59–61, 68, 117).

Schleicher, A., N. Palomero-Gallagher, P. Morosan, S. Eickhoff, T. Kowalski, K. De Vos, K. Amunts, and K. Zilles (2005). “Quantitative architectural analysis: a new approach to cortical mapping”. In: *Anatomy and Embryology* 210.5-6, pp. 373–386 (cit. on pp. 11, 14).

Selvaraju, R. R., M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra (2017). “Grad-CAM: visual explanations from deep networks via gradient-based localization.” In: *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, pp. 618–626 (cit. on pp. 34, 67).

Sergeev, A. and M. D. Balso (2018). “Horovod: fast and easy distributed deep learning in TensorFlow”. In: *arXiv preprint arXiv:1802.05799* (cit. on p. 123).

Sharif Razavian, A., H. Azizpour, J. Sullivan, and S. Carlsson (2014). “CNN features off-the-shelf: an astounding baseline for recognition”. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 806–813 (cit. on p. 36).

Shin, H.-C., H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers (2016). “Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning”. In: *IEEE Transactions on Medical Imaging* 35.5, pp. 1285–1298 (cit. on p. 54).

Silver, D., A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. (2016). “Mastering the game of Go with deep neural networks and tree search”. In: *Nature* 529.7587, p. 484 (cit. on pp. 1, 20).

Snoek, J., H. Larochelle, and R. P. Adams (2012). “Practical bayesian optimization of machine learning algorithms”. In: *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, pp. 2951–2959 (cit. on p. 31).

Spitzer, H., K. Amunts, S. Harmeling, and T. Dickscheid (2017). “Parcellation of visual cortex on high-resolution histological brain sections using convolutional neural networks”. In: *2017 IEEE 14th International Symposium on Biomedical Imaging (ISBI 2017)*. IEEE, pp. 920–923 (cit. on pp. 54, 71, 72, 76, 81, 141).

Spitzer, H., K. Amunts, S. Harmeling, and T. Dickscheid (2018). "Compact feature representations for human brain cytoarchitecture using self-supervised learning". In: *1st International Conference on Medical Imaging in Deep Learning*. URL: `https://openreview.net/forum?id=HyjJ812of` (cit. on pp. 91, 106, 112, 142).

Spitzer, H., K. Kiwitz, K. Amunts, S. Harmeling, and T. Dickscheid (2018). "Improving cytoarchitectonic segmentation of human brain areas with self-supervised siamese networks". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*. Springer, pp. 663–671 (cit. on pp. 91, 92, 95, 98, 141, 142).

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov (2014). "Dropout: a simple way to prevent neural networks from overfitting". In: *The Journal of Machine Learning Research* 15.1, pp. 1929–1958 (cit. on p. 20).

Sutskever, I., J. Martens, G. Dahl, and G. Hinton (2013). "On the importance of initialization and momentum in deep learning". In: *Proceedings of the 27th International Conference on Machine Learning (ICML-13)*. PMLR, pp. 1139–1147 (cit. on pp. 29, 30).

Tajbakhsh, N., J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang (2016). "Convolutional neural networks for medical image analysis: full training or fine tuning?" In: *IEEE Transactions on Medical Imaging* 35.5, pp. 1299–1312 (cit. on p. 54).

Toga, A. W. and P. M. Thompson (2003). "Mapping brain asymmetry". In: *Nature Reviews Neuroscience* 4.1, p. 37 (cit. on p. 94).

Trepel, M. (2004). *Neuroanatomie: Struktur und Funktion (3. neubearbeitete Aufl.)* Urban & Fischer Verlag/Elsevier GmbH (cit. on pp. 13, 14, 19).

Tuceryan, M. and A. K. Jain (1993). "Texture analysis". In: *Handbook of Pattern Recognition and Computer Vision*. World Scientific, pp. 235–276 (cit. on p. 52).

Vincent, P., H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol (2010). "Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion". In: *Journal of Machine Learning Research* 11.12, pp. 3371–3408 (cit. on p. 55).

Vogt, C. and 0. Vogt (1919). "Allgemeinere ergebnisse unserer hirnforschung (english translation: results of our brain research in a broader context)". In: *Journal für Psychologie und Neurologie* 25.I, pp. 292–398 (cit. on p. 14).

Wachinger, C., M. Reuter, and T. Klein (2017). "DeepNAT: deep convolutional neural network for segmenting neuroanatomy". In: *NeuroImage* (cit. on pp. 46, 47).

Waehnert, M., J. Dinse, M. Weiss, M. N. Streicher, P. Waehnert, S. Geyer, R. Turner, and P.-L. Bazin (2014). "Anatomically motivated modeling of cortical laminae". In: *NeuroImage* 93, pp. 210–220 (cit. on pp. 2, 69).

Wagstyl, K., C. Lepage, S. Bludau, K. Zilles, P. C. Fletcher, K. Amunts, and A. C. Evans (2018). "Mapping cortical laminar structure in the 3D BigBrain". In: *Cerebral Cortex* 28.7, pp. 2551–2562 (cit. on pp. 44, 61).

Wagstyl, K., L. Ronan, I. M. Goodyer, and P. C. Fletcher (2015). "Cortical thickness gradients in structural hierarchies". In: *NeuroImage* 111, pp. 241–250 (cit. on pp. 15, 107, 108).

Wang, C., J. Shi, Q. Zhang, and S. Ying (2017). “Histopathological image classification with bilinear convolutional neural networks”. In: *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC),* IEEE, pp. 4050–4053 (cit. on p. 52).

Wang, D., A. Khosla, R. Gargeya, H. Irshad, and A. H. Beck (2016). “Deep learning for identifying metastatic breast cancer”. In: *arXiv preprint arXiv:1606.05718* (cit. on pp. 49, 50).

Wang, H. and P. Yushkevich (2013). “Multi-atlas segmentation with joint label fusion and corrective learning—an open source implementation”. In: *Frontiers in Neuroinformatics* 7, p. 27 (cit. on p. 44).

Wang, X. and A. Gupta (2015). “Unsupervised learning of visual representations using videos”. In: *2015 IEEE International Conference on Computer Vision (ICCV).* IEEE, pp. 2794–2802 (cit. on p. 56).

Ward Jr, J. H. (1963). “Hierarchical grouping to optimize an objective function”. In: *Journal of the American Statistical Association* 58.301, pp. 236–244 (cit. on p. 112).

Wei, Y., L. H. Scholtens, E. Turk, and M. P. van den Heuvel (2018). “Multiscale examination of cytoarchitectonic similarity and human brain connectivity”. In: *Network Neuroscience* 3.1, pp. 124–137 (cit. on p. 138).

Xu, Y., Z. Jia, L.-B. Wang, Y. Ai, F. Zhang, M. Lai, I. Eric, and C. Chang (2017). “Large scale tissue histopathology image classification, segmentation, and visualization via deep convolutional activation features”. In: *BMC Bioinformatics* 18.1, p. 281 (cit. on pp. 50, 54).

Yosinski, J., J. Clune, Y. Bengio, and H. Lipson (2014). “How transferable are features in deep neural networks?” In: *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, pp. 3320–3328 (cit. on pp. 1, 35, 54, 98).

Yu, F. and V. Koltun (2015). “Multi-scale context aggregation by dilated convolutions”. In: *arXiv preprint arXiv:1511.07122* (cit. on p. 27).

Zanjani, F. G., S. Zinger, B. E. Bejnordi, J. A. van der Laak, et al. (2018). “Stain normalization of histopathology images using generative adversarial networks”. In: *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018).* IEEE, pp. 573–577 (cit. on p. 50).

Zhang, R., P. Isola, and A. A. Efros (2016). “Colorful image colorization”. In: *Computer Vision – ECCV 2018.* Springer, pp. 649–666 (cit. on p. 56).

Zheng, S., S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr (2015). “Conditional random fields as recurrent neural networks”. In: *2015 IEEE International Conference on Computer Vision (ICCV).* IEEE, pp. 1529–1537 (cit. on p. 89).

Zilles, K. and N. Palomero-Gallagher (2015). “Gyrification in the human brain”. In: *Brain Mapping.* Ed. by A. W. Toga. Waltham: Academic Press, pp. 37–44 (cit. on p. 10).

Zilles, K. and K. Amunts (2010). “Centenary of brodmann’s map – conception and fate”. In: *Nature Reviews Neuroscience* 11.2, pp. 139–145 (cit. on pp. 14, 18).

Zilles, K. and K. Amunts (2012). “Architecture of the cerebral cortex”. In: *The Human Nervous System (Third Edition)*. Elsevier, pp. 836–895 (cit. on pp. 9, 10).

Zilles, K., A. Dabringhaus, S. Geyer, K. Amunts, M. Qü, A. Schleicher, E. Gilissen, G. Schlaug, and H. Steinmetz (1996). “Structural asymmetries in the human forebrain and the forebrain of non-human primates and rats”. In: *Neuroscience & Biobehavioral Reviews* 20.4, pp. 593–605 (cit. on p. 94).

Zilles, K., A. Schleicher, C. Langemann, K. Amunts, P. Morosan, N. Palomero-Gallagher, T. Schormann, H. Mohlberg, U. Bürgel, H. Steinmetz, et al. (1997). “Quantitative analysis of sulci in the human cerebral cortex: development, regional heterogeneity, gender difference, asymmetry, intersubject variability and cortical architecture”. In: *Human Brain Mapping* 5.4, pp. 218–221 (cit. on p. 2).

Zitova, B. and J. Flusser (2003). “Image registration methods: a survey”. In: *Image and Vision Computing* 21.11, pp. 977–1000 (cit. on p. 16).

Eidesstattliche Erklärung

Ich versichere an Eides Statt, dass die Dissertation von mir selbständig und ohne unzulässige fremde Hilfe unter Beachtung der “Grundsätze zur Sicherung guter wissenschaftlicher Praxis an der Heinrich-Heine-Universität Düsseldorf" erstellt worden ist.

Ort, Datum	Unterschrift