Diss. ETH No. 24028

Towards expert-aware computer vision algorithms in medical imaging

A dissertation submitted to
ETH Zurich

for the degree of
Doctor of Sciences

presented by
Dmitry Laptev
Dipl. Inf., Moscow State University
born 3 February 1989
citizen of Russian Federation

accepted on the recommendation of
Prof. Dr. Joachim M. Buhmann, examiner
Prof. Dr. Marc Pollefeys, co-examiner
Prof. Dr. med. Adriano Aguzzi, co-examiner

2016

Abstract

The field of computer vision experienced a significant breakthrough in image categorization and semantic segmentation during recent years. In some natural image recognition tasks, especially in supervised learning setting, modern algorithms, for example deep convolutional neural networks, achieve results comparable to human performance. This progress is due to very powerful models trained on large enough data sets.

While in general this progress is very beneficial for the conceptual development of computer vision, it showed rather limited impact on highly specialized tasks, especially when they arise from a very narrow domain, or with very few (if any at all) training samples. In this thesis we consider computer vision tasks with such limitations, focusing mostly on medical imaging problems.

The key to advance in these tasks is to develop *expert-aware algorithms*, i.e. to incorporate strong prior knowledge from field experts into modern algorithmic pipelines.

In case of supervised learning, our approaches result in significant accuracy gains and performance improvements when compared to conventional models. In case of unsupervised or weakly supervised learning, we introduce a framework that requires no training and leads to reasonable non-subjective solutions based only on expert priors. In regard to applications, we focus on medical imaging, video data and natural image analysis.

Overall in this thesis we focus on three types of expert-aware al-

gorithms. These algorithms are based on: (i) weakly specified expert approach to the problem, e.g. how additional data can be used to resolve ambiguities, (ii) variety of facts about the input data itself, e.g. known invariances in the data, and (iii) facts about the desired properties of a solution, e.g. detected objects statistics.

The first type of algorithms mimic the style of experts when they solve a problem at hand. As a main example we show how to adapt segmentation and restoration algorithms to deal with both spatial and temporal "anisotropy" – a characteristic property of image data that is very common in the domain of medical imaging and in video data. Similar to experts resolving ambiguities from related samples, we resolve the correspondences between different data points and reinforce the patterns learned across samples.

The second set of approaches presented in this thesis allows one to develop computer vision detection algorithms using no training data at all, just based on some properties of the output defined by experts. In medical domain problems these properties can be formulated from known biological facts about the imaged objects. We present a framework that enables the user to employ these biologically motivated priors for tuning internal parameters of an algorithm, and, thereby, deriving a non-subjective final solution. This thesis demonstrates the power of this approach by applying it to a challenging task of large-scale senile plaques segmentation.

The third type of algorithms addresses a more powerful feature learning process by incorporating invariance of the imaged objects to a specific set of expert-formulated transformations. We demonstrate this approach by applying it to both a very efficient greedy algorithm and to highly flexible deep learning architectures. We do not only experimentally demonstrate that this approach works superior when applied to a narrow-domain application, but also show that even general computer vision problems can benefit from incorporating common-sense knowledge about nuisance variations in the world.

Zusammenfassung

Die Forschung im Bereich maschinelles Sehen hat in den vergangenen Jahren bedeutende Durchbrüche bei der Kategorisierung und semantischen Segmentierung von Bildern feiern können. Moderne Algorithmen wie tiefe neuronale Konvolutionsnetzwerke erreichen in manchen, insbesonders überwachten, Bilderkennungsaufgaben ein Leistungsniveau, das menschlichen Fähigkeiten in nichts nachsteht. Dieser Fortschritt beruht vor allem auf komplexen Modellen, die auf genügend grossen Datensätzen trainiert werden.

Obwohl sich dieser Fortschritt für die konzeptuelle Weiterentwicklung der Bilderkennung als sehr hilfreich erwies, beeinflusst er das Leistungsniveau bei höchst spezialisierten Aufgaben eher wenig. Insbesondere bei Aufgaben in Nischenbereichen mit sehr wenigen (bis keinen) Trainingsbeispielen erweisen sich die tiefen neuronalen Netze als weit weniger effektiv als bei den Standardanwendungen. In dieser Dissertation lösen wir Probleme des maschinellen Sehens mit solchen Einschränkungen, wobei die entwickelten Methoden schwerpunktmässig auf Bildern aus dem medizinischen Bereich getestet werden.

Um solche Aufgaben zu lösen, müssen effiziente Algorithmen entwickelt und diese mit Expertenwissen ausgestattet werden, das heisst moderne Algorithmen mit dem a-priori Wissen von Experten eines Bereichs zu erweitern.

Im Fall des überwachten Lernens führt unser Ansatz zu einer signifikanten Steigerung der Genauigkeit und Leistung im Vergleich zu konventionellen Modellen. Für nicht überwachtes bzw. wenig

überwachtes Lernen stellen wir ein System vor, das ohne Training zufriedenstellende, nicht subjektive Ergebnisse liefert, die nur auf a-priori Wissen von Experten beruhen. Im Bezug auf Anwendungen konzentrieren wir uns auf medizinische Bilder, Videos und andere Bilder.

Insgesamt werden in dieser Arbeit drei Arten von Experten-gesteuerten Algorithmen behandelt. Diese Algorithmen basieren auf: (i) schwach spezifizierte Expertenlösungen für ein Problem, d.h. wie zusätzliche Daten Unklarheiten beseitigen können; (ii) verschiedenen Fakten über die Daten selbst, z.b. bekannte Symmetrien geschätzt werden können; (iii) Fakten bezüglich der gewünschten Eigenschaften einer Lösung, z.b. statistische Eigenschaften von den zu erkennenden Objekten.

Der erste Typ von Algorithmen ahmt die Herangehensweise von Experten nach, wenn diese ein konkretes Problem lösen. Als Hauptbeispiel zeigen wir, wie man Segmentierungs- und Restaurationsalgorithmen anpassen kann, um mit örtlicher und zeitlicher Anisotropie umgzugehen, welche eine charakteristische Eigenschaft von medizinischen Bilddaten und Videodaten ist. So wie Experten Unklarheiten beseitigen, indem sie ähnliche Beispiel betrachten, benutzen wir die Korrespondenz zwischen verschiedenen Datenpunkten um verstärkt Beispiel-übergreifende Muster zu lernen.

Die zweite Klasse von Methoden, die in dieser Arbeit präsentiert werden, erlauben Objekterkennung ganz ohne Trainingsdaten zu entwickeln, nur basierend auf Lösungseigenschaften, die von Experten gegeben sind. Im medizinischen Bereich können diese Eigenschaften von bekannten biologischen Fakten der abgebildeten Objekte abgeleitet werden. Wir stellen ein System vor, das den Benutzer befähigt, solch biologisches a-priori Wissen zum Einstellen von internen Parametern eines Algorithmus zu verwenden, sodass eine nicht subjektive Lösung erreicht werden kann. Diese Arbeit demonstriert die Wirksamkeit dieser Methode anhand der herausfordernden Aufgabe im Hochdurchsatzverfahren Plaques zu segmentieren.

Die dritte Art von Algorithmen behandelt das verbesserte Lernen von Merkmalen, indem Symmetrien der abgebildeten Objekte mitein-

bezogen werden, die durch eine spezifische Menge von Experten-definierten Transformationen gegeben sind. Wir demonstrieren diese Methode einerseits an einem sehr effizienten Greedy-Verfahren und andererseits an den höchst flexiblen Architekturen von tiefen neuronalen Netzwerken. Wir zeigen experimentell, dass diese Herangehensweise nicht nur in Nischenbereichen überlegen funktioniert, sondern auch, dass man allgemeine Kenntnisse über Störungen und Variationen der Welt, wie wir sie wahrnehmen, gewinnbringend für die Lösung von Problemen des maschinellen Sehens nutzen kann.

List of publications

Parts of this thesis have been published in the following papers:

- Dmitry Laptev, Alexander Vezhnevets, Sarvesh Dwivedi, Joachim M. Buhmann. Anisotropic ssTEM image segmentation using dense correspondence across sections. *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2012*, pages 323–330, 2012.

- Dmitry Laptev, Alexander Vezhnevets, Joachim M. Buhmann. SuperSlicing frame restoration for anisotropic ssTEM. *IEEE 11th International Symposium on Biomedical Imaging–ISBI 2014*, pages 1198–1201, 2014.

- Dmitry Laptev, Joachim M. Buhmann. SuperSlicing frame restoration for anisotropic ssTEM and video data. *ECML 2014 Neural Connectomics workshop – NCW ECML 2014, JMLR Workshop and Conference Proceedings 46*, pages 93–103, 2015.

- Dmitry Laptev, Joachim M. Buhmann. Convolutional Decision Trees for feature learning and segmentation. *German Conference on Pattern Recognition – GCPR 2014, Pattern Recognition 8753*, pages 95–106, 2014.

- Dmitry Laptev, Joachim M. Buhmann. Transformation-Invariant Convolutional Jungles. *IEEE International Conference on Computer Vision and Pattern Recognition – CVPR 2015*, pages 3043–3051, 2015.

- Ignacio Arganda-Carreras, Srinivas C. Turaga, Daniel R Berger, Dan Cireşan, Alessandro Giusti, Luca M. Gambardella, Jürgen Schmidhuber, Dmitry Laptev, Sarvesh Dwivedi, Joachim M. Buhmann, et al. Crowdsourcing the creation of image segmentation algorithms for connectomics. *Frontiers in Neuroanatomy 9, 2015*, article 142.

- Dmitry Laptev, Nikolay Savinov, Joachim M. Buhmann, Marc Pollefeys. TI-Pooling: transformation-invariant pooling for feature learning in Convolutional Neural Networks. *IEEE/CVF International Conference on Computer Vision and Pattern Recognition – CVPR 2016.*

- Dmitry Laptev, Daniel Kirschenbaum, Joachim M. Buhmann, Adriano Aguzzi. Biologically-motivated priors for non-parametric plaque analysis in mouse brains. *The paper in preparation.*

- Daniel Kirschenbaum, Oliver Bichsel, Dmitry Laptev, Michael B. Smith, Fabian F. Voigt, Joachim M. Buhmann, Adriano Aguzzi. Rapid electrophoretic tissue clearing and molecular labelling in whole mount mouse brain. *The manuscript in preparation.*

To my grandparents and parents

Acknowledgments

First and foremost, I would like to thank my supervisor, Joachim M. Buhmann, for giving me the opportunity to pursue my doctoral studies in one of the best environments imaginable. He taught me to see the bigger picture and to ask the right scientific questions. Joachim advised and supported me in both scientific challenges and in private matters. I would like to especially thank him for giving me opportunity to carry out my own research, balancing the sufficient freedom with useful guidance from his side when needed.

I would like to thank Marc Pollefeys and Adriano Aguzzi for agreeing to review my thesis. My interactions with them and their students always broadened my understanding of the field and the problem at hand, and resulted in significant shifts towards the solution of these problems. It is an honour to have such distinguished scientists in my examination committee.

I am also very grateful to other collaborators whom I had a chance to learn from during my years in ETH. To name a few, Alexander Vezhnevets got me up to speed in the very beginning of my studies, Richard Hahnloser and Jan Funke introduced me to the field of Connectomics, which appeared to be a major application of my future research. Pascal Fua and his students helped me to advance my ideas and broaden my understanding of medical applications of computer vision. Brian McWilliams introduced me to the mathematics behind modern machine learning techniques. Nikolay Savinov offered his expertise in the field of neural networks. Daniel Kirschenbaum

was extremely involved in every step of the plaque analysis pipeline development – he is a medical collaborator a computer scientist can usually only hope for.

I would like to separately thank Dmitry Vetrov, my scientific adviser and mentor from Lomonosov Moscow State University. He showed me the beauty of applied mathematics, made me interested in machine learning and learned me how to carry out research up to the highest international standards.

Many thanks to all the members of the Machine Learning Institute at ETH Zurich. I have learned so much from the people around and from their diverse backgrounds, both in science and in real life. They became not only my colleagues, but friends. Very special thanks goes to Rita Klute – she is the heart of the group that keeps it running.

My deepest gratitude I would like to address to my family and friends for all the support. Special thanks to Stefan, Martin, Alex and Viktor, who helped to proofread this thesis. My loving wife Valentina, my parents, grandparents and my sister are the reasons why I chose this path in my life and why I managed to achieve my goals. My father Anatoly and my grandfather Boris are the two most important scientists in my life, who always inspire me to learn and push the boundaries of known. I thank my friends in Russia, Switzerland and all over the world for all their support and all the fun we have together. It is my greatest pleasure to share the important moments of my life and everyday experiences with you.

Contents

1 Introduction **1**

1.1 Background and motivation 1

1.1.1 Modern supervised computer vision 1

1.1.2 Open challenges 3

1.2 Towards expert-aware algorithms 4

1.2.1 Types of expert prior knowledge 4

1.2.2 Objectives and contributions 5

1.3 Areas of application 7

1.4 Structure of the thesis 8

2 Anisotropic data **11**

2.1 Introduction 11

2.1.1 Anisotropic data definition 12

2.1.2 Anisotropic data and connectomics 15

2.1.3 Related work 18

2.2 Segmentation using dense correspondence 20

2.2.1 Method description 22

2.2.2 Experiments 29

2.3 SuperSlicing frame restoration 31

2.3.1 Reconstruction method description 33

2.3.2 SuperSlicing for neuronal segmentation . . . 37

2.3.3 Experiments 39

2.4 Contributions 44

3 Global biological priors **47**
3.1 Amyloid plaques in cleared mouse brains 48
3.1.1 Motivation 48
3.1.2 Tissue clearing and imaging 49
3.1.3 Local analysis 50
3.2 Biologically motivated priors for tuning-free algorithms 53
3.2.1 Whole-brain analysis 53
3.2.2 Related work 57
3.2.3 Method description 58
3.2.4 Feedback-loop for parameters tuning 61
3.2.5 Experimental results 64
3.3 Contributions 66

4 Transformation-invariance **67**
4.1 Introduction 67
4.1.1 Feature learning 68
4.1.2 Transformation-invariance 70
4.2 Convolutional Decision Trees 72
4.2.1 Related work 73
4.2.2 Method description: split 76
4.2.3 Method description: tree 81
4.2.4 Experiments 83
4.3 Transformation-Invariant Jungles 88
4.3.1 Related work 91
4.3.2 Method description 93
4.3.3 Experiments 105
4.4 Transformation-Invariant Pooling 109
4.4.1 Transformation-invariance in deep learning . . 109
4.4.2 Related work 112
4.4.3 Method description 115
4.4.4 Experiments 124
4.5 Contributions 129

5 Conclusion & discussion **133**
5.1 Findings 133
5.2 Future work 135
5.3 Concluding remarks 137

Chapter 1

Introduction

In this thesis we address the limitations of current computer vision pipelines by developing *expert-aware algorithms*. The proposed algorithms benefit a range of computer vision problems, especially in highly-specialized fields, such as medical imaging. The current chapter introduces open challenges in modern computer vision and outlines how these challenges can be solved by leveraging expert knowledge.

1.1 Background and motivation

1.1.1 Modern supervised computer vision

Computer vision is currently experiencing one of the fastest developments in its history. The community develops novel methods and models, as well as enhances those already known for a very long time (in large part Convolutional Neural Networks [LB95]). These algorithms are now successfully applied to solve problems that seemed almost impossible to solve just years ago.

Challenges such as "ImageNet Large Scale Visual Recognition Challenge (ILSVRC)" [RDS+15] demonstrated the power of modern computer vision methods when it comes to detection, classification and localization of objects in natural images. Other famous exam-

ples of supervised problems that experienced great progress recently include classification of the morphologies of distant galaxies [gal], real-time human pose recognition [SSK+13a] and segmentation of neuronal structures [ACTB+15].

Three of the most important drivers for these breakthroughs in modern supervised computer vision are the following.

- **Abundance of training data.** Variations in the appearance of real world objects are usually quite large. They arise from both intrinsic variation of objects (different breeds of dogs can look very different, but they still belong to the same object class) and imaging variations (scale, lighting, pose, occlusion and viewpoint). All these variations need to be presented to an algorithm in the training process in order to be captured and accounted for. That is the main reason why the large size of the data sets used for training turned out to be crucial for the development of automated techniques. Within the last five years, natural imaging data sets grew up orders of magnitude in terms of number of training images (5717 images in PASCAL VOC 2012 challenge [EVGW+10] versus 456567 images in ILSVRC 2014 [RDS+15]).

- **Effective training methods.** Both computer vision and optimization techniques evolved significantly during the last decade. Large data sets enabled us to train complex and flexible models with only small risk of overfitting. New data representation methods, new imaging techniques, as well as advanced machine learning models are being constantly developed and improved. Different regularization and optimization methods increase stability and convergence rates of algorithms, speeding up the training process and allowing parallelizable implementations. It is impossible to give a comprehensive overview of all the developments, but we discuss some of the approaches in the later chapters of this thesis in great details.

- **High performance computations on desktop comput-**

ers. Steady growth of available computational power allows researchers in computer vision to process larger and larger volumes of data, and to train more and more complex models. One of the most important milestones being the spread of GPU (Graphics processing unit) computations, as opposed to CPU (Central Processing Unit) computations. Performing vector operations on GPU allows one to employ highly efficient parallel computations without the need for a supercomputer implementations. This development ultimately enables training of models with millions of parameters on data sets consisting of hundreds of thousands of images.

Arguably, a lot of supervised problems in computer vision can be solved when training flexible enough model for long enough time on a large enough representative data set. This *black box approach* proved its efficiency, but only for a very limited range of problems.

1.1.2 Open challenges

Unlike all the examples described above, many problems in supervised and weakly supervised computer vision still can not by design be solved with this "black box" approach. Here we identify three types of problems, that we will also focus on in this thesis, where complementary or completely different approaches are required.

- **Limited number of samples.** Many domains share the property of the data sets being extremely hard to collect. Limiting the number of training samples prevents one from blindly training complex models as they will result in overfitting. The medical domain is one of the most important examples: collecting data can require very expensive and long procedures, such as invasive tissue sampling, or growing an experimental subject animal.

- **Unrepresentative data sets.** In some cases the data just does not contain enough information to solve the problem. If

image resolution is small, it could be impossible to distinguish small objects even for a trained human. Similarly, recognizing the object can be hard if only a part of the object is present in the image. Without some additional structural information this problem cannot be solved.

- **Limited number of labels.** Supervised learning requires a data set to have accurate labels. While in some cases the labelling process might be simple and inexpensive, in many fields image annotation requires highly-specialized knowledge and has to be performed by experts, whose time is usually very limited and therefore, valuable. For example, most natural image data sets are labelled with the use of crowdsourcing, but annotating most of the medical data sets cannot be outsourced, and has to be performed by a specialist in the field.

1.2 Towards expert-aware algorithms

1.2.1 Types of expert prior knowledge

Each of the issues described above, data scarcity, atypical samples or lack of labels, poses a challenge to the community to develop new approaches, that would incorporate all the available information in an intelligent way, rather than just feeding it to powerful general-purpose algorithms. The nature of this knowledge can be different, and can enforce different types of constraints on the algorithms being developed. The people who possess this specialized knowledge are most likely trained domain professionals.

In this thesis we focus on how to obtain maximum relevant information, and how to use it to improve computer vision pipelines across various problems, mostly focusing on medical imaging applications, video data and natural image recognition.

In most cases human experts cannot formalize and describe in detail the precise reasoning or algorithmic procedure which they follow when solving a problem, otherwise automating the solution would be

a simple matter of implementation. However, experts do provide prior knowledge in different ways, the most common being a combination of the following three.

- **Input data properties.** Experts can usually formulate some properties of the data that are specific to the objects captured or to the imaging process. If some of these properties do not correspond to the usual assumptions on the data sets such as those discussed in section 1.1.1, then one can incorporate this knowledge and maybe benefit from it.

- **Recognition process.** Sometimes just observing how an expert solves the problem at hand can provide a lot of useful insights into the structure of the data and the labels. In most cases pattern recognition pipelines operate locally, "looking" at only a patch of the image at a time. This approach is by itself inspired by models of the human visual system. But the most interesting insights come from how experts resolve ambiguities when local appearance is insufficient. They start to employ complex dependencies in the data, that usually come from deep understanding of the underlying data generation processes.

- **Solution properties.** Finally, experts familiar with the field can usually formulate either expected or desired properties of solutions. Even some very broad knowledge on global statistics of the solutions can be very beneficial, especially when the number of labels is small or the labels are unavailable altogether.

In this thesis we focus on all three types of prior knowledge and show multiple examples of how to formulate *expert-aware computer vision* pipelines employing these priors.

1.2.2 Objectives and contributions

The underlying *hypothesis* of the thesis is that incorporating domain-specific knowledge into modern computer vision algorithmic pipelines

can result in significant accuracy gains and performance improvements when compared to conventional models.

To support the claims of this hypothesis, we pursue the following three main objectives in different chapters of the thesis.

1. Creating expert-mimicking methods to resolve ambiguities in the data. These human-inspired techniques incorporate the knowledge about the global structure of the data into the pipeline operating on local patches. This step from local to global appearance can simultaneously result in better accuracy of the final solution to the supervised problem (such as segmentation), as well as in better data representations (in case of image reconstruction or enhancement).

2. Developing weakly supervised algorithms using little to no training data, but some prior knowledge from experts. This strategy includes information about the common solution stages and the desired properties and statistics of the final result. In many cases these statistics can be considered proven in the field, and therefore non-subjective. By incorporating proven biologically motivated priors, we permit a non-subjective pipeline that can be applied to large-scale problems.

3. Improving the feature learning process by incorporating all the knowledge on nuisance variations in the input data. We show how to improve stability, convergence rates and accuracy of existing algorithms by enforcing invariance to specific types of expert-formulated transformations of the input data.

The pursuit of these objectives results in the development of multiple novel algorithms, which are grouped into three branches mentioned above. These algorithms are evaluated and their properties are extensively studied on different problems and data sets.

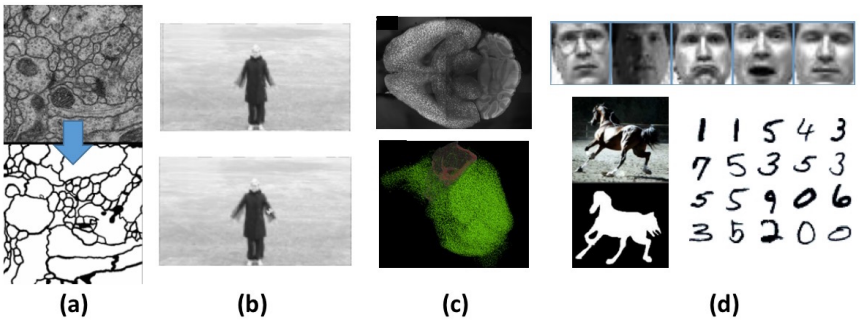

Figure 1.1: Major applications of the methods developed in the thesis. (a) Neuronal structures segmentation (one electron microscopy slice and its binary segmentation). (b) Low frame-rate video enhancement (two consecutive blurred frames before enhancement). (c) Weakly supervised amyloid plaques detection (one microscopy slice and a cloud of detected plaques in green in 3d). (d) Various standard narrow-domain problems (face recognition, horse silhouette segmentation, hand-written digit recognition).

1.3 Areas of application

We demonstrate the properties of the developed methods on multiple problems and benchmarks (see figure 1.1).

- **Connectomics.** The field of connectomics, described in detail in chapter 2, serves as a major application throughout the thesis. Advancing this branch of neuroinformatics is expected to resolve a great number of questions about the functions and mechanisms of different brain structures. The problem of reconstructing neuronal geometry is substantially different from standard natural image recognition. The microscopy data has many unique properties, such as anisotropy and transformation-invariance, and employing these properties is crucial to the development of automated reconstruction techniques.

- **Video data.** Videos can be interpreted as a sequence of dependent images. When working with some specific types of data sets, such as full-exposure low frame-rate videos, employing these dependencies can largely assist in the problems of video enhancement.

- **Amyloid plaques segmentation.** Detecting and segmenting plaques in mice brains and building statistics of their distribution is considerde to be a key step to evaluate new types of drugs against Alzheimer's disease and assist researchers in further drug development. For this problem we build a completely expert-driven, yet non-subjective processing pipeline for neuroimaging. It uses biologically motivated priors on multiple stages to minimize human interactions and it does not requires any labeled data.

- **Narrow-domain object recognition.** We also consider multiple small computer vision classification and segmentation tasks that cover a narrow domain of objects: face classification, horse silhouette segmentation and hand-written digit recognition. These general problems also benefit from some of the expert-formulated priors, such as transformation-invariance.

While these are the main examples that we use throughout the thesis, the range of applications that could benefit from the presented ideas is significantly broader and includes basically any pattern recognition problem that can be manually solved by a trained expert.

1.4 Structure of the thesis

The thesis is divided into five chapters, the first of which is this introduction.

In chapter 2 we discuss how computer vision and image processing algorithms can benefit from expert priors in case of "anisotropy" – a common property of structured sequential data, e.g. in electron

microscopy and video. We introduce two novel methods in this chapter: anisotropic ssTEM image segmentation algorithm using dense correspondence across different sections, and SuperSlicing frame restoration algorithm for anisotropic ssTEM and video data.

Chapter 3 presents a novel technique for amyloid plaque detection from volumetric images at a whole-brain scale with up to a single-cell resolution. This method defines a weakly supervised algorithm that employs biologically motivated priors on the statistics of the solution. These priors are used to tune internal parameters of an algorithm, resulting in a non-subjective and tuning-free pipeline.

Chapter 4 starts with the introduction of *Convolutional Decision Trees* – novel fast and greedy segmentation and classification method. On top of it we develop *Transformation-Invariant Convolutional Jungles* – an algorithm that incorporates the information on nuisance variations of the data (such as rotation- or scale-invariance). We then generalize the approach to more powerful deep learning models, resulting in TI-pooling – a *transformation-invariant pooling operator* for feature learning in Convolutional Neural Networks.

Finally, chapter 5 summarizes the contributions, introduces directions for future developments, and concludes the thesis.

Chapter 2

Anisotropic data

Medical experts often use additional information to resolve ambiguities in the data. For example, by exploring the similarities between structures in different images, they gain the information required to solve the recognition problem at hand. In this chapter we show how to leverage cross-image dependencies in case of anisotropic data by solving the correspondence problem. Employing these dependencies results in the development of two novel algorithms for anisotropic data segmentation and restoration.

2.1 Introduction

Many problems and examples in textbooks on statistics and machine learning start with one common assumption: "data being distributed identically and independently". While data points are collected independently in many tasks and data sets, there exists a whole class of real-world problems, where this assumption is often not satisfied. For example, neuronal activity in different brain regions is not independent: one neuron activation influences the activation of another. Such data is called "structured" as it implies some structural dependencies between data points. Two of the most common examples of structured data are time series and images: the value of a time-series

signal at one point in time usually depends on its value in the previous points in time, as well as color and intensity of one pixel in the image depends on the appearance of the neighboring pixels.

Exploiting these dependencies resulted in development of new machine learning and computer vision methods: Structured Support Vector Machines [TJHA05] for general Structured Prediction [BHS+07], virtual Markov Models for time series [BP66], min-cut/max-flow algorithms [BK04] for Markov Random Fields [KS+80], Convolutional Neural Networks [LB95] for feature extraction in computer vision and time series problems.

In computer vision, these methods allow us to exploit dependencies between the appearance of neighboring pixels: through pairwise potentials in min-cut/max-flow segmentation or through the convolution or pooling operators in Convolutional Neural Networks. Incorporating this prior information on a local level leads to significant improvements of the results on a global level. Some modern approaches go beyond cross-pixel dependencies, and include also cross-image dependencies. For example, one can assume with confidence that labels of similarly-looking superpixels [Mor05] from different images are connected and therefore dependent [VFB11].

2.1.1 Anisotropic data definition

In this chapter we show how to employ cross-image dependencies in case of *anisotropic data sets*. Anisotropic data set is a collection of sequential images (a stack) representing a continuous evolution of structures, in which the resolution across one dimension of the stack is much lower than the resolution of the other two dimensions. Throughout the chapter we consider two major examples of anisotropic data sets: serial section transmission electron microscopy and full-exposure low frame rate videos.

Serial section transmission electron microscopy (ssTEM) [CSP+10] is the only available imaging technique that guarantees sufficient resolution for reconstructing neuronal structures on the synapse level: below 5 square nanometers per pixel. Thereby, ssTEM imaging sup-

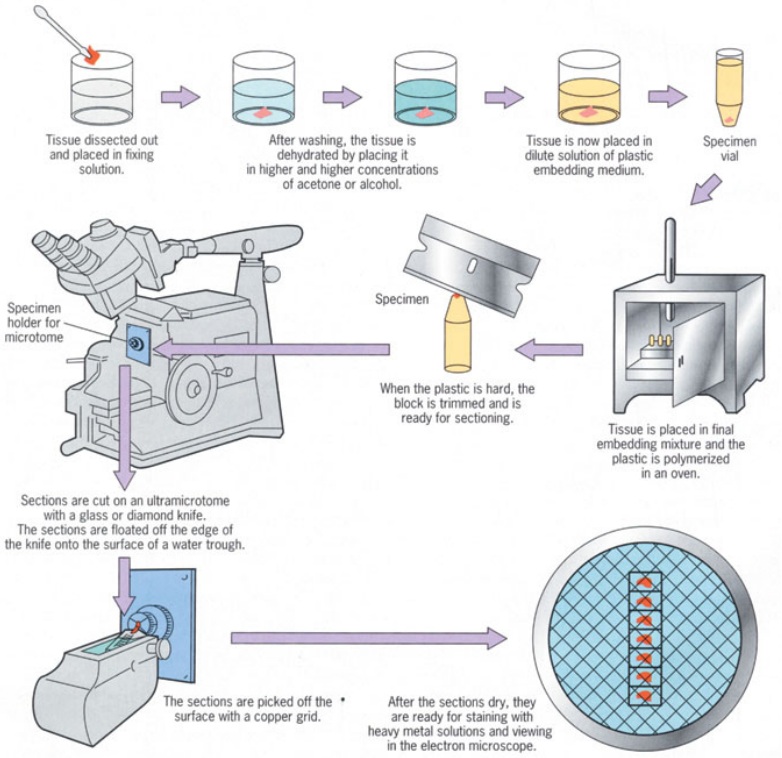

Figure 2.1: Speciment preparation process for ssTEM imaging. Image credit and detailed process description [KF11].

ports the scientific goals of connectomics [Seu12] to understand brain functions. Unfortunately, this technique requires physically cutting brain tissue into thin sections and then imaging individual sections with transmission electron microscopy (figure 2.1). The resolution across the vertical dimension of a stack (limited by the precision of serial cutting) is significantly lower than across the dimensions of a section (limited by the precision of the electron microscopy imaging).

The same phenomenon can be found in a low frame rate full-exposure video recording. Video is called *full-exposure* if its exposure

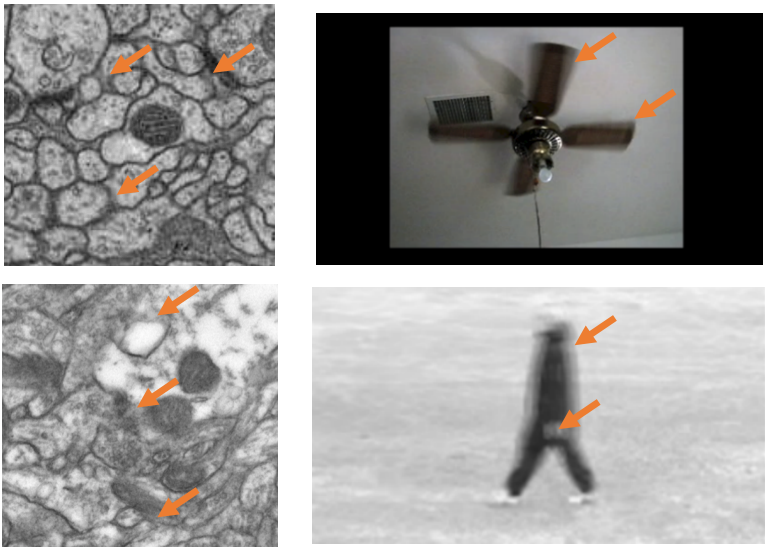

Figure 2.2: Examples of anisotropic data: two ssTEM sections of different tissues (left) and two low frame-rate video frames captured with full exposure (right). Arrows point out the details that are averaged away because of anisotopicity of the data.

time equals to the time between two frames. In case of *anisotropic video*, the resolution of each frame (spatial resolution) is higher than the temporal resolution of the video.

In both examples, the anisotropy of the imaging process causes the individual sections or frames to be blurred. In case of ssTEM, the intensity of a pixel is the cumulative energy that represents the average tissue density along the vertical dimension of the section. Because of this projection, the details of a scene that are smaller than the thickness of the section are *averaged away* (figure 2.2). Similarly, changes in the video scene that occur faster than the frame exposure time fall below temporal resolution of the camera and appear to be blurred.

An expert-aware approach to process anisotropic data is to consider not a single image at a time, but to join the information from multiple consecutive images. Exploiting the similarities between structures in different sections/frames, a human expert gains the information required to solve the recognition problem that is too ambiguous to be solved from a single image. In the following sections we show how to formalize this approach and how to incorporate it into the segmentation (section 2.2) and reconstruction (section 2.3) pipelines.

2.1.2 Anisotropic data and connectomics

ssTEM data is a major example of anisotropic data that we consider in this thesis. This imaging techniques support multiple biological goals that lie in the field of neuronal geometry extraction. In this section we show why these goals are important and what motivates the development of expert-aware algorithms in this field.

Human brains approximately consist of 70 billions neurons, and each neuron is connected with thousands of different neurons – that all form the so-called *Connectome* – the most difficult and unstudied structure in our body, a comprehensive map of connections within an organism's nervous system and its brain. *Connectomics* [Seu12] is the production and study of connectomes – a field where neuroanatomists face the challenging task of reconstructing neuronal structure with synaptic resolution in order to gain insights into the functional connectivity of brain.

Reconstructing the whole brain structure of an animal with all the connections is a very long-term challenge with many intermediate steps. One day neuroscientists will be able to decode the information in the connectome and understand the questions such as: how memory works, how we learn, how to cure mental illnesses, and much-much more. But already today there are challenging biological questions that can be answered if neuronal structures are to be reconstructed even within small region of the brain.

- Neuroscientists know tens of different neuron types within the

retina (light-sensitive layer of an eye). For some of them the functional role is known and for others the role is still to be discovered. One of the challenges is to see what types of neurons are present in different brain parts and to understand, which of them are likely to have specific functional roles.

- Combining the patterns of neural spikes with a realistic connectivity within the region of interest would constitute a large step to understanding how the information is passed through and processed in neuronal tissue. Even more, exploring one specific region of the brain and looking at structural differences for different individuals, one could track how information is encoded in the brains for some specific skill ([CHH14] is an example of active research on bird singing patterns encoded in the neuronal connectivity).

- For some mental disorders like schizophrenia and autism, it has been difficult to identify a clear neuropathology in the brain on larger scales. Automatic detection of miss-wirings on the resolution of a single neuron may help us point out the cause of these mental disorders.

Performing connectome reconstruction requires the imaging technique with the resolution good enough to capture the structures smaller than 10 nanometers. Electron microscopy (EM) has revealed novel facts about synapses and other subcellular structures in the mammalian nervous system [BH12]. Serial EM has been most famously used to reconstruct the connectivity of the *Caenorhabditis elegans* nervous system [WSTB86, JWB+12] – a small creature with only about 300 neurons. More recent improvements in this technique have led to imaging of much larger volumes of brain tissue, and exciting insights into invertebrate nervous systems [BRRS13, TBL+13, KHB+15], and mammalian neural circuits [BHD11, KGZ+14].

In a recent study, about one thousand neurons were reconstructed from a mouse retina using 20 thousand hours of human labor [HBT+13]. Despite of this great effort, the reconstructed retinal volume was just

0.1 mm on each side, only large enough to encompass the smallest types of retinal neurons. This study employed semiautomated methods, using advances in machine learning to automate most of the reconstruction [JST10]. Without the automation, the reconstruction would have required 10–100 times more human effort.

All these recent studies demonstrate that performing this geometry extraction manually is a tedious and error prone process that requires an impractical amount of time. They point to an important need for the development of new computational technology to aid the analysis of EM imagery of brain tissue. Therefore, accurate algorithms for automatic neuronal segmentation are indispensable for large scale geometric reconstruction of densely interconnected neuronal tissue.

To support the goals of connectomics, we investigate two important problems of ssTEM data analysis and propose novel computer vision algorithms to solve them.

- The segmentation problem that aims to annotate neuronal structures in tissue as either membranes or the inside volume of neurons. This segmentation enables tracking the boundaries of individual neurons – the task crucial for both geometry extraction and connectivity estimation.

- The reconstruction problem that aims to increase the depth resolution and de-blur the individual sections, compensating for the artifacts of anisotropic imaging techniques. This processing step is essential to simplify any further analysis and to get insights into the structures even below the imaging resolution.

For both problems we explore the expert approach to handle the dependencies in anisotropic data. In order to resolve the ambiguities of one blurred section, a human expert estimates the correspondences between the structures across multiple images, and joins the information from neighboring sections to better represent the appearance of this structure.

In the following sections we show a way of formalizing this correspondence estimation using registration techniques [ZF03], and how to incorporate the joint information through either relying on machine learning algorithm to achieve that (in case of segmentation), or through the joint optimization problem (in case of reconstruction).

2.1.3 Related work

Segmentation

There are three general approaches for anisotropic data segmentation. The first approach focuses on the detection of neuron membranes in each section independently [KFB10a]. For example, the software package Fiji [Fij] implements this approach: first, in every pixel the vector of features is evaluated, and then this vectors are used to train Random Forest classifier. We use this package for feature extraction, described in detail in section 2.2.1.

The second approach incorporates context from different sections without correspondence alignment. In [KFB10b] the authors propose two terms for graph cut segmentation, one of them incorporates context from neighboring sections. In contrast to our algorithm, this term depends only on the feature vector evaluated in the pixel in a direct z-neighborhood, with no correspondence alignment. Since the difference between the sections is usually large, incorporating of this term does not produce significant improvements.

The third approach [VRHG+11] generates many, possibly contradictory, segmentation hypotheses in individual sections and combines them in order to optimize the global agreement functional that is defined on the whole stack. In contrast to this approach, we are not dealing with given segmentation hypotheses, but incorporate the context from neighboring sections to improve the segmentation of every single section.

Our two novel contributions in the area of anisotropic data segmentation are both based on the same idea motivated by how experts resolve ambiguities in the data. This idea is to exploit context from

neighboring sections by solving the correspondence problem.

First approach that we present in section 2.2 concatenates the information about local appearance of the structure from multiple slices. This resolves ambiguities if some detail is blurred in one slice, but sharp on another.

Second approach described in section 2.3 introduces reconstruction as a preprocessing step for the segmentation. This reconstruction recovers enhanced sub-frames and solves the problem of persistent blurry membranes. As the sub-frames contain finer details, the segmentation algorithm is able to identify the neuronal structures with higher accuracy than methods without preprocessing.

Reconstruction

The first group of related techniques for frame enhancement interpolates between two neighboring frames. The simplest approach is a linear frame interpolation, which, although simple and fast, produces blurry results even when the initial frames are sharp. A more advanced technique [BSL+11] is based on optical flow estimation and frame warping.

However this method would not be able to reconstruct sharp details from initially blurred images, as it often happens in anisotropic data. In contrast, the approach proposed in section 2.3 reconstructs the changes *within* the frame, therefore recovering crisp details in each sub-frame. It becomes possible by taking into account the information on how the imaging is performed and by incorporating the continuity priors on the imaged structures evolution.

Another approach [HNIV+12] to solving the problem of spatial enhancement relies on imaging data from multiple angles, resulting in multiple electron microscopy stacks. Combining multiple representations of the same tissue volume can produce good results, but requires a stack to be imaged multiple times, which is impossible for some specimen in case of ssTEM, because the tissue is usually physically destroyed after the first imaging. Unlike this method, we are considering a more general case and use only one sequence of frames

from one ssTEM stack.

The third type of approaches [SOSS10] is based on exploring the recurrence of small self-similar patches in space and time, which proved to be very beneficial in case of natural images. However, these methods assume that similar patches appear repeatedly within the frame sequence which is almost never the case for neuronal structures. In contrast to these methods we do not rely on high recurrence of self-similar patches and therefore, solve a more general problem.

The proposed approach instead tries to reconstruct the virtual slices from which the imaged slice is composed. This is possible to some extend by incorporating the information on the imaging process itself, and leveraging the information on continuity of the structures across multiple slices.

2.2 Segmentation using dense correspondence

In this section we present an approach for the automatic membrane segmentation in anisotropic stacks. The key challenge is to segment the structures that appear blurred, because these structures can be easily confused with others from the local appearance. This section describes how to use the context from the neighboring sections to resolve these appearance ambiguities.

The problem of membrane segmentation is briefly presented in figures 2.3. Given histological slices of neuronal tissue, the task is to annotate the pixels as either depicting the inner area of the neuron, or the membranes – the tissue surrounding the neuron and isolating one neuron from another. The membranes are usually of higher density, and therefore appear darker than most other structures, but this appearance of neuronal structures is not always the case. The hardest regions for automated methods to segment are mitochondria, synaptic vesicles, and just blurred membranes.

As one can see, local appearance around the pixel in a section may be insufficient to discriminate between the membrane or the inner area of a neuron. This ambiguity is caused by the projection

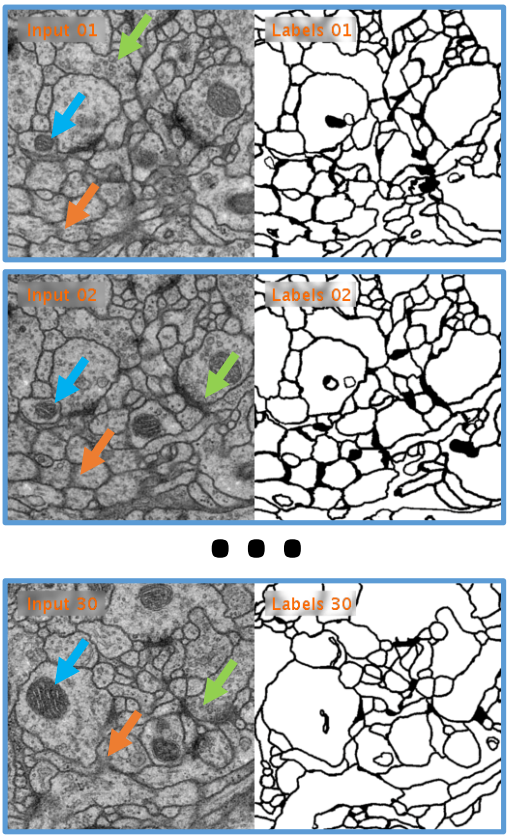

Figure 2.3: The task of membrane segmentation (ISBI 2012 challenge data set [ACTB+15, cha]). The data set consists of a sequence of anisotropic sections captured with electron microscopy together with the corresponding binary labels. Each image is a projection of the whole thickness of the physically cut section. Arrows point out some of the regions that are more difficult to segment than others: mitochondria boundaries marked with blue, synaptic vesicles – with green, and blurred membranes with orange.

mechanism of transmission electron microscopy which projects whole three-dimensional section onto the image plane. The projection causes some of the membranes to appear very blurred as shown in figure 2.3 when they are not orthogonal to a cutting plane. Using larger context from the surrounding area in the same section can resolve some ambiguities [KFB10b, KRP10], but such a reasoning is often not sufficient.

When experts try to label ambiguous pixels, they inspect the neighboring sections to see whether a global correspondence between structures in the sections can be established. They then use the appearance of corresponding structures from neighboring sections to resolve these ambiguities. This task appears to be straightforward for a human even though the images are quite different from one another due to anisotropy.

To enable automatic methods to exploit information from neighboring sections we have to resolve the correspondence problem – finding a mapping from a neighboring section to the current one. We propose to solve this problem by finding global dense correspondence with SIFT flow algorithm [LYT11] and to use the features from different sections to perform segmentation.

2.2.1 Method description

Let $\tau = \left\{I^k, Y^k\right\}_{k=1}^{K}$ be a training set, consisting of K images with a given labeling. Here $I^k = \{x_p^k\}_{p=1}^{N}$ represents an input image of section k, x_p^k corresponds to a pixel in section k. $Y^k = \{y_p^k\}_{p=1}^{N}$ represents the labels of the corresponding pixels p for a section k. The label $y_p^k = 1$ denotes the class *membrane* and $y_p^k = 0$ background or other non-membrane structures. Let $\varphi(x_p^k)$ be a feature vector for the pixel x_p^k. We pursue the goal to build a segmentation algorithm that would automatically label new sets of images.

The proposed method constructs a dense correspondence between the neighboring sections and it uses features that are evaluated in all the corresponding pixels for classification. Our workflow is illustrated in figure 2.4. For a given section I^k we first find warping from the

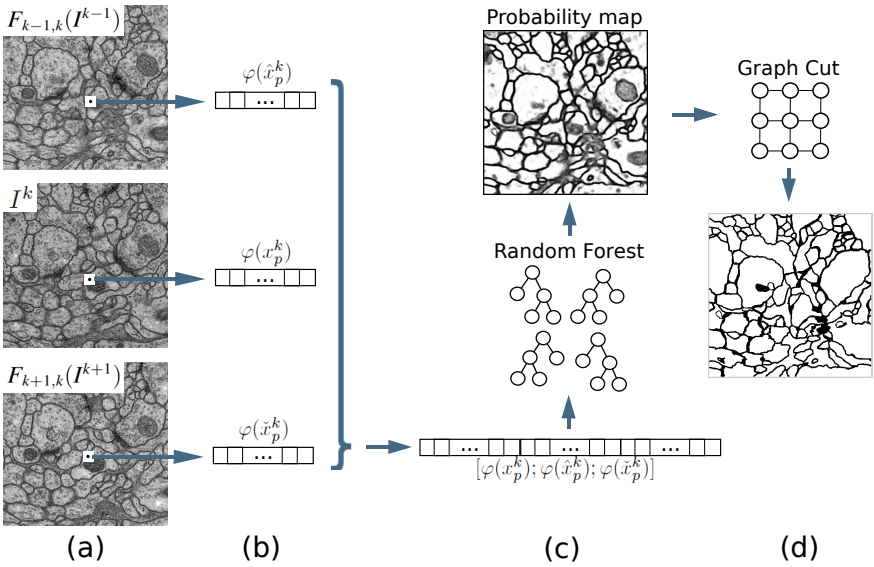

Figure 2.4: Based on the non-linear correspondings $F_{k-1,k}$ and $F_{k+1,k}$, the algorithm evaluates the warped images $F_{k-1,k}(I^{k-1})$ and $F_{k+1,k}(I^{k+1})$ (a). Then, feature vectors in the corresponding pixels are evaluated: $\varphi(\hat{x}_p^k)$, $\varphi(x_p^k)$, $\varphi(\check{x}_p^k)$ (b). After that the method concatenates them and passes the concatenated feature vector to a Random Forest classifier (c). The classifier estimates a probability map that is further segmented by Graph Cut algorithm (d).

neighboring sections I^{k+1} and I^{k-1}: $F_{k+1,k}$ and $F_{k-1,k}$. Then, for every pixel x_p^k we find the corresponding pixels $\hat{x}_p^k$ and $\check{x}_p^k$. Next, we calculate features in all three pixels $\varphi(\hat{x}_p^k)$, $\varphi(x_p^k)$, $\varphi(\check{x}_p^k)$, concatenate the feature vectors and use this *extended feature vector* as input to a Random Forest (RF) classifier. Finally, the RF returnes probabilities of features that enters the Graph Cut segmentation.

Framework

Suppose a non-linear warping $F_{k-1,k}$ is given and it establishes the correspondence between the pixels in the image I^{k-1} and I^k. How such a correspondence can be obtained will be described in section 2.2.1. We introduce two more images to the data set: $I^0 \equiv I^1$ and $I^{K+1} \equiv I^K$ both for training and test sets, so that now there are two neighbors for every section from 1 to K. Every pixel x_p^k is then assigned to the corresponding pixels in the neighboring sections:

$$\hat{x}_p^k = F_{k-1,k}(x_p^k), \quad \check{x}_p^k = F_{k+1,k}(x_p^k). \tag{2.1}$$

To incorporate the context from neighboring sections, an extended feature vector has to capture the contextual feature information associated with the pixel x_p^k itself, as well as with the pixels $\hat{x}_p^k$ and $\check{x}_p^k$. The extended feature vectors form a training set for a Random Forest classifier [Bre01].

$$\tau = \left\{ [\varphi(x_p^k); \varphi(\hat{x}_p^k); \varphi(\check{x}_p^k)], y_p^k, \ 1 \le p \le N, 1 \le k \le K \right\}. \tag{2.2}$$

A trained Random Forest returns the probability of every pixel of the image to belong to a membrane, i.e., a probability map. Afterwards, graph cut segmentation [BK04] with the probability map as unary potentials partitions the image into semantically meaningful segments.

Dense correspondence.

To find a dense correspondence between the sections we use the recently proposed method "SIFT flow" [LYT11]. SIFT flow finds the non-linear warping $F_{1,2}$ on the pixel grid x_p^1 between the images I^1

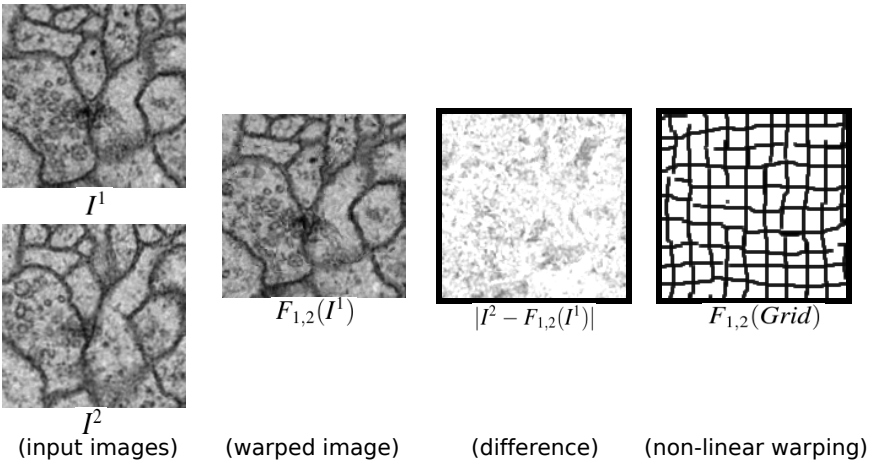

Figure 2.5: An example of non-linear warping between images I^1 and I^2 found by SIFT flow algorithm. Image $F_{1,2}(I^1)$ shows the warping applied to image I^1 and image $F_{1,2}(Grid)$ visualizes the warping by applying it to a grid image.

and I^2 by minimizing the following energy:

$$E(F_{1,2}) = \sum_{p=1}^{N} \min\Big\{\|s(x_p^2) - s\big(F_{1,2}(x_p^1)\big)\|, t\Big\} + \sum_{p=1}^{N} \gamma D\big(x_p^1, F_{1,2}(x_p^1)\big) + \sum_{(p,q)\in\epsilon} \min\Big\{\alpha D\big(F_{1,2}(x_p^1), F_{1,2}(x_q^1)\big), d\Big\}. \tag{2.3}$$

$E(F_{1,2})$ is comprised of a data term, a small displacement term and a smoothness term. The first term constrains the SIFT descriptors $s(x_p^2)$ [Low99] evaluated in pixel x_p^2 to be matched along with the descriptors evaluated in pixel $F_{1,2}(x_p^1)$. The small displacement term minimizes the differences between the original image and a wrapped one. D is equal to the distance between the two pixels in a pixel grid. The smoothness term favors a transformation of adjacent pixels to be similar. In this objective function, truncated L_1 norms are used in both the data term and the smoothness term to account for matching outliers and discontinuities, with t and d as the threshold, respectively. As most of the neuronal structures are continuous, the use of L_1 norm is not essential in our case, which we discuss in section 2.3.1. Figure 2.5 shows the results of applying SIFT flow algorithm to two consecutive images from ISBI 2012 segmentation data set. For further information we refer to [LYT11].

Features

The whole set of features provided by toolbox [Fij] is used in this study: Gaussian blur, Sobel filter, Hessian, Difference of gaussians, Membrane projections, Variance, Mean, Minimum, Maximum, Median, Anisotropic diffusion, Bilateral, Lipschitz, Kuwahara, Gabor, Laplacian, Structure, Derivatives.

Additionally, we incorporate all the components of SIFT histogram [Low99] in the pixel and some newly developed features that proved

to be informative for neuronal reconstruction: radon-like features [KRP10], ray features [LSA+10] and line filter transform [SB07].

Overall 626 features are being calculated in each pixel. When we incorporate the features from the neighboring sections into an extended feature vector, this number increases to 1878. RF performs well even in presence of lots of noisy features [Bre01], therefore we need no feature selection procedure.

Graph cut segmentation

The probability of a pixel belonging to a membrane is evaluated by RF independently for every pixel in the image. We use graph cut segmentation to take into account the fact that neighboring pixels are more likely to have the same label. For simplicity we drop the upper index in the following equations, as our graph cut algorithm processes one section at a time: $y_p = y_p^k$.

To determine the labels $Y = \{y_p\}_{p=1}^N$, we combine the approaches described in [KFB10b] and in [VS01]. The segmentation task is formulated as an energy minimization problem $\hat{Y} = \arg\min_Y E(Y)$, where

$$\begin{aligned} E(Y) = &\sum_{p=1}^{N} E_u(y_p) + \lambda_s \sum_{(p,q)\in\epsilon} E_s(y_p, y_q) + \\ &\lambda_{gf} \sum_{p=1}^{N} E_{gf}(y_p) + \lambda_{gc} \sum_{(p,q)\in\epsilon} E_{gc}(y_p, y_q). \end{aligned} \tag{2.4}$$

Here the first term specifies a unary potential that equals to the negative log probabilities given by the RF in every pixel. Let $i(x_p)$ be an intensity of the image in pixel x_p. Then the second term models a smoothness constraint that penalizes discontinuities in the segmentation of neighbored pixels with similar intensities:

$$E_s(y_p, y_q) = \exp\left(-\frac{(i(x_p) - i(x_q))^2}{2\sigma_s^2}\right) \frac{\delta(y_p, y_q)}{D(x_p, x_q)}, \tag{2.5}$$

where $\delta(y_p, y_q)$ is a Kronecker function that equals 0 if $y_p = y_q$ and 1 otherwise.

The gradient flux term [VS01] is defined as follows:

$$E_{gf}(y_p) = \begin{cases} \max\{0, F(x_p)\} & \text{if } y_p = 1 \\ -\min\{0, F(x_p)\} & \text{if } y_p = 0, \end{cases} \tag{2.6}$$

where $F(x_p)$ denotes a gradient flux,

$$F(x_p) = \sum_{x_q:(x_p,x_q)\in\epsilon} < u_{x_p,x_q}, v_{x_p} >,$$

u_{x_p,x_q} represents a unit vector pointing from pixel x_p to the neighboring pixel x_q and vector v_{x_p} corresponds to the gradient vector at pixel x_p.

The good-continuation term [KFB10b] is defined as follows:

$$E_{gc}(y_p, y_q) = | < v_{x_p}, u_{x_p,x_q} > | \exp\left(-\frac{(i(x_p) - i_m)^2}{2\sigma_{gc}^2}\right) \frac{\delta_{\rightarrow}(y_p, y_q)}{D(x_p, x_q)}, \tag{2.7}$$

The variable i_m encodes the average gray value of membrane pixels and σ_{gc} is estimated as the variance of these gray values. The factor $\delta_{\rightarrow}(y_p, y_q) = 1$ for $y_p = 1$, $y_q = 0$ and equals 0 for all other cases.

The minimum of $E(Y)$ is computed by min-cut/max-flow computation [BK04]. The cross-validation procedure determines the unknown parameters λ_s, λ_{gf}, λ_{gc} such that the results generalize in an optimal way.

As a post-processing procedure two steps are performed iteratively: region removing and line filter transform [SB07]. Region removing is performed by a series of thresholding operations based on region properties such as Area, Solidity, Euler Number and Eccentricity. The line filter transform makes segmentation results smoother and fills the gaps between membrane segments.

2.2.2 Experiments

Data

The experiments are performed with the data provided for the ISBI 2012 challenge "Segmentation of neuronal structures in EM stacks" [ACTB+15, cha], example of training images are shown in figure 2.3. The data set [CSP+10] is comprised of a training and a test set. Each set consists of 30 sections from a ssTEM of the Drosophila first instar larva ventral nerve cord (VNC), imaged at a resolution of $4 \times 4 \times 50$ nm/pixel and cover a $2 \times 2 \times 1.5$ micron cube of neural tissue. Training and test sets are selected from different volumes of the same VNC.

Error metrics

Two metrics are used for the task of membrane segmentation: *Pixel error* and *Splits and Mergers Warping error*. Given the estimated labeling $\hat{Y}$ and ground truth $Y^\star$, the pixel error is defined as the Hamming distance between the two labelings $\sum_p \delta(\hat{Y}_p, Y^\star_p)$.

Splits and Mergers Warping error is a segmentation metric that penalizes topological disagreements between the two labelings [JBR+10]. The warping error is the squared Euclidean distance between $Y^\star$ and the "best warping" L of $\hat{Y}$ onto $Y^\star$ such that the warping L is from the class Λ that preserve topological structure: $\min_{L \in \Lambda} \sum_p \delta(L(\hat{Y})_p, Y^\star_p)$.

Both types of errors are evaluated automatically on the test set when the results are submitted to the testing server. The challenge also has determined the errors caused by discrepancy in human labeling.

Results

Our experiments are conducted with the default parameters of the SIFT flow algorithm: $\gamma = 0.05$, $t = 0.1$, $\alpha = 2$, $d = 40$. We compare the results of three different versions of our algorithm: with no context from neighboring sections (*one slice*), with direct correspondence (we incorporate the context from the pixels being a direct z-neighbors,

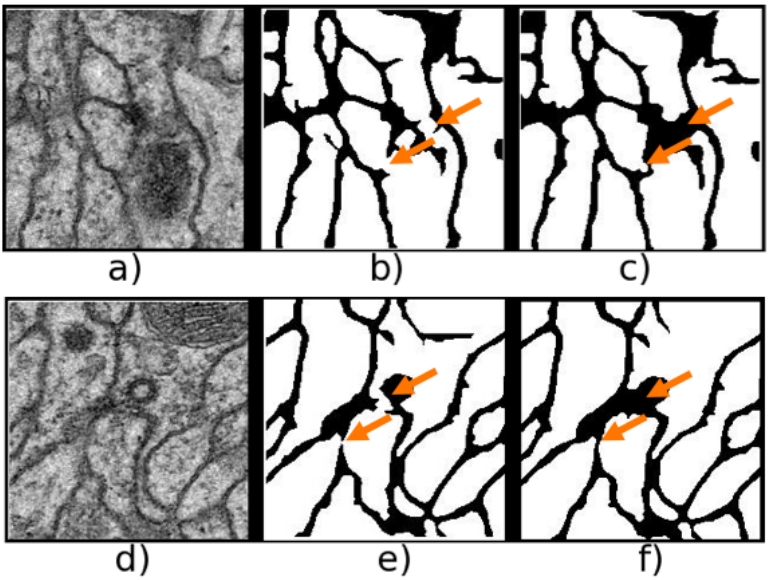

Figure 2.6: Original images: (a, d), results using only one slice: (b, e), results using dense correspondence across three slices: (c, f). Arrows point out some corrections of neuronal morphology.

Method	Pixel error	Warping error
Human	$6.7 * 10^{-2}$	$3.4 * 10^{-4}$
Dense *ETH*	$7.9 * 10^{-2}$	$6.2 * 10^{-4}$
Direct *ETH*	$8.0 * 10^{-2}$	$6.5 * 10^{-4}$
One slice *ETH*	$8.5 * 10^{-2}$	$6.4 * 10^{-4}$
IDSIA	$6.0 * 10^{-2}$	$4.3 * 10^{-4}$
CSIRO	$8.7 * 10^{-2}$	$6.8 * 10^{-4}$
Utah	$1.3 * 10^{-1}$	$1.6 * 10^{-2}$
NIST	$1.5 * 10^{-1}$	$1.6 * 10^{-2}$

Table 2.1: Comparison of error results on a testing set for different versions of the algorithm and the results of other teams. **Human** denotes the error of human annotators.

with no warping procedure), and with dense correspondence found by SIFT flow algorithm.

Results are presented in table 2.1. Some examples of the resulting images are presented in figure 2.6. Incorporating the context from the neighboring sections with direct correspondence leads to improvement in terms of pixel error, but it performs worse in terms of warping error. On the other hand, using dense correspondence leads to improvement in both objectives: 3.6% improvement in warping error and 6.4% for pixel error.

Most of other algorithms applied in the ISBI challenge exploited the context of only one single slice [ACTB+15]. The NIST team [IG13] and the CSIRO team [TS12] employed Support Vector Machine (SVM) as a classifier. A team from Scientific Computing and Imaging Institute, University of Utah [JWG+13] designed series of Classifiers and Watershed Trees. The Swiss AI Lab IDSIA team [CGGS12] trained Deep Neural Networks which appeared to be competitive to ours and their solution was slightly better in quantitative terms. The last approach, however, requires almost a week of training time with specialized hardware, and it is therefore much more difficult to apply in real-world scenarios.

2.3 SuperSlicing frame restoration

The key issue of anisotropic data is that some of the details can be averaged during the imaging process. The averaging blurs the images and introduces ambiguities in the data. The information from different slices can improve the segmentation quality by resolving some of the uncertainties, as we demonstrated in the previous section 2.2. But if some detail is ambiguous in multiple slices, combining features can be not enough.

This section presents a reconstruction algorithm that resolves some of the averaged details, giving insight into what is happening inside a ssTEM slice beyond anisotropic resolution or within a blurred frame beyond temporal resolution. We demonstrate the reconstruction al-

gorithm to recover even the details that appear blurred throughout the data.

Digital imaging defines a quantization of the visual appearance of the world. The intensity of a pixel is the *cumulative* energy that has reached the physical sensor. In consequence, details of a scene that are smaller than the spatial resolution of the sensor are *averaged away* (see figure 2.7). Visually, averaging overcomes the problem of aliasing, but causes spatial blur in anisotropic setting.

An anisotropic frame can be modelled as an average of *virtual isotropic sub-frames* (see figure 2.7). In this part of the thesis we focus on reconstructing *isotropic sub-frames* from anisotropic data to support subsequent image processing tasks like image annotation.

For example, one can model an ssTEM image of a thick section as an average projection of a set of thin sections. Reconstructing these thin sections most often results in improved insights into structure changes below the depth resolution and lead to better geometry extraction. Or in case of low frame rate video, one can interpret the captured frame as an average of *virtual sub-frames* captured with shorter exposure time. The goal is then is to increase temporal resolution: estimate a high frame rate video from low frame rate.

We propose a method called SuperSlicing (Super resolution frame Slicing). It reconstructs isotropic *virtual sub-frames* from a sequence of anisotropic frames, thereby increasing the depth or temporal resolution. This reconstruction states an inherently ill-posed problem as there exists an infinite number of possible sub-frames that can produce the same observed frame. We propose a regularisation that uses the information from the neighboring frames to resolve these ambiguities. The problem is formulated as energy minimization which appears to be convex and therefore guarantees the global optimum. The objective function is guided by two principal considerations: i) the physical constraints of the imaging process; ii) the structures in sub-frames should follow the correspondence between structures in the neighboring frames. To formalize the latter SuperSlicing uses optical flow [Liu09] to find the correspondences between neighboring

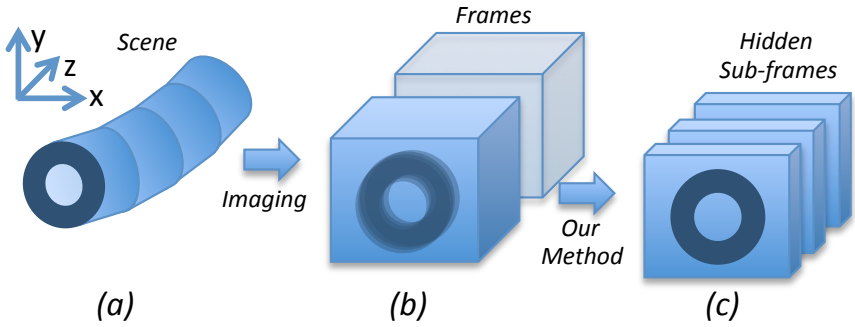

Figure 2.7: A schematic illustration of our approach: (a) neuronal structure in brain tissue sample; (b) the tissue sample is cut and captured with ssTEM, producing anisotropic frames with blur; (c) the proposed method SuperSlicing reconstructs *virtual sub-frames* with sharp details.

frames and interpolates them into sub-frames.

Reconstruction is an important goal by itself, but we also demonstrate how SuperSlicing enables a novel automated method to perform neuronal structure segmentation (section 2.3.2). It recovers the crisp image of these structures and facilitates recognition of neural structures. The experiments on the Drosophila VNC data set (described in section 2.2.2) demonstrate significant improvement over the baselines.

2.3.1 Reconstruction method description

Let Y^n be the observed sequence of frames, $n \in [1, \ldots, N]$, y_p^n – pixel p of the frame Y^n, $i(y_p^n)$ – the intensity of pixel y_p^n. Let $\epsilon(x_p^n)$ be a set of neighbors of pixel x_p^n. We want to reconstruct L *virtual sub-frames* $X^{n,l}$, $l \in [1, \ldots, L]$ of the observed frames Y^n.

Optimization task

We define optimization problem 2.9 to approximate *virtual sub-frames* as an energy minimization problem for given correspondences Ω. The energy 2.9 consists of three terms.

The first term, the data term, represents the relaxed physical constraints that the observed frame should be equal to the average of the *virtual sub-frames*:

$$i(y_p^n) = \frac{1}{L}\sum_{l=1}^{L} i(x_p^{n,l}), \forall y_p^n \in Y^n. \tag{2.8}$$

The second term promotes smoothness by favoring an alignment of pixel's intensities in the sub-frames along the structure's progression between the frames. The algorithm proceeds by finding correspondences between the anisotropic frames using optical flow and then interpolates them into the sub-frames using bilinear interpolation (see section 2.3.1).

The third term encourages the resulting sub-frames to be smooth to avoid visual artifacts. This goal is achieved by minimizing the difference of intensities between the neighboring pixels.

The sum of these three terms forms the final energy function:

$$\begin{aligned} E(X^{n,1},\ldots,X^{n,L}) = \sum_{y\in Y^n}\Big(i(y) - \frac{1}{L}\sum_{l=1}^{L} i(x_p^{n,l})\Big)^2 + \\ \lambda \sum_{(\hat{x}_p^{n,l},\hat{x}_q^{n,l+1})\in\Omega}\Bigg(\sum_{x\in\epsilon(\hat{x}_p^{n,l})} w(x,\hat{x}_p^{n,l})i(x) - \sum_{x\in\epsilon(\hat{x}_q^{n,l+1})} w(x,\hat{x}_q^{n,l+1})i(x)\Bigg)^2 + \\ \gamma \sum_{\substack{x_p^{n,l};x_q^{n,l}\in\epsilon(x_p^{n,l}) \\ l=1,\ldots,L}}\Big(i(x_p^{n,l}) - i(x_q^{n,l})\Big)^2 \end{aligned} \tag{2.9}$$

Here λ and γ are Lagrange parameters that control the degree of regularization versus data fidelity. $E(X^{n,1},\ldots,X^{n,L})$ is a quadratic

functional with respect to $i(x_q^{n,l})$ and therefore we can efficiently calculate the *global optimum* with any convex optimization technique (we used interior point method [NNY94] in our experiments).

Corresponding pixels

How can we find the set Ω of corresponding pixels? A central idea is to utilize the context of neighboring frames for reconstructing sub-frames. We first find the correspondences between the pixels in neighboring frames, same as in section 2.2. We then interpolate these correspondences through sub-frames. The major difference with the dense correspondences described in previous section 2.2 is that instead of SIFT flow [LYT11] algorithm we employ optical flow algorithm [Liu09]. While being faster than SIFT flow, it provides good enough priors for the optimization problem.

Assume that we observe the sequence of three images: Y^1, $Y^2 \equiv Y$, Y^3. For every pixel y_p^2 of Y^2 we find the corresponding pixel y_p^k from image Y^k, $k \in \{1, 3\}$ by finding the set $\Omega_Y^k = \{(y_p^2, y_q^k) | \forall y_p^2 \in Y^2\}$ minimizing optical flow energy:

$$E_{fl}(\Omega_Y^k) = \sum_{y_p \in Y} \Big(i(y_p) - i(y_q^k) \Big)^2 + \alpha \sum_{y_p \in Y^2} \rho(y_p, y_q^k)^2 \qquad (2.10)$$

Here α is a model parameter, $\rho(y_p, y_q)$ is Euclidean distance between the pixels y_p and y_q in pixel grid. Optical flow results in good correspondences, even though it considers only integer displacements, because the membrane displacements are smooth and need to be estimated only up to the thickness of a membrane, which is on average 3 to 7 pixels.

As soon as we have corresponding sets Ω_Y^1 and Ω_Y^3, we can draw a curve φ through y_p^1 to y_q^2 and y_t^3 for every two correspondings (y_p^1, y_q^2) and (y_q^2, y_t^3). Then we interpolate the pixels curve φ crosses in virtual sub-slices: $\hat{x}_{\varphi(1)}^1, \ldots, \hat{x}_{\varphi(L)}^L$ (see figure 2.8). Then

$$\Omega_\varphi = \{(\hat{x}_{\varphi(l)}^l, \hat{x}_{\varphi(l+1)}^{l+1}) | l \in [1, \ldots, L-1]\}.$$

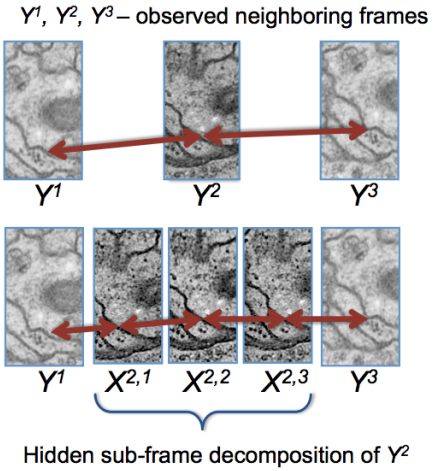

Figure 2.8: An illustration of correspondence interpolation. Top: arrows show correspondences between original frames. Bottom: arrows shows interpolated correspondences between sub-frames. The second term of the energy function encourages the corresponding pixels to have low difference in intensities.

The final set Ω is a union of all sets Ω_φ.

We then write the second set of constraints enforcing that corresponding pixels of sub-frames have the same intensity:

$$i(\hat{x}_p^{n,l}) = i(\hat{x}_q^{n,l+1})), \forall(\hat{x}_p^{n,l}, \hat{x}_q^{n,l+1}) \in \Omega, \tag{2.11}$$

where Ω is a set of all pairs of corresponding pixels.

If pixel $\hat{x}_p^{n,l}$ does not fit to the pixel grid, we employ the bilinear interpolation technique and rewrite it as a weighted sum of direct neighbors in a grid $\hat{x}_p^{n,l} = \sum_{x \in \epsilon(\hat{x}_p^{n,l})} w(x, \hat{x}_p^{n,l})x$, $w(.) \geq 0$, $\sum_{x \in \epsilon(\hat{x}_p^{n,l})} w(x, \hat{x}_p^{n,l}) = 1$. Here $w(x_1, x_2)$ is a bilinear weight that is closer to 1 if the distance between x_1 and x_2 is small and closer to 0 otherwise. We then rewrite the second set of constraints in the following manner:

$$\sum_{x \in \epsilon(\hat{x}_p^{n,l})} w(x, \hat{x}_p^{n,l}) i(x) = \sum_{x \in \epsilon(\hat{x}_q^{n,l+1})} w(x, \hat{x}_q^{n,l+1}) i(x), \tag{2.12}$$

2.3.2 SuperSlicing for neuronal segmentation

In this section we show how segmentation pipeline discussed in section 2.2 can benefit from the use of the proposed reconstruction technique.

We propose a method that first reconstructs virtual sub-frames and uses features that are evaluated in pixels of recovered sub-frames for classification. Our workflow is illustrated in figure 2.9. For a given section Y^n we first recover sub-frames $X^{n,1}, \ldots, X^{n,L}$ with SuperSlicing. Then, similar to the ideas from section 2.2, for every pixel $x_p^{n,l}$, $l \in [1, \ldots, L]$ we calculate features $\varphi(x_p^{n,l})$, concatenate the feature vectors and use this extended feature vector as input to a Random Forest classifier [Bre01]. A huge simplification of the pipeline comes from the fact that ones virtual sub-frames are reconstructed, only direct correspondence needs to be used.

We select the method parameters γ and λ as well as optical flow parameter α with cross validation. We use Random Forest with 255 trees and perform training on 10% of all the pixels. As features we use

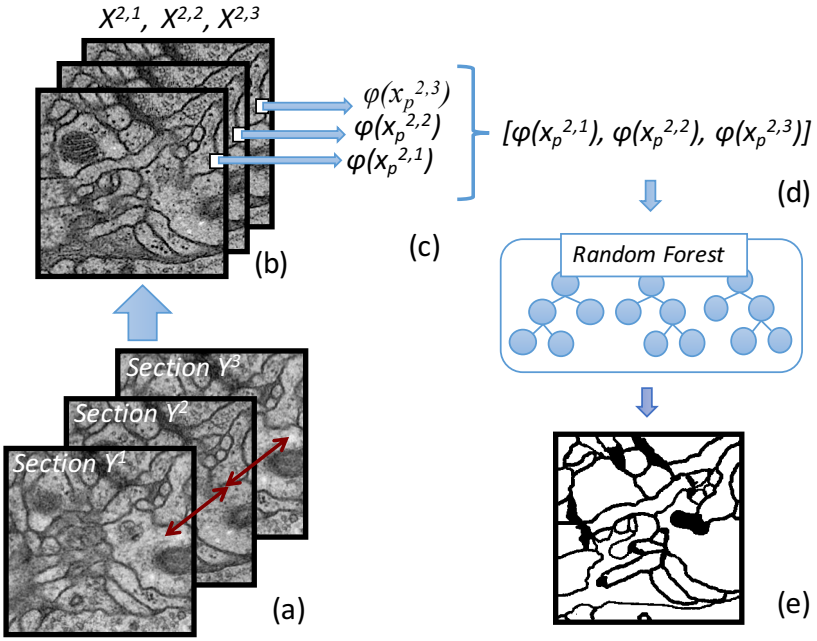

Figure 2.9: An illustration of the SUPERSLICING pipeline for neuronal structures segmentation. Based on the non-linear correspondings between neighboring frames Y^1, Y^2 and Y^3 (a) the algorithm evaluates virtual sub-frames $X^{2,1}$, $X^{2,2}$, $X^{2,3}$ (b). Then, feature vectors in sub-frame pixels are evaluated: $\varphi(x_p^{n,1}), \ldots, \varphi(x_p^{n,L})$ (c). After that the method concatenates them and passes the concatenated feature vector to a RF classifier (d) that returns the final segmentation (e).

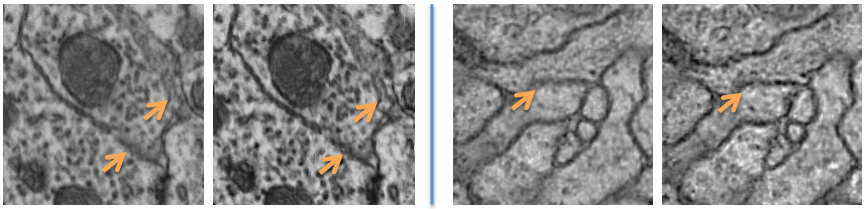

Figure 2.10: Two fragments of neuronal tissue captured with ssTEM: original sections (left) and one of sub-frames (right). Arrows point out membranes that were blurred out in the original images and appear more visible after sub-frame decomposition.

per pixel SIFT histograms [Low99] and line filter transforms [SB07] with different parameters.

2.3.3 Experiments

To evaluate SuperSlicing approach we perform experiments on several different tasks and data sets. For all of the following experiments we select the method parameters γ and λ as well as optical flow parameter α with 5-fold cross validation and with respect to the corresponding metric.

ssTEM imaging and Neuronal Reconstruction

The first set of experiments is performed on an anisotropic electron microscopy stack, imaging neuronal structures. The data set is described in details in the previous section 2.2.2. Figures 2.10 and 2.11 qualitatively shows the results of our algorithm for virtual frame recovery. Membranes recovered in the sub-frames using SuperSlicing are much sharper than the ones produced by the baseline methods.

To quantitatively test the approach for neuronal membrane segmentation presented in section 2.3.2, we compare segmentation results with two more methods: RF segmentation based on only features evaluated in one layer [KFB10a], and RF segmentation based on context

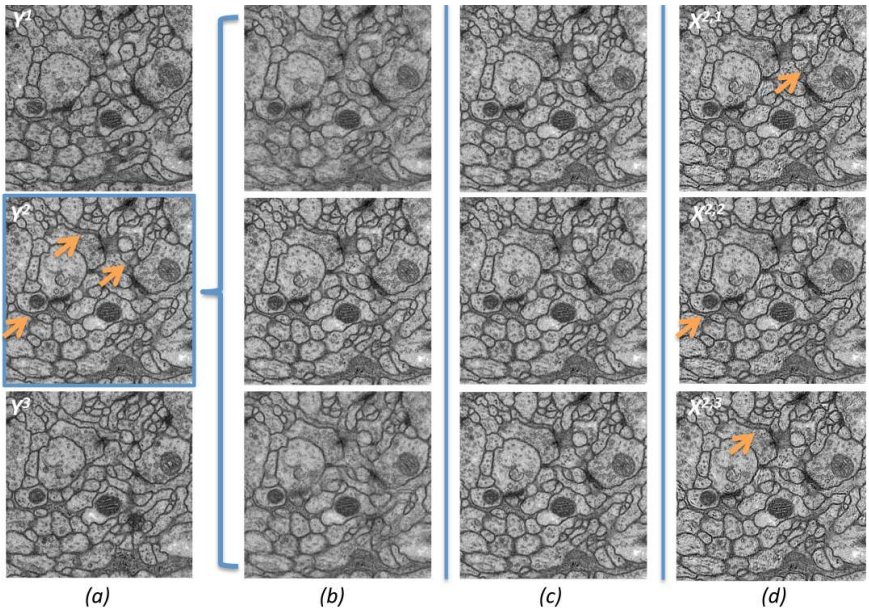

Figure 2.11: A qualitative comparison of our method with the baselines. Column (a) shows original anisotropic sections. Three following column shows $L = 3$ interpolated frames estimated with: linear interpolation (b), optical flow warping (c), SuperSlicing (d). Arrows point out blurred membranes that are better visible after sub-frame reconstruction.

Method	Warping error
One-section segmentation [KFB10a]	$2.88 * 10^{-3}$
Three consecutive sections (chapter 2)	$2.69 * 10^{-3}$
SUPERSLICING segmentation	$2.38 * 10^{-3}$

Table 2.2: Warping error on a testing set for one-section segmentation, segmentation based on three consecutive sections and for SUPERSLICING. The proposed method outperforms the baseline one-section method by 17% and the method proposed in section 2.2 by 11%.

from neighboring sections (see chapter 2 for detailed description). For fair comparison we implement the same set of features for all three methods and use the same RF structure with no post-processing to measure the impact of SUPERSLICING.

In this set of experiments we omit pixel-wise error metric, as it is less relevant to neurons topology [ACTB+15], and compare the results in terms of warping error [JBR+10], which is also described in section 2.2.2. The results are summarized in table 2.2. The results on sub-frame stack produced by SUPERSLICING are 17% better than one sections segmentation and 11% better then the previous results based on three neighboring sections.

Natural videos

Rotating Fan We test the proposed algorithm on a rotating fan video from [SFI11] to evaluate our method qualitatively [1]. As the rotation speed is higher than the shutter speed the frame renders blurred fan blades. Based on three neighboring frames and no prior information we estimate $L = 3$ virtual sub-frames with linear interpolation, optical flow interpolation and the proposed method. Figure 2.12 shows the results of comparison. As can be seen linear interpo-

[1] We do not compare with [SFI11] directly, as their method operates under different assumptions and, moreover, they provide no quantitative results.

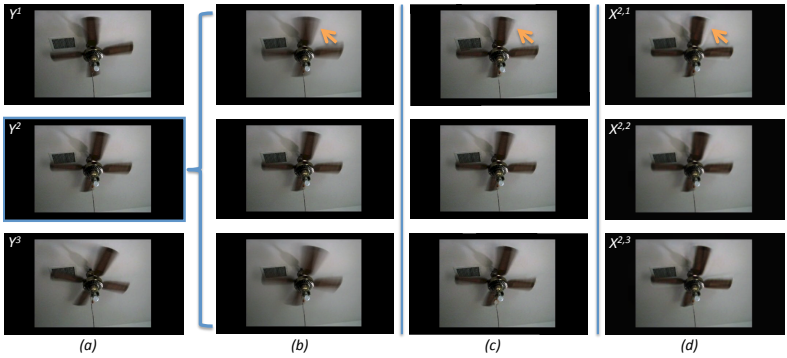

Figure 2.12: A comparison of SuperSlicing with the results of alternative methods. Column (a) shows original frames Y^1, Y^2 and Y^3. Each following column shows three interpolated frames estimated with: linear interpolation (b), optical flow warping (c), SuperSlicing (d). Arrows point out that SuperSlicing results in less blurred fan blades.

lation blurs sub-frames even more. Optical flow interpolation shows the rotation of the fan, but as the initial frames are blurred, the resulting warping is blurred as well. SuperSlicing shows superior results: it reconstructs the original shape of the blades and renders sharp sub-frames.

KTH data set We perform synthetic experiments on the KTH action database [SLC04] to quantify the quality of SuperSlicing reconstruction. This database consists of videos recorded at 24 frames per second. We first downsample the frame rate to 8 frames per second while taking an average of three neighboring frames (low frame rate videos). Then we reconstruct sub-frames with four different methods: frame repetition, linear interpolation, optical flow warping and SuperSlicing. Figure 2.13 shows qualitative results for the number of virtual sub-frames equal $L = 2$ or 3.

Boxplots in figure 2.14 visualises the comparison of peak signal to

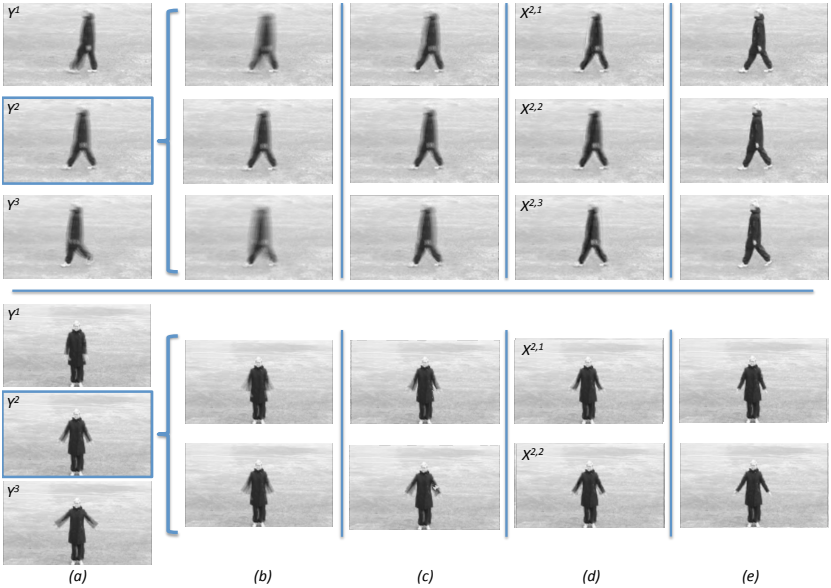

Figure 2.13: A comparison of our reconstrction results with the results of different methods and with ground truth. Top: walking person video reconstruction with $L = 3$ virtual sub-frames. Bottom: hand waving person video reconstruction with $L = 2$ virtual sub-frames. Column (a) shows original frames Y^1, Y^2 and Y^3 from low frame rate video. Three following column shows $L = 3$ interpolated frames estimated with: linear interpolation (b), optical flow warping (c), SuperSlicing (d). Column (e) shows ground truth from high frame rate video. Our results are less blurred and they are qualitatively closer to the ground truth than the results of the baseline methods.

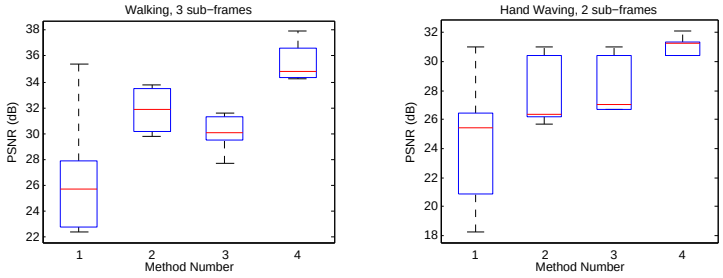

Figure 2.14: An illustration of quantitative results on KTH videos for different methods. Left plot: walking person video with $L = 3$ virtual sub-frames, right plot: hand waving person video with $L = 2$ virtual sub-frames. Each boxplot shows statistics for PSNR (in dB) evaluated for: frame repetition (1), linear interpolation (2), optical flow warping (3) and our method (4).

noise ratio (PSNR) evaluated on 25 frames of video for $L = 3$ and $L = 2$ respectively. SuperSlicing outperforms baseline methods for almost all frames and the average quantitative results appear to be significantly superior: 23% better for frame repetition and 10% for both linear interpolation and optical flow warping.

2.4 Contributions

In this chapter we investigate *anisotropic data* – an important example of specialized sequential data with many non-standard properties, that render standard techniques less applicable. Expert approaches to resolve ambiguities in this data lead to formulation of two ideas on how to employ properties of anisotropic data.

- The sequential nature of the data introduces dependencies between the pixels in neighboring sections/frames. These dependencies represent the evolution of a physical object, and therefore continuous, but usually non-rigid. The complexity of these

dependencies prevent isotopic techniques to compensate for the blurring. However, using dense correspondence between the consecutive images can introduce additional information crucial to the solution of the problem at hand.

- Each section of anisotropic data represent an accumulation of the information across one of the axes (either spatial in case of ssTEM or temporal in case of video). While this projection results in a blurred representation of some of the details, together with the cross-image dependencies, it also defines a physical model that can be used to formulate priors leading to a plausible solution to a reconstruction problem.

Exploiting these data properties results in the following contributions.

- **A novel membrane segmentation method using dense correspondences across sections.** If some neuronal structure is not clearly visible in one section, but visible in another, then bringing the information from multiple corresponding regions together can introduce the required information to the machine learning algorithm. The proposed algorithm combines the information from different sections by robustly solving the correspondence problem with SIFT flow and then using an extended feature vector for training. The resulting algorithm achieves second-best quality in the ISBI 2012 challenge, while being very fast to train. When compared to the baselines, the method is 3.6% more accurate in warping error, and 6.4% in pixel error.

- **SuperSlicing reconstruction and enhancement algorithm.** The effect of blur in anisotropic data sets can be modelled as an average of virtual isotopic frames representing an evolving structure. While in general reconstructing these isotropic frames is an ill-posed problem, we are able to formulate priors on evolution continuity and smoothness, and achieve a solution that is both deblurred. We demonstrate qualitative benefits on

various anisotropic data sets. Quantitatively, our reconstruction results are on average 10% better in terms of PSNR than state of the art.

- **SuperSlicing for membrane segmentation.** Combining the two approaches described above leads to the method that first reconstructs virtual sub-frames with SuperSlicing and then uses them to compose an extended feature vector for further segmentation. Such an approach can successfully identify a neuronal structure, even if it appears blurred in all the images. This processing step results in a quantitative boost of up to 17% in terms of warping error when comparing with the baselines.

Chapter 3

Global biological priors

Field experts often posses knowledge on how to validate whether the solution to the problem at hand is reasonable. For example, in case of object detection problem, experts can often formulate expected objects properties, such as object size or position. If an algorithm detects objects with very different properties than expected – probably the algorithm needs to be improved.

In this chapter we discuss how to leverage prior information on expected properties of the solution to enable algorithm parameter tuning. The proposed approach permits parameter tuning even in cases when supervised training is not feasible due to lack of labelled data.

The major application for the methods and ideas described in this chapter is amyloid plaque detection in mouse brains. In the following section 3.1 we describe the relevance of the problem, explore variations in different data sets and investigate some statistics of interest.

Section 3.2 focuses on one specific data set and presents a novel computational pipeline, that by design, consists of only simplistic components and depends only on very few parameters. By limiting the complexity of the pipeline, we enable fully automatic tuning, i.e., all internal parameters of the proposed method are automatically selected. The parameter choice is achieved in such a way so that the

results correspond to expert-formulated biologically motivated priors. This leads to a tuning-free pipeline and minimizes subjectivity in calculations.

3.1 Amyloid plaques in cleared mouse brains

3.1.1 Motivation

The research of amyloyd plaques is mostly interesting because of the connection between the distribution of these plaques in the brain and the stage of Alzheimer's disease progression. Alzheimer's disease is a chronic neurodegenerative disease that affects tens of millions of people worldwide [PBC+09] and is considered one of the most financially costly diseases in developed countries [MMS98].

The cause of Alzheimer's disease is poorly understood. However, some studies [PBC+09] claim a connection between the disease progression and the number of amyloid plaques in the brains of affected animals. Analysing the distribution of plaques in different brain regions might help to estimate the effectiveness of applied medication.

Automated methods for this problem are expected to serve as an enabling technology, since it is infeasible for an expert to accurately count the number of plaques in the whole-brain due to the very large numbers of them (typical affected brain contains tens of thousands of plaques). Subsampling, as one of the approaches, is very efficient, but can lead to non-representative and subjective results, because plaque distribution is highly non-uniform in different regions of the brain.

Automated large-scale pipelines, on the other hand, can operate on the whole brain volumes and are at the same time less subjective. Even when slightly biased, they usually allow accurate relative comparison between the results of different treatments and can possibly accelerate the development of new types of medications.

3.1.2 Tissue clearing and imaging

Recent advances in tissue preparation and volume processing allow imaging and single-cell analysis of whole brains [TYHD14, KBL+16]. These techniques enable experts to perform various tasks in volumetric analysis, reaching far beyond plaques distribution estimation.

Compared to slice-by-slice imaging techniques, such as serial section electron microscopy, described in chapter 2, tissue clearing with subsequent optical microscopy imaging enables three-dimensional analysis of the whole volume in isotropic setting and reduces imaging and reconstruction artifacts.

The idea of tissue clearing is to change the tissue sample in such a way that most of the tissue becomes transparent for some wavelengths. Preprocessing tissue in such a way makes it possible to capture the whole volume of the sample and structures of interest with optical microscopy. But in order to do so, light scattering should be prevented, which mostly occurs because of the lipids in the tissue. Therefore, most of the tissue clearing methods proceed by removing most of the fats while preserving the structure of the relevent objects in the tissue [Jac13].

One of such methods is called CLARITY [TYHD14], and it works by removing the lipids through the series of chemical treatments. During the extraction process, almost all of the original proteins and nucleic acids are left in place. To achieve the structure preservation, CLARITY employs hydrogel monomers, which link to relevant objects in the tissue, and forms a rigid structure with the relevant objects preserved in place.

Tissue clearing technique that is used for all the experiments in this thesis is *Focused Electrophoretic Tissue Clearing* (FETC) [KBL+16]. It works in a similar way as CLARITY, but allows significantly faster clearing by applying a focused electric field across tissue. It is also more robust and does not require special training to perform, because it uses a design of a clearing chamber reproducible with 3D printing technologies.

For imaging, SPIM microscopy is used [HSB+04]. This microscopy

technique highlights one virtual section of the tissue with light sheet illumination, and images this section with an objective lens, that is orthogonal to a light sheet plane. Moving the illumination plane through the tissue sample allows to capture the whole 3D volume with very high resolution achievable with optical microscopy.

If the clearing process is successful, and the tissue is transparent enough, then the captured volume would perfectly represent the images of highlighted sections. However, real-live process is never perfect, so blurring can occur due to light scattering as the illuminated plane goes further from the objective lens. It is a minor problem when the captured volume is very small, as discussed in the next section, but becomes a serious issue when the whole-organ analysis needs to be performed, which we consider in section 3.2.

3.1.3 Local analysis

The first data set that we consider is very small in terms of volume captured and the number of plaques. Small volume implies that the imaging quality is approximately the same across the data set, and also that local statistics correspond to global ones: there is no discrepancy between different brain regions, between tissue densities and structure distributions.

These properties make a data set quite easy to analyse for even a non-expert human, and allow one to formulate a simple sequence of steps to solve the problem of plaques detection (see figure 3.1).

All the required information formulated by an expert include the following points:

1. high-intensity regions correspond to plagues, background is in general of lower, but non-zero intensity;

2. background intensity is mostly Gaussian noise for this data set;

3. plaques are uniformly-distributed and compact, i.e. mostly ball-like structures.

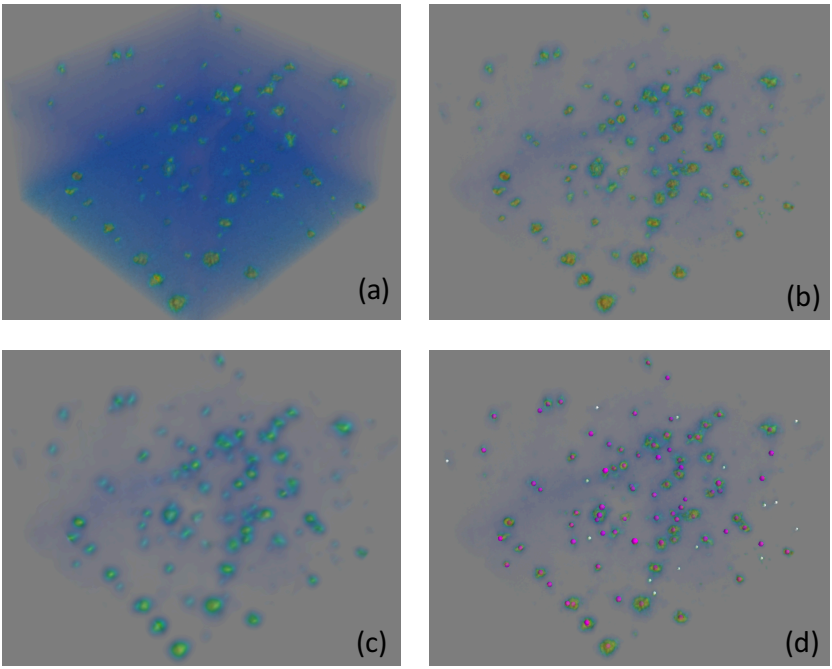

Figure 3.1: A simple algorithm to analyse plaque distribution over a small volume of brain tissue captured with 20× magnification. Because the volume captured is very small, global statistics can be used for the algorithm and the results can be easily verified by a human expert. (a) 3D visualization of the volume. (b) Background-foreground separation with global threshold. (c) Smoothing out plaque regions for further watershed segmentation. (d) The final result of the algorithm (pink dots highlighting plaques).

This description naturally transforms into the following algorithm 1 (see visualization in figure 3.1). First, we remove background voxels to separate foreground from background. Then we smooth the shape of plaques using density estimation with Gaussian kernel [Par62] (i.e. get rid of local maximums of intensity within one plaque regions). And finally, perform watershed segmentation [Mey94] of the density to get plaques as compact segmented regions.

Algorithm 1 Local plaque detection

Require: volume $V \in \mathbb{R}^{D \times N \times M}$, threshold θ, kernel bandwidth h,
$V_{i,j,k} := 0, \forall\ (i,j,k) : V_{i,j,k} < \theta,$ ▷ remove background voxels
$S := \mathrm{KDE}(V, h),$ ▷ kernel density estimation
$M := \{p : S_{i,j,k} > S_{p,q,r}, \forall (p,q,r) \in \epsilon(i,j,k)\},$ ▷ local maximums
$R := \mathrm{Watershed}(S, M)$ ▷ watershed segmentation
return R

The algorithm 1 has only two parameters: θ and h. The first one is estimated given the information about background voxel intensity distribution. We robustly fit Gaussian distribution to the voxel intensities, and select a threshold θ as a 99% percentile of this estimated Gaussian distribution (see figure 3.2). The value of 99% is also somehow manual, yet it proves to nicely work for different data sets, unlike selecting a fixed intensity threshold.

The second parameter h is tuned manually. This manual tuning pursues the idea to determine a final plaque size in line with the expected values. We investigate how this process can be automated to perform competitively on a highly challenging data set in the next section.

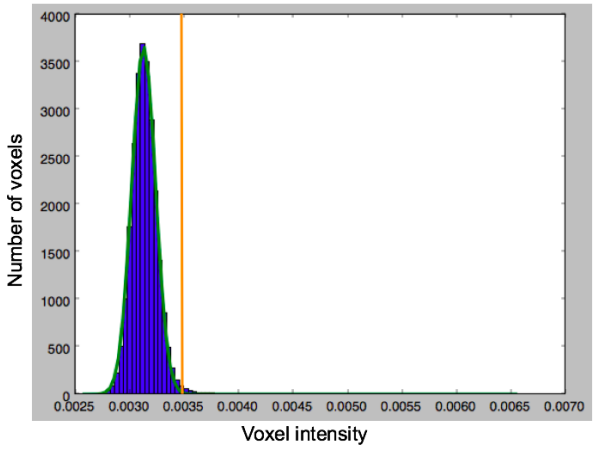

Figure 3.2: Voxel intensity distribution histogram (blue) and the estimated background noise density distribution (green). The value of θ (orange) is selected as a percentile of this distribution.

3.2 Biologically motivated priors for tuning-free algorithms

3.2.1 Whole-brain analysis

In this section we elaborate on the problem of whole-brain plaque analysis. Unlike previously discussed local estimation, the key obstacles are the following:

- global thresholds cannot be used, because the tissue properties differ significantly from one region to another, as well as because of imaging artifacts (see figure 3.3);
- whole-brain analysis needs to be computationally-efficient to be performed on regular hardware in a laboratory environment;
- manual parameter tuning is less feasible, because lower-resolution images provide less tractable information that can vary within

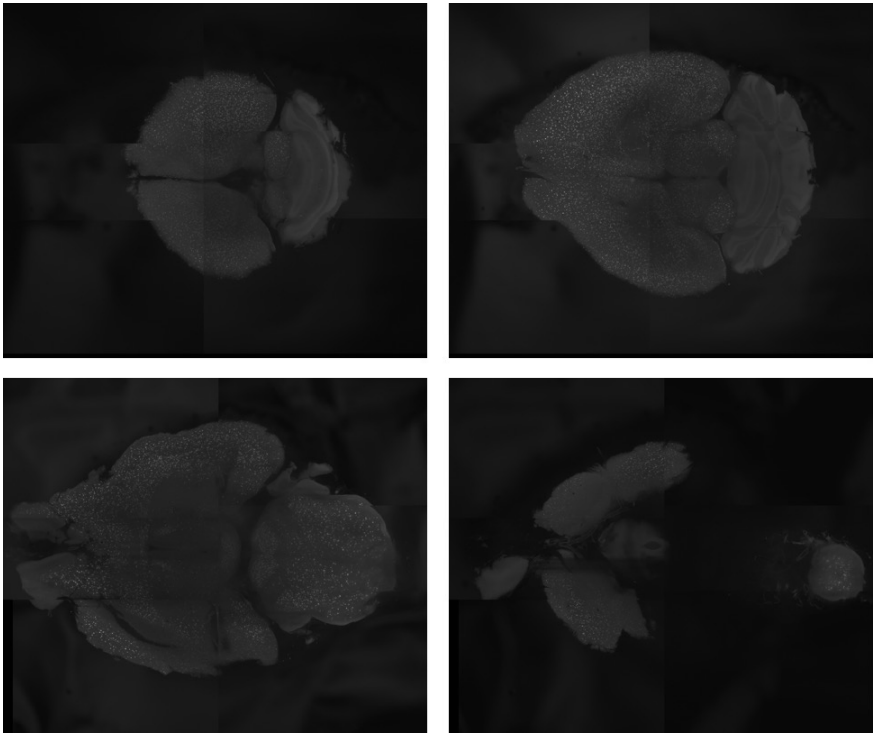

Figure 3.3: Four non-consecutive sections of the whole-brain volume. The data set poses a challenge of analysing the regions with varying properties. Small white dots represent plaques. Non-continuous transitions within one section represent either corrupted tissue, or the artifacts of stitching: because the volume cannot be imaged at once, different sub-volumes (up to 16 in this case) need to be aligned and stitched together after the imaging is done.

the imaged volume;

- furthermore, intensive human involvement can introduce subjectivity into the analysis and make the result incomparable between two experts.

In this section we propose a method that produces analysis of cleared brain volumes with minimum human intervention. Tuning-free fashion of the method limits possible bias or subjectivity in the analysis. The method is computationally efficient and works for a whole-brain scale at up to a single-cell resolution on a standard laptop.

First, we separate the plaques from background using adaptive local thresholding. Then the algorithm segments the brain tissue using random-walker segmentation and aligns it with the reference atlas using 3D registration. Finally, the discovered plaques are filtered, mapped onto biological brain regions and quantified. The method has some internal parameters that are tuned automatically in order to correspond to biologically motivated priors.

The main novelties of the proposed pipeline include the following points.

- We introduce a feedback-loop to fit inner parameters of the algorithm by incorporating biologically motivated priors. We demonstrate this strategy to be a generally-applicable framework and we theoretically justify it in lemma 1.

- The proposed method is non-parametric (or tuning-free) and, therefore, renders the final output to be less sensitive to an experimentalist's possible bias, which is especially important in the pharmaceutical industry.

- The method efficiently analyses data in 3D and robustly estimates not only the number of plaques in a slice, but also plaques volumetric information.

- We also map the plaques to the reference Waxholm space brain atlas [JBB+10], which provides insights in how exactly the plaques

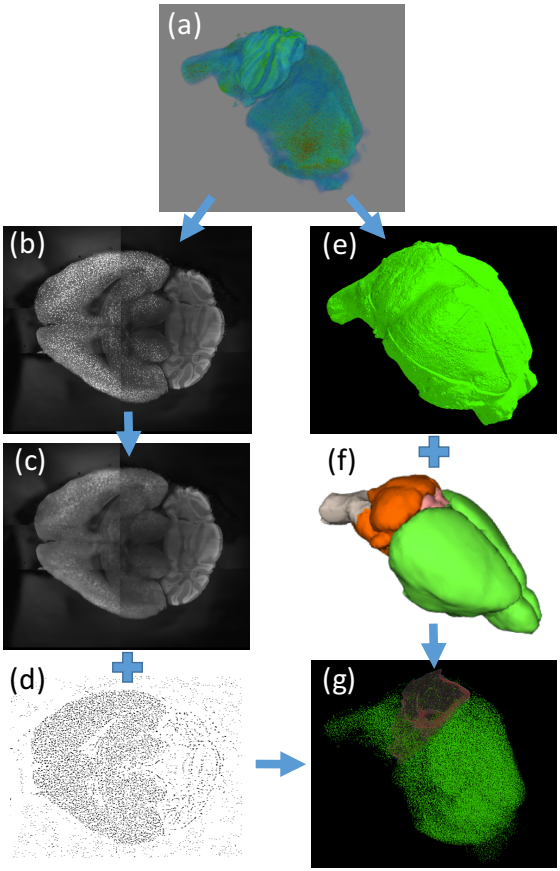

Figure 3.4: The brain volume imaged with SPIM (a) is initially processed per slice to allow for efficient computations. Each slice (b) is separated into background (c) and plaque candidates (d). Then plaque candidates are then aggregated and filtered in 3D. The volume (a) is downsampled for efficiency, then segmented (e) and registered with the reference atlas (f). The atlas is used to map plaques to biological regions. An example of the result (g) shows detected plaques in green and cerebellum highlighted in red.

are distributed in different biological regions and how this distribution changes as the disease progresses.

The high-level pipeline without automated parameter selection is sketched in figure 3.4 and is described in detail in section 3.2.3. The general-purpose approach to fit the inner parameters of the algorithm is sketched in figure 3.5 and described in section 3.2.4.

3.2.2 Related work

Plaques analysis. Many studies have been published on the connection between Alzheimer's disease and the number of plaques in brains. Most of the studies only analyse a single physically cut brain section at a time [PBC+09]. This approach fails to incorporate reasonable volumetric information and does not allow us to perform whole-brain analysis.

To introduce volumetric information, the authors of [JBWB+15] perform analysis on cleared brains in 3D, which follows the idea of the research presented also in the current section. However, their method still works only for small areas manually selected by experts and relies on selecting clustering parameters for detecting the plaques and therefore can lead to subjective results. Our method, on the contrary, enables whole-brain analysis and does not rely on manual tuning and instead uses biological priors to automatically select inner parameters.

Parameter tuning. Sensitivity to parameter selection is especially crucial for unsupervised or weakly supervised problems. In this work we show how to fit the parameters using external knowledge on some properties of the result. One paper that follows a similar approach describes a method to tune kernel clustering parameters [LTSG14]. The method performs grid search over the possible values of parameters and selects the one value that corresponds to an optimal walking time. We generalize this approach and show that it is applicable for a broad range of problems with different types of priors.

3.2.3 Method description

Notation. Let $V \in \mathbb{R}^{D \times N \times M}$ be the initial brain volume with x-, y- and z-sizes being correspondingly N, M and D. We refer to the individual brain slice as $V_i \in \mathbb{R}^{N \times M}$, $i \in \{1, \dots, D\}$. We also introduce a reference atlas $T \in \{0, \dots, N_T\}^{D \times N \times M}$, where N_T is the number of biological regions (we assume same size as V for simplicity) and $T_{i,j,k} = p$ means that voxel (i, j, k) belongs to a region p.

Plaque candidates

We first aim to separate the slice V_i into background $B_i \in \mathbb{R}^{N \times M}$ and plaque candidates mask $C_i \in \{0, 1\}^{N \times M}$. B visualizes the volume in such a way as if there were no plaques. C is an indicator: voxel (i, j, k) can belong to a plaque only if $C_{i,j,k} = 1$ (the opposite is not necessary true, at this stage we only identify candidates, so false positives are expected).

Left-hand side of the figure 3.4 visualizes the initial slice V_i, the brain background B_i and the candidate mask C_i (black dots correspond to C_i equal to 1). We process the data per-slice because of two reasons: for efficiency reasons and in order to compensate for non-uniform lightning of some slices during the SPIM imaging process.

Let $\epsilon(j, k)$ be a neighbourhood of the pixel (j, k), such that this neighbourhood is guaranteed to be larger than a plaque size. In our experiments we define $\epsilon(j, k)$ as a set of pixels (p, q) s.t. $(i - p)^2 + (j - q)^2 \leq r^2$. Here r defines a neighbourhood radius and depends on the image resolution. We set it to be two times larger than the maximum possible plaque diameter in pixels: $r := 10$.

Then we can define background slices B_i as a median filter with the neighbourhood ϵ applied to the original slice V_i. Formally, for every pixel (j, k) we compute the 50% percentile of the distribution of pixel values in $\epsilon(j, k)$:

$$B_{i,j,k} = \text{percentile}_{0.5}\Big(\big\{V_{i,p,q} \big| \, (p, q) \in \epsilon\,(j, k)\big\}\Big)$$

The plaque candidates are the pixels that are significantly brighter

than the background. We formulate this using the threshold γ of the ratio between the two:

$$C_{i,j,k} = 1 \iff \frac{V_{i,j,k}}{B_{i,j,k}} \geq \gamma$$

This procedure results in many false positive noise regions outside of brain tissue (low values of V). We further show how to improve the results by segmenting the tissue and leaving only the plaques within the brain.

Volumetric analysis

In order to progress from candidate plaques mask C to final plaque estimation and to map the discovered plaques to specific biological regions, we perform three steps, also shown in the right-hand side of figure 3.4:

1. Segment the brain background B to get the segmented volume mask $S \in \{0,1\}^{D \times N \times M}$: $S_{i,j,k} = 1 \iff$ voxel (i,j,k) belongs to the brain tissue.

2. Filter C such that it does not include small noisy regions and regions outside of the brain tissue indicated by S.

3. Register atlas volume T to S to get the correspondence of plaque center (i,j,k) to a brain region.

We downsample the brain background B to obtain matrix S (we downsample six times across x- and y-axes and three times across z-axes. Then we upsample matrix S again to match the original volume size. This scale change permits faster computations while preserving resolution that is sufficient for brain tissue segmentation (plaque discovery requires full resolution). But for simplicity we assume below that the matrices S and T has the same size as the original volume matrix V.

First we initialize the random walker segmentation [Gra06] with seeds from regions outside of the brain tissue and inside of it. The idea

of random walker algorithm is that given seeds with known labels, the unlabeled voxels are each imagined to release a random walker, and the probability of this random walker reaching a seed with each label is computed. If the probability of reaching a seed "within" brain is higher than the probability of reaching a seed "outside" of the brain, then the voxel is labelled as also belonging to the brain. This computation may be determined analytically by solving a system of linear equations [Gra06].

Formally, we initialize seeds $S_{i,j,k} = 1$ (as belonging to the brain tissue) if the voxel (i, j, k) has high relative intensity and $S_{i,j,k} = 0$ if the voxel has low intensity:

$$S_{i,j,k} = 1 \iff V_{i,j,k} > \text{percentile}_{0.9}\left(\{V_{p,q,r} | \forall\ p, q, r\}\right)$$

$$S_{i,j,k} = 0 \iff V_{i,j,k} < \text{percentile}_{0.1}\left(\{V_{p,q,r} | \forall\ p, q, r\}\right)$$

To discard small outliers and noise of the imaging process, we filter the initialized values in matrix S using median filtering over the neighbourhood ϵ defined above. Then we run the random walker algorithm to propagate the seeds to the whole volume. Random walker segmentation has one parameter β, which controls the smoothness versus appearance trade-off. We describe how to fit it in the section 3.2.4.

To filter matrix C, the method analyses the connected components [HS91]. A connected component P of matrix C is represented as a set of voxels $\{(i, j, k)\}$. The size of the connected component is then defined as $|P|$. We filter out small regions ($|P| < 5$) and filter the regions that correspond to the areas outside of brain tissue, though leaving only the components P such that $S_{i,j,k} = 1\ \forall (i, j, k) \in P$. The components that are left after filtering represent the discovered plaques.

The final step is to register the atlas volume T to the brain tissue volume S and to assign every plaque to a specific brain region. Any volume registration technique can be applied [MVS99], but we use a simple heuristic approach which appears to be very efficient for our purposes. We first stretch the atlas T to match the size of the brain

tissue in each of the x-, y- and z- directions. Then for every slice $i \in 1, \ldots, D$ we perform 2d affine registration [JS01] and warping of T_i and S_i. Because of continuity in shape of both atlas and brain tissue, the warping is smooth across z-axes.

Algorithm assigns plaque P to brain region with index k if $T_{i,j,k} = k \forall (i,j,k) \in P$. This information is then be aggregated to build the distribution of plaques numbers per biological brain region, which we discuss in section 3.2.5.

3.2.4 Feedback-loop for parameters tuning

The core idea behind the feedback-loop for parameter tuning is to asses the value of the algorithm parameter by validating the solution produced by the algorithm. By changing the parameters of the algorithm and comparing some properties of the solution to the known priors, one can find the most appropriate value, i.e. the value producing the solution in line with prior knowledge. The outline of the process is sketched in figure 3.5.

One exact implementation of the proposed parameter tuning process is described in algorithm 2. This process is able to find the optimal parameters Θ of an arbitrary algorithm $\mathcal{A}$ when the conditions of the following lemma 1 are satisfied.

Lemma 1. *Let the algorithm $\mathcal{A}$ depend on the parameter set Θ. Also assume we have formulated a property f of the output of $\mathcal{A}$ as a statistic $f(\mathcal{A}(\Theta))$. If the function $f(\mathcal{A}(\theta))$ is smooth and strictly monotonic for every $\theta \in \Theta$, then the binary search procedure 2 is able to identify the value of the parameters θ such that $f(\mathcal{A}(\Theta))$ is equal to the desired prior on f up to any tolerance level within a finite amount of steps.*

The proof follows from the basic properties of smooth monotonic functions and of binary search. The formal proof of this lemma is out of scope of this thesis.

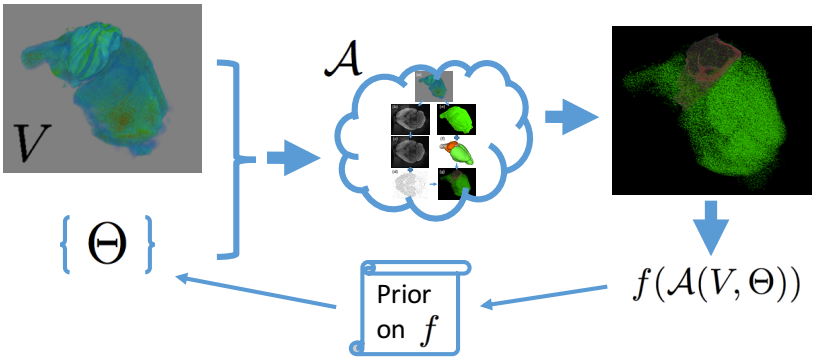

Figure 3.5: The algorithm $\mathcal{A}$ starts with the data V (brain volume) and the initial parameters values Θ. Some statistic $f(\mathcal{A}(V,\Theta))$ is then computed on the results. If this statistic f satisfies the conditions formulated in lemma 1 and the prior knowledge on the expected values of f is known, then we can adjust the values of Θ in a feedback-loop and repeat the whole process until the prior constraints are satisfied. For example, in our case f computes the average size of the plaque, which is known from different studies and therefore can be used to increase or decrease the value of threshold parameter γ.

Algorithm 2 Feedback-loop with binary search

Require: algorithm $\mathcal{A}$, parameter boundaries $\theta_{\min}, \theta_{\max}$, property f, expected property value E, tolerance level ϵ.

repeat
  $\theta := \frac{1}{2}(\theta_{\min} + \theta_{\max})$
  if $\big(f(\mathcal{A}(\theta_{\min})) - E\big)\big(f(\mathcal{A}(\theta)) - E\big)$ **then**
    $\theta_{\min} := \theta$
  else
    $\theta_{\max} := \theta$
  end if
until $||f(\mathcal{A}(\theta)) - E||_2^2 < \epsilon$
return θ

Feedback-loop for plaque analysis

The method for whole-brain plaque analysis described in the previous section depends on some internal parameters, such as threshold γ and smoothness parameter β of the random walker segmentation. These parameters, if tuned by experts, can introduce subjectiveness into the results of the pipeline. On the other hand, if these parameters are to be fixed for different brains, the algorithm will not be able to adapt to varying factors of clearing and imaging.

Here we demonstrate how the feedback-loop can be used to find the optimal value for those parameters. We do that by incorporating biologically motivated priors on the average plaque size and on the volume of the brain, i.e. we define properties f_γ and f_β to be respectively the average size of a plaque and the overall volume of the brain tissue:

$$f_\gamma\left(\mathcal{A}\left(\gamma\right)\right) = \frac{\sum_{\text{plaques } P} |\{P\}|}{\text{number of plaques } P}$$

$$f_\beta\left(\mathcal{A}\left(\beta\right)\right) = \sum_{i,j,k} S_{i,j,k}$$

For those properties, we have some prior desired values coming from biological knowledge. From studies like [PBC+09] we know that f_γ should be approximately $6.5 * 10^{-5}$ mm^3, which corresponds to about 20 voxels in our resolution. Desired value for f_β is known from experimentalists and is unique for every brain, but on average is about $1.5 * 10^7$ voxels.

One can also note that $f_\gamma(\mathcal{A}(\gamma))$ is a strictly monotonic function for $\gamma \in [0, 1]$, and $f_\beta(\mathcal{A}(\beta))$ is strictly monotonic for reasonable values of $\beta \in [10, 10^3]$. Therefore, we can apply use algorithm 2 to find the values of γ and β approximately corresponding to the desired priors.

This approach permits to efficiently estimate the inner parameters γ and β and therefore the final pipeline with this feedback-loop does not require any manual tuning.

One important point to notice is the following: the prior information has to be formulated in an implicit manner, and should not

describe the values being estimated. For example, a prior of the number of plaques would be harmful, because one will always achieve the desired values even for brains with no plaques detecting purely noise. On the other hand, these types of priors can be additionally introduced in Bayesian formulation, as desired, but not enforced.

3.2.5 Experimental results

We apply the described pipeline to six APPPS1 mouse brain volumes [RBK+06]: three young brains (3–4 months) and three old (9 months). The results coincide with the preliminary expectations that the older brains should have significantly more plaques in most of the brain regions.

For example, with the proposed approach we can say that more than three quarters of all the plaques are developing in neocortex. In this area we observe significant difference in both size of the plaques and their numbers. Young brains contain respectively 26859, 19602 and 34152 plaques while old brains respectively 41924, 57292 and 50136 plaques. This result, together with manual result inspection, serves as evidence for the reasonable parameter tuning by the proposed feedback-loop.

From a biological point of view, we are also able to discover that cerebellum contains almost no plaques in young brains, but for older brains the numbers grow very fast, showing how the plaques are propagating through the brain regions: young brains have 177, 428 and 173 plaques while old brains have 1019, 1862 and 2268.

Other biologically-relevant results are discussed in [KBL+16].

Expert-tuning comparison

Validation of the proposed pipeline is challenging, as with any unsupervised problems. Because the plaques are not labeled, there does not exist an error function that can be used to compare different parameter-tuning approaches.

Method	Error
Trained on V_i, estimated on V_i	0.8% (training)
Trained on V_j, estimated on V_j	1.4% (training)
Trained on V_j, estimated on V_i	18% (validation)
Trained on V_i, estimated on V_j	34% (validation)
Ours estimated on V_i	12%
Ours estimated on V_j	15%

Table 3.1: Comparison of the proposed tuning-free approach with cross-validation parameter training. By not overfitting to one specific section, the proposed method achieves better generalization.

To quantitatively validate the approach, we select two sections V_i and V_j of one brain volume for manual analysis. For every section, we ask experts to estimate the number of plaques in these sections.

Then we compare the results of the proposed algorithm (that does not use any information about the number of plaques) with the cross-validation approach. The cross-validation approach finds the optimal value of γ by minimizing the prediction error on section V_i. Then the accuracy of the learned value is estimated on section V_j. The quantitative results are described in table 3.1.

Similar results are achieved when using V_j for training, and V_i for error estimation. Also, experts looking at only one section at a time tend to agree with the values of the parameter found by cross-validation algorithm, even though it poorly generalizes across different brain regions.

This experiment shows how the proposed approach manages to find a reasonable non-subjective solution when both experts focusing on one part of the brain, and supervised techniques with limited labeled data fail.

3.3 Contributions

In this chapter of the thesis we introduce the problem of automated analysis of whole-brain volumes and present a computational pipeline to estimate the distribution of amyloid plaques in cleared mouse brains.

The core components of the method include adaptive plaque candidate proposal, filtering by incorporating volumetric information, atlas alignment and a feedback-loop to tune the inner parameters of the algorithm.

The key novelty of the approach lies in using biologically motivated priors for parameter selection. Incorporating priors on the expected solution properties results in a tuning-free, efficient and non-subjective tool for experimentalists.

Other novelties described in the chapter include the following points.

- The method efficiently analyses data in 3D and robustly estimates not only the number of plaques in a slice, but also plaques volumetric information.
- The proposed method efficiently maps the plaques to the reference Waxholm space brain atlas [JBB+10], which provides insights how exactly the plaques are distributed in different biological regions and how this distribution changes as the disease progresses.
- The described pipeline efficiently performs volumetric analysis on a whole-brain scale as opposed to all currently existing methods.
- An introduced feedback-loop is demonstrated to be a generally-applicable framework and theoretically justify it in lemma 1.

Chapter 4

Transformation-invariance

4.1 Introduction

Often in the experimental sciences, data sets display a parameter dependence that is causally related to the measurement process and that cannot be primarily attributed to the data source. Such dependencies could be lighting conditions in imaging, parameter settings of the experimental apparatus, and other nuisance factors in the data acquisition process. When we like to draw conclusions about the data source and its properties, then these nuisance factors have to be identified and possibly compensated for by problem adapted priors and transformations. Therefore, we discuss a last important type of expert priors that relates to information on nuisance variations in the data set.

Such nuisances can be formulated as a set of transformations s.t. these transformations can be applied to the initial data without affecting the final solution, e.g., classification, clustering, etc., of the problem at hand. An expert biologist might decide in advance that, e.g. the translation of a cell does not matter for the data analysis, because cells are by nature oriented randomly in our samples. In this case the expert formulates the problem as possessing a property of *rotation-invariance*.

Incorporating this type of information, as we show later in the chapter, is an efficient way to distinguish between relevant and irrelevant properties of the data, without formulating hand-crafted rules and manually selecting features. It allows an algorithm to focus only on relevant properties by ignoring irrelevant ones. Both from an experimental and a theoretical perspectives, we argue that incorporating this information leads to smaller errors, faster training and better generalization capacity for various types of data sets than processing the data in their original form. The identification of nuisance influences on data also highlight a core problem in statistical inference: to separate the signal that is informative for choosing hypotheses in the learning process from that part of the signal, which shows little to no dependence on the solution space. The more often discussed problem to distinguish signal from noise could be far easier than the informative signal nuisance signal dichotomy.

While some classes of features permit a transformation-invariant formulation, in most cases these classes are very limited. To exploit the class of features as general as possible, we incorporate the information on transformation-invariance directly into the process of feature learning. Combining the power of feature learning with prior knowledge on possible nuisance variations often results in highly informative features for the current problem, that also preserve the desired properties of transformation-invariance.

4.1.1 Feature learning

One of the most important and critical steps for the overwhelming majority of computer vision tasks is concerned with the feature design. Researchers face a challenging problem of describing local appearance of a patch around the pixel or voxel with a set of features. These local descriptors should be robust and sufficiently rich for further processing with different machine learning algorithms.

A very important but by no means the only example of such an image processing application is medical image segmentation. The problem of constructing relevant features arises in this field in the

most acute way. Experts that label pixels manually often rely only on local appearance, but are unable to mathematically define the features that appear to be most relevant for them.

Standard or **commonly used features** have been developed to address as many computer vision applications as possible. Different filters [mat], SIFT [Low99] and HoG features [ZYCA06] are only few examples of such features. Even though these features work great for some tasks, they are unable to adapt to the specific problems and, therefore, often do not encode all the relevant information.

For some applications experts are able to formalize the desired properties of the object patches based on the local appearance. For such applications **domain-specific features** can be developed. Line filter transform [SB07] is used for blood vessels segmentation, context cue features [BAKF12] – for synapse detection. Domain-specific features proved to be very informative, however the development of these features is time-consuming and expensive while not always possible and it does not generalize to other domains.

Unsupervised feature learning overcomes the domain-specificity, as this approach generates features based on the data itself. The "bag of visual words" representation [YJHN07] and dictionary learning [KDMR+03] for sparse coding are procedures that fall in this category together with denoising autoencoders [VLBM08]. Even though these methods are powerful for data-representation, compression and image restoration, they exhibit serious limitations when applied to supervised problems. This phenomenon happens because neither of the methods rely on the information about the label of the pixels and therefore learns *reconstructive*, not *discriminative* representations.

Supervised feature learning, in contrast, learns the features of the data jointly with learning the classification functions. Sparse coding algorithm can be adapted to this procedure [MBP+08], but only with classification functions limited to linear and bilinear models. Convolutional Neural Networks (CNN) [LGTB97] are able to learn more complex classification functions and more complex feature representation in one optimization procedure, without separating one

from another.

The most successful techniques in different domains usually fall into one of the two categories: either they use very powerful domain-specific features, or they fully rely on supervised feature learning.

In this chapter of the thesis we gradually develop multiple methods to combine the best of these two approaches: to incorporate domain knowledge into the process of supervised feature learning. We specifically focus on transformation-invariance as one very common type of domain knowledge.

4.1.2 Transformation-invariance

Human visual perception proves to be extremely stable to a broad class of variations in scenes. If objects in images are rotated, or scaled, or even non-linearly distorted – in most cases we still can recognise these objects. If we are in advance aware of transformations that can occur in a data set, than this information can be used to design a better recognition algorithm with higher generalization capacity and therefore, higher accuracy than agnostic learners. Being capable of generalizing over different transformations is a very important property of any machine learning approach, and especially of computer vision algorithms.

The set of transformations to be considered highly depends on the task that one has to solve and is usually to be defined by a domain expert. Some common examples are presented in figure 4.1, but a possible transformation set is neither limited to these examples, nor requires to include all of them. For example, rotation-invariance should be used wisely for digit recognition task, since rotating the digit "6" by 180° could lead to its confusion with "9". However, smaller rotations of up to ±15° proved to significantly improve accuracy in the MNIST classification benchmark [CMS12]. Scale-invariance can also harm classification performance if object size is at least somehow informative, for example, in case of classifying healthy cells from cancer cells [SFO+10].

Transformation-invariance is one of the most common types of

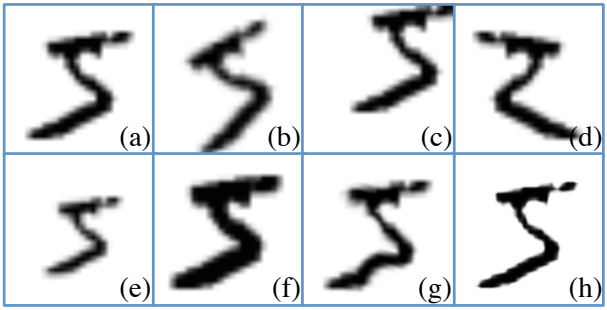

Figure 4.1: Example of transformations ϕ that are usually considered in computer vision tasks applied to a handwritten digit "5" from MNIST data set [LBBH98]. (a) shows the original image X, (b)–(h) show different transformation results $\phi(X)$: rotation (b), shift (translation) (c), reflection (d), scaling (e), morphological operations (f), non-linear distortions (g), brightness, contrast change (h).

expert priors that is often available, but rarely used. The reason for it being that the most powerful feature learning approaches are able to incorporate this information only in a very limited manner, e.g. through data augmentation only. We, on the contrary, propose to reformulate features being learned in such a way, that they will be provably transformation-invariant. We discuss the relations between our approach and augmentation in detail in section 4.4.

The structure of the chapter is the following. In section 4.2 we start with formulating a general method for segmentation and classification problems, called *Convolutional Decision Trees* – this method does not rely on hand-crafted features, but works with raw pixel values, similar to CNN. We then extend this approach to Transformation-Invariant Convolutional Jungles by introducing transformation-invariance in section 4.3. Finally, we generalize the approach to incorporate transformation-invariance into more complex algorithms, such as Deep CNNs in section 4.4, resulting in Transformation-Invariant Pooling.

4.2 Convolutional Decision Trees

In order to overcome the problem of feature design, different methods were proposed to automatically learn discriminative local descriptors [KDMR+03, LB95]. Among them, Deep Convolutional Neural Networks (CNN) [LB95] emerged as probably one of the most attractive methods for supervised feature learning nowadays. The key CNN design strategy is not to separate feature learning process from complex decision function learning, which results in better synergy between the two. All of the approaches proposed in this chapter also follow this strategy.

Convolutional Neural Networks demonstrated to achieve superior performance for different tasks like face recognition [LGTB97], handwritten character recognition [GS08] and neuronal structure segmentation [CGGS12]. On the other hand, CNN suffer from the significant disadvantage that they require very large training data sets and consume an often impractical amount of time to learn the network parameters. Therefore, special hardware cluster architectures have been developed to make CNN applicable for real world tasks [CGGS12]. These constraints render the process of using CNN for end users very difficult and often even unfeasible.

This section presents Convolutional Decision Trees (CDT): a significantly accelerated algorithm for adaptive feature learning and segmentation. It belongs to a family of oblique decision tree algorithms [HKS93] adapted for structural data such as spatial structure of the patches in image segmentation. As we show in the following section 4.3, it also serves as a basis for incorporating transformation-invariance. The algorithm builds on the following ideas:

- we recursively build multivariate (oblique) decision tree,
- each tree split is represented by a convolution kernel, and therefore encodes a feature of the patch around the pixel,
- convolution kernels are learned in a supervised manner while maximizing the informativeness of the split,

- regularization of kernel gradients produces interpretable and generalizable features,

- regularization parameter adaptively changes from one split to another.

These structured oblique trees significantly differ from non-structural by smoothness regularization of the learned kernels. Complexity control of feature learning renders the optimization problem more robust, regularized learning produces more interpretable features and it largely prevents overfitting. The key advantage is that the features learned adaptively for one task are informative and meaningful and, therefore, can be used for other tasks.

These ideas generate a significant performance increase compared to CNN training procedure (up to several orders of magnitude faster training), while keeping the accuracy level comparable to state of the art. The combination of high accuracy and fast training enables anyone to use this algorithm on general purpose single processor desktop hardware.

The procedure demonstrates convincing result improvements both for medical and for natural image segmentation, as demonstrated in section 4.2.4. Here we focus on a segmentation task. Following the method described in [GYR+11], however, the approach can be adapted also for tasks like object detection, tracking and action recognition.

4.2.1 Related work

Feature learning

The proposed algorithm falls into the category of supervised feature learning approaches as it learns the features of the data jointly with learning the classification functions. As discussed in the introduction section 4.1, CDT ideology is similar to the one of a CNN, but with training speed being the key advantage.

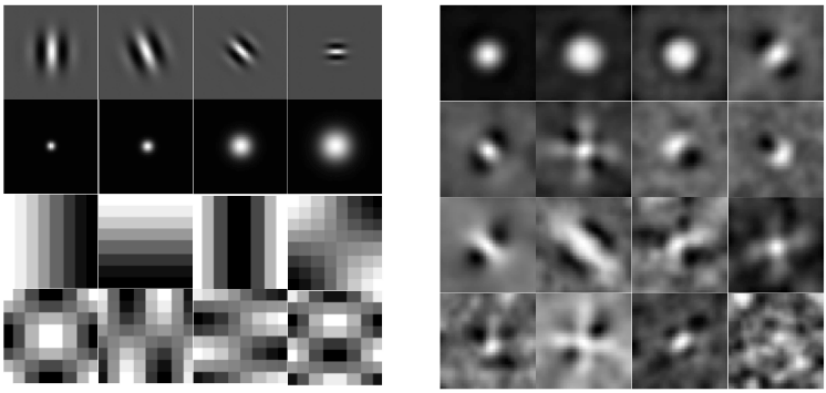

Figure 4.2: Examples of different convolutional kernels: commonly used kernels (left) and the kernels obtained with the proposed algorithm (right). The algorithm finds the most informative kernels in a supervised manner, discovering meaningful kernels that look like gaussian blur, edge filters, corner and junction detectors, texture filters, etc.

For example, for neuronal segmentation data set [CSP+10], the authors of [CGGS12] use CNN that achieves impressive accuracy, but that trains for almost a week using a special-purpose GPU cluster. In contrast, the proposed method combines the flexibility of arbitrary convolutional kernels with the speed of decision tree training. Depending on the task, first reasonable results for smaller trees can be obtained within one hour, while larger trees produce state of the art results in less than 12 hours training with one CPU which makes this method feasible for "plug-and-play" experiments. CNNs with the same training time (smaller CNNs or CNNs trained with different strategies) do not achieve comparable results.

Binary decision trees

Learning decision trees pursues the idea to consecutively split the data space into parts according to a predicate ϕ (s.t. $\phi(x) = 0$ for points x in one half-space, and $\phi(x) = 1$ for x in another half-space). The predicate is selected in such a way that it maximizes a task dependent measure of informativeness, e.g. Information Gain or Gini's diversity index [BFOS84]. $x \in \mathbb{R}^d$ is a vector of features or attributes of the object: x^j represent j-th feature of the object.

Decision trees most commonly are **univariate** [BFOS84]. Formally that means that the form of the predicate is limited to $\phi(x) = [x^j > c]$. Here $[statement]$ denotes Iverson brackets which equals to 1 if *statement* is true and zero otherwise. j and c are the parameters of the split. The choice of only univariate splits limits the computational complexity and it allows us to efficiently find the most informative split. However, it has been demonstrated that in many cases univariate trees require many more splits to learn a classifier and lead to results that are difficult to interpret [HKS93].

Therefore **multivariate (oblique) trees** were proposed [SM80, BFOS84, HKS93], that allow the predicate to learn arbitrary linear splits: $\phi(x) = [x^T\beta > c]$. Here $x^T\beta$ is a linear combination of the attributes, $\beta \in \mathbb{R}^d$ and c are the parameters of the split. Depending on the criteria of informativeness, most algorithms only return locally

optimal splits. Optimization procedures to find the parameters maximizing informativeness of the split include local clustering [SM80], hill climbing [BFOS84] and general-purpose optimization techniques such as simulated annealing [HKS93].

The proposed algorithm develops the idea of oblique trees for learning convolution kernels in the context of image segmentation problems. Through regularization it incorporates the structural information about the spatial neighborhood of the pixels. Introducing this regularization helps the learned splits to be more interpretable and the optimization problem to be more robust.

4.2.2 Method description: split

Notation. As a training set we assume K pixels $i \in \{1, \ldots, K\}$ with associated binary labels $y_i \in \{-1, 1\}$. Local pixel appearance is described with a patch around it. Let the size of a patch be $w \times w$, then each pixel i is represented with w^2 intensities of the pixels in the patch. In homogeneous coordinates, pixel i is described by a vector $x_i \in \mathbb{R}^{w^2+1}$ with $x_{i,1} \equiv 1$. All the vectors stored row-wise form a data-matrix $X \in \mathbb{R}^{K \times w^2+1}$, $X = [x_1, \ldots, x_K]^T$.

The main idea of the method is to find a smooth convolution kernel that would be informative and discriminative for separating one class from another. A kernel of a convolution is again a $w \times w$ matrix. We also extend a vectorized kernel with a shift parameter b for the predicate. We define vector β to encodes both shift and kernel parameters: $\beta \in \mathbb{R}^{w^2+1}$, $\beta_1 \equiv b$ and $\beta_{2:w^2+1}$ encodes the kernel of the convolution.

The predicate form can be now defined as $\phi(x_i, \beta) = [\beta^T x_i > 0]$. As here we care about the sign of the convolution, we also introduce the constraint $||\beta||_2^2 = 1$ to overcome the disambiguities induced by different scalings.

Information Gain

We want to estimate the parameter vector β that would maximize the information gain $\mathrm{IG}(\beta)$. Information gain depends on the distribution of positive and negative samples before and after a split with the predicate ϕ. The intuition behind the information gain is that it shows how much does the entropy changes after the split is performed.

Let us define P as the number of all positive samples: $P = \sum_i [y_i = 1]$, $N = K - P$ — the number of negative samples. After the split, the half-space, where $\phi(x, \beta) = 1$, will contain k samples: p positive and n negative. $n = \sum_i [\phi(x_i, \beta) = 1]$, $p = \sum_i [\phi(x_i, \beta) = 1, y_i = 1]$, $n = k - p$. p, n and k depend on the parameters β. Then

$$\mathrm{IG}(\beta) = H\left(\frac{P}{K}\right) - \left(\frac{k}{K} H\left(\frac{p}{k}\right) + \frac{K-k}{K} H\left(\frac{P-p}{K-k}\right)\right) \tag{4.1}$$

where H denotes the Bernoulli entropy:

$$H(q) = -q \log_2 q - (1-q) \log_2 (1-q).$$

Then the problem of finding the most informative split is formalized as follows:

$$\beta \in \arg\max_{\beta} \mathrm{IG}(\beta) \tag{4.2}$$

Unfortunately, the maximum of $\mathrm{IG}(\beta)$ cannot be found efficiently because of discontinuity in $\phi(x_i, \beta)$ as it contains an indicator function $[\beta^T x_i > 0]$. To overcome this issue we use an approximation from [MTS+13]:

$$\hat{\phi}_\alpha(x_i, \beta) = \frac{1}{1 + \exp(-\alpha \beta^T x_i)} \tag{4.3}$$

We also introduce $\hat{p}_\alpha = \sum_{i:y_i=1} \hat{\phi}_\alpha(x_i, \beta)$, $\hat{n}_\alpha = \sum_{i:y_i=-1} \hat{\phi}_\alpha(x_i, \beta)$, $\hat{k}_\alpha = \hat{p}_\alpha + \hat{n}_\alpha$. Then the information gain can be approximated with

$$\hat{\mathrm{IG}}(\beta) = H\left(\frac{P}{K}\right) - \left(\frac{\hat{k}_\alpha}{K} H\left(\frac{\hat{p}_\alpha}{\hat{k}_\alpha}\right) + \frac{K - \hat{k}_\alpha}{K} H\left(\frac{P - \hat{p}_\alpha}{K - \hat{k}_\alpha}\right)\right) \quad (4.4)$$

It is easy to see that $\hat{\phi}_\alpha(x_i, \beta)$ converges to $\phi(x_i, \beta)$ in the limit $\alpha \to \infty$. Also $\hat{p}_\alpha$ and $\hat{n}_\alpha$ converges to p and n, respectively. This asymptotics renders it possible to solve the original problem (4.2) by estimating a limit process, i.e., we investigate the sequence of solutions of a relaxed problem for increasing α:

$$\beta \in \arg\max_\beta \mathrm{IG}(\beta) = \lim_{\alpha \to +\infty} \arg\max_\beta \hat{\mathrm{IG}}_\alpha(\beta)$$

Regularization

Maximizing the information gain with respect to β usually results in a split that separates the classes, but unfortunately not interpretable (see for example figure 4.3). More than that, when the number of training samples goes to the range of $O(w^2)$ (approximately equals to the number of parameters), the linear split model starts to overfit. This problem requires us to introduce a regularization parameter λ that penalizes the complexity of the learned kernel parameters β. We want to assure that the kernel is smooth, and, therefore, we penalize the gradient of the kernel:

$$\beta_\alpha \in \arg\max_\beta L_\alpha(\beta) \quad \text{with} \quad L_\alpha(\beta) = \hat{\mathrm{IG}}_\alpha(\beta) - \lambda ||\Gamma\beta||_2^2 \quad (4.5)$$

Here $\Gamma \in \mathbb{R}^{2w(w-1)\ \mathrm{x}\ (w^2+1)}$ is a matrix of a 2D differentiation operator in a vectorized space, that is a Tikhonov regularization matrix.

Regularization serves two main goals. First of all, it guarantees interpretability of the kernels learned (see figures 4.2 and 4.3). And second, from an optimization point of view, a strictly concave regularization term steers the gradient descent optimization algorithm out of local minima.

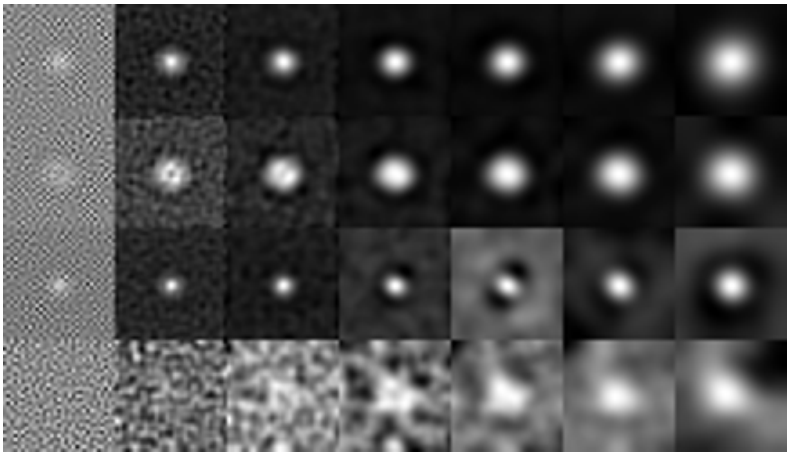

Figure 4.3: Examples of convolution kernels learned with different regularization parameters λ. Each row represents a kernel from different levels of CDT (respectively from 1 to 4). Each column stands for different regularization parameters: 0.001, 0.01, 0.1, 0.5, 2, 10. Increasing regularization helps the learned features to be more interpretable (compare first columns with the last ones). However, increasing λ too much results in smoothing out relevant information (for example the orientation of the third kernel disappears in the last two columns).

Optimization

In practice, we need to choose an initial point and the gradient of the functional to effectively find a solution of the problem 4.5. As initial point, we use the solution to a simple regularized linear regression:

$$\beta_0 = \arg\min_{\beta} \frac{1}{K}||X\beta - Y||_2^2 + \lambda||\Gamma\beta||_2^2 \tag{4.6}$$

Here $Y = [y_1, \ldots, y_K]^T$ is a vector of all the responses and Γ denotes a Tikhonov matrix associated with the regularization above. The analysis is equivalent to ridge regression, except of the form of the Γ matrix. The analytical solution to problem (4.6) is equal to

$$\beta_0 = \left(\frac{1}{K}X^TX + \lambda\Gamma^T\Gamma\right)^{-1} X^TY.$$

The derivative of the functional $L_\alpha(\beta)$ in (4.5) can be also found analytically:

$$\frac{dL_\alpha}{d\beta} = \frac{1}{K}\sum_i \frac{\alpha\exp(\alpha\beta^Tx_i)}{(1+\exp(\alpha\beta^Tx_i))^2}x_i\left(-\log_2\frac{k}{K-k} + [y_i = 1]\log_2\frac{p}{P-p} + [y_i = -1]\log_2\frac{n}{N-n}\right) - 2\lambda\Gamma^T\Gamma\beta \tag{4.7}$$

As an optimization algorithm, we employ Quasi-Newton Limited-memory BFGS (L-BFGS) [Noc80] that estimates Hessian with low-rank approximation and, therefore, selects optimal step size.

Assuming optimization procedure as a subroutine called `L-BFGS`, we sketch the algorithm that finds one split by learning the most informative convolution kernel in algorithm 3. We do not directly estimate the limit in equation 4.2, but instead we iteratively increase α and initialize the optimization procedure in the next step with the solution in the previous step.

Algorithm 3 Function `findSplit`: learning most informative split

Require: training samples $x_i, i = 1, \ldots, K$ with classes y_i; λ
  $\beta_0 := \left(\frac{1}{K} X^T X + \lambda^2 \Gamma^T \Gamma\right)^{-1} X^T Y$ ▷ initialize with MSE solution
  $\beta_0 := \beta_0 / ||\beta_0||_2^2$ ▷ project on the unit sphere
  Set $\alpha := 1$
  repeat
    $\beta_\alpha := \texttt{L-BFGS}(L_\alpha(\cdot), \frac{dL_\alpha}{d\beta}(\cdot), \beta_{\alpha-1})$ ▷ find $\arg\max_\beta L_\alpha(\beta)$
    $\beta_\alpha := \beta_\alpha / ||\beta_\alpha||_2^2$ ▷ project on the unit sphere
    $\alpha := \alpha + 1$
  until $||\beta_\alpha - \beta_{\alpha-1}||_2^2 < \epsilon$ OR $\alpha >$ MaxIterations
  return β_α

4.2.3 Method description: tree

How can we use the procedure `findSplit` to build a classifier? A well-known idea is to recursively split the data space into parts. The recursion stops when we achieve certainty about the label of every part. All sequential splits are encoded in a binary decision tree.

The idea is very straightforward, so we do not discuss it in detail, except for one important question. So far we defined the regularization parameter λ for only one split, but in principle we can change it from split to split, or from one layer of the tree to the next. Experiments show that choosing one parameter λ for all splits in a tree often results in kernel overfitting as the tree grows large.

Assume that we want to find two splits in two different parts A and B with volumes respectively $\mathrm{Vol}(A)$ and $\mathrm{Vol}(B)$. The relation between λ_A and λ_B for this two problems can be established from the following intuition: we want the range of both problems to be the same: for the data part $\frac{1}{K}||X\beta - Y||_2^2$ and for the regularization part $\lambda||\Gamma\beta||_2^2$. We provide the following heuristic rule to adaptively change regularization coefficient:

$$\lambda_B = \lambda_A \left(\frac{\mathrm{Vol}(A)}{\mathrm{Vol}(B)}\right)^{2/(w^2+1)}.$$

Lemma 1 explains the choice of this heuristic rule and assumptions behind the choice.

We can compute the volumes ratio $\frac{\text{Vol}(A)}{\text{Vol}(B)}$ exactly as both compact regions are polyhedra, but in practice we approximate this ratio with just the fraction of the data points falling into each of the compacts A and B. The final recursive algorithm for building the convolutional decision tree is sketched in algorithm 4.

Algorithm 4 Function `buildTree` for decision tree construction

Require: set of indices I, λ, MaxSamples
  $P := \sum_{i \in I}[y_i = 1]$;
  $N := \sum_{i \in I}[y_i = -1]$;
  treeStruct.answer $= \frac{P}{P+N}$
  if $P <$ `MaxSamples` or $N <$ `MaxSamples` **then** ▷ terminal node
    treeStruct.left $= null$
    treeStruct.right $= null$
  else ▷ split recursively
    $\beta :=$ `findSplit`$(x_i, y_i, \forall i \in I; \lambda)$
    $I_{\text{left}} := \{i \in I : \beta^T x_i > 0\}$
    $I_{\text{right}} := \{i \in I : \beta^T x_i <= 0\}$
    $\lambda_{\text{left}} = \lambda \left(\frac{|I|}{|I_{\text{left}}|}\right)^{2/(w^2+1)}$ ▷ change lambda
    $\lambda_{\text{right}} = \lambda \left(\frac{|I|}{|I_{\text{right}}|}\right)^{2/(w^2+1)}$
    treeStruct.left $=$ `buildTree`$(I_{\text{left}}, \lambda_{\text{left}}, \text{MaxSamples})$
    treeStruct.right $=$ `buildTree`$(I_{\text{right}}, \lambda_{\text{right}}, \text{MaxSamples})$
  end if
  return treeStruct

Lemma 1. *Assume that training data points are uniformly distributed in a compact set A and in a compact set B that is equal A up to a rescaling constant. Then the optimal relation between λ_A and λ_B is*

defined through the following equation:

$$\frac{\lambda_B}{\lambda_A} = \left(\frac{Vol(A)}{Vol(B)}\right)^{2/(w^2+1)},$$

where optimality is defined through the range of problems solved for A and B, i.e. we want both first and second term of the functional 4.6 to be of similar values for A and B.

Proof. From the assumption we make, A is equal to B with some scaling factor c, where scaling is applied in every dimension.

Then the volumes are also connected multiplicatively with the factor c^{w^2+1}, where w^2+1 is dimensionality. Therefore $\frac{\text{Vol}(A)}{\text{Vol}(B)} = c^{w^2+1}$ and $c = \left(\frac{\text{Vol}(A)}{\text{Vol}(B)}\right)^{1/(w^2+1)}$.

Let D_A be the data matrix of the points contained in A and D_B – the data points contained in B. As D_A is uniformly distributed in A and there is a multiplicative scale constant from B to A, up to renumerating the samples we can approximate $D_A \simeq cD_B$.

We want the first term of functional 4.6 to be of similar values:

$$\frac{1}{|K_A|}||D_A\beta_A - Y||_2^2 \simeq \frac{1}{|K_B|}||D_B\beta_B - Y||_2^2.$$

To achieve that, the values of β_A should be rescaled inversely to the relation between the data matrices: $\beta_A \simeq \frac{1}{c}\beta_B$.

Incorporating the last equation into $\lambda_A||\Gamma\beta_A||_2^2 \simeq \lambda_B||\Gamma\beta_B||_2^2$ (the second term of functional 4.6), we get $\lambda_A\frac{1}{c^2}||\Gamma\beta_B||_2^2 \simeq \lambda_B||\Gamma\beta_B||_2^2$.

Dividing by $||\Gamma\beta_B||_2^2$ and plugging c finishes the proof. □

4.2.4 Experiments

Experimental settings

We test Convolutional Decision Trees on biological and natural image data sets. First 2/3 of the images are selected for training, and accuracy is reported on the last 1/3. Because in both data sets the classes

are imbalanced, we measure the accuracy in F-score: a commonly used metric that combines precision and recall:

$$\text{F-score} = \frac{2\text{Precision Recall}}{\text{Precision} + \text{Recall}}.$$

We obtain probability maps inferred by the proposed algorithm with the following parameters fixed for both data sets: $w = 31$, $\lambda = 0.5$ (initial value for the first call of `buildTree` function) and `MaxSamples` $= 50$.

All the experiments are performed on a single AMD Opteron 6174 CPU. The speed/accuracy tradeoff is controlled by the number of iterations of the `L-BFGS` subroutine. We set it in such a way that all the experiments finish within 12 hours (overnight experiment).

To get the final segmentation from the probability maps, we apply standard Graph Cut algorithm [BK04] with the parameters selected by 5-fold cross-validation.

Neuronal segmentation data set

As an example of a biological data set, we use the neuronal segmentation data set described in section 2.2.2.

The best results on this data set are achieved using Convolutional Neural Networks. In terms of F-score, the accuracy of this algorithm is approximately the same as the accuracy of a human expert [CGGS12]. Our results appear to be just 2.2% worse in absolute values (82.9% CDT vs 85.1% CNN), however, the CNN training time for this data set is around one week using GPU, comparing to just 12 hours for the proposed method. Training CNN for 12 hours produces results comparable to the other state of the art method that trains in a reasonable time, such as our results described in section 2. A Random Forest (RF) algorithm with specially designed features produces a probability map that is then segmented with a Graph Cut algorithm that uses special potentials. Even though we use a simpler segmentation algorithm, CDT produces better probability maps.

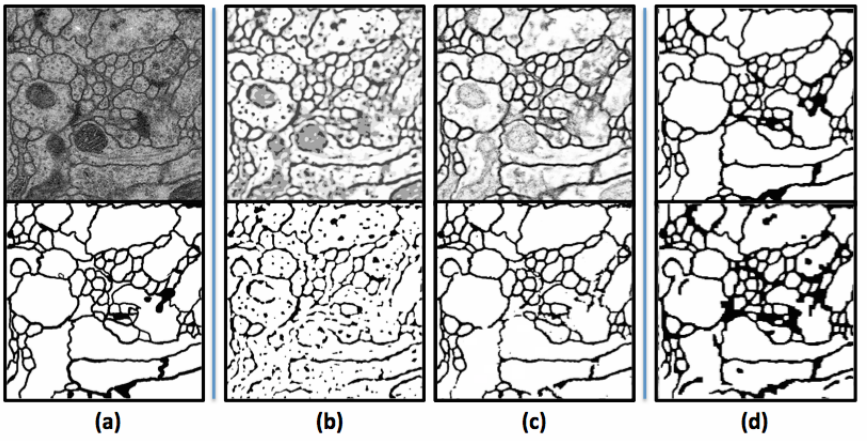

Figure 4.4: The results of the proposed algorithm on a neuronal segmentation data set. Column (a) shows the input image and the ground truth. Columns (b) and (c) demonstrate the qualitative results of the algorithm for the small tree (depth = 3) and the full tree (depth = 17). The results include the probability maps (top) with a Graph Cut segmentation (bottom). The last column (d) shows the results of CNN (top) and RF with predefined features (bottom). Qualitatively the results are comparable with CNN and looks much better than RF results.

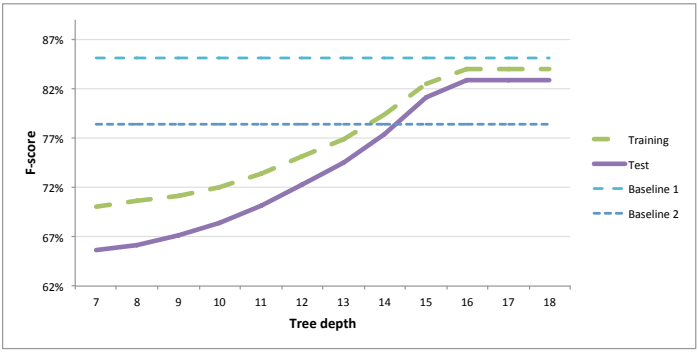

Figure 4.5: Quantitative comparison. Baseline 1 is a CNN [CGGS12] which produces slightly better results, but is infeasible to train on a single CPU. Baseline 2 is a RF with Graph Cut segmentation (described in detail in chapter 2), which we outperform by 4.5%.

Quantitatively this choice results in 4.5% increase in F-score (82.9% CDT vs 78.4% RF).

Weizmann Horse data set

The Weizmann Horse data set [BU02] is well-known in the computer vision community. It consist of 328 manually labelled images of horses in different environments.

There are many methods that perform well on this data set [BU02, GYR+11, ZCY10]. As a baseline we consider general purpose segmentation method based on superpixel grouping [LSD12]. This method produces the best results on this data set compared to methods that do not use prior information: 79.7%. Quantitatively we achieve 80.4% and outperform it by 0.7% (insignificantly). There are also other methods that exploit domain-specific prior information on the shape of the horse silhouette [ZCY10] and achieve superior results of 89.2% (up to 8.8% better). However, we do not compare with them as they are limited to specific segmentation tasks where shape information is

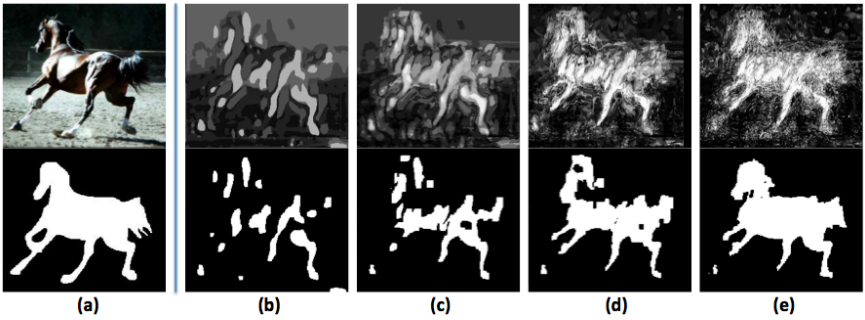

Figure 4.6: The results on the Weizmann Horse data set. Left column (a) shows the input image and the ground truth label. Each of the following columns (b)–(e) demonstrates the qualitative results of the algorithm for different tree sizes (respectively 4, 8, 12, 18). The results include the probability map (top) and the Graph Cut segmentation (bottom). Qualitatively the results improve significantly as the tree grows and more advanced features are learned.

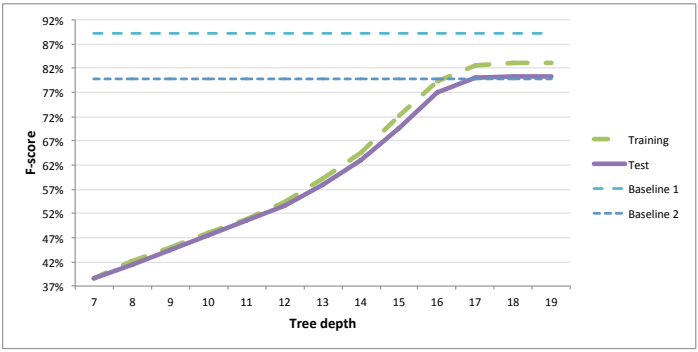

Figure 4.7: Quantitative comparison of the proposed algorithm with two state of the art approaches. Our method slightly outperforms baseline 2 that is the best approach across methods with no prior information [LSD12]. Baseline 1 is better by 8.7%, but uses domain-specific prior information [ZCY10].

know a priori and our method is applicable to a much broader class of segmentation tasks.

4.3 Transformation-Invariant Jungles

The importance of the transformation-invariance to different computer vision problems is difficult to overestimate. The transformation-invariance is one of the most common types of expert-formulated priors that helps to distinguish between relevant object features from variability of irrelevant nuisance factors, as discussed in section 4.1.

In this section we present an algorithm called Transformation-Invariant Convolutional Jungles (TICJ), that is based on the previously discussed Convolutional Decision Trees (CDT), but supports to efficiently incorporate transformation-invariance in the learning process.

As a first step, expert needs to carefully identify a set of trans-

formations that most likely do not affect the results. Once a set of transformations for a given task is known and fixed, there are three main ways to incorporate this prior knowledge: (i) change the data set itself, (ii) use transformation-invariant features or (iii) modify the learning algorithm. Arguably the most popular approach is the first one, which enlarges the original data set by adding the images that were transformed according to our prior beliefs. This strategy enables the recognition algorithm to observe all the instantiations of the data and, in case of flexible models to adapt to all considered transformations.

Enlarging the data set, however, implies also to significantly extend the training time and it requires extensive computational resources. But very large data sets pose only one of the problems: to cope with larger variations in data sets, data analysts usually have to increase the number of parameters in modeling, requiring even more training time, more memory, and posing the risk of over- or underfitting. Therefore, current research investigates advanced techniques to avoid this pitfall, as reviewed in section 4.3.1.

The proposed method is inspired by the idea of the pooling operation, that preserves some local invariances and seems to be biologically plausible [RP99]. We follow this idea to incorporate prior knowledge about the transformations through learning transformation-invariant features of the images. We define these features through the convolutional kernels, but instead of convolving the image itself with the kernel, we propose to compute the maximum over many convolutions: with the given image, and with all the considered transformations of this image. This nonlinear operation assures transformation-invariance, as the value of the maximum is exactly the same for the original image, and the image that was transformed (see lemma 2 for more details).

Section 4.3.2 discusses how these features can be efficiently trained in a supervised manner to fit the needs of the specific task, and how they can be regularized to get interpretable transformation-invariant features that generalize well.

The proposed algorithm TICJ uses this transformation-invariant feature learning procedure to build an image classification algorithm (or per-pixel segmentation algorithm). This goal is achieved by iteratively learning the features and combining them together in a feed-forward modification of the Decision Jungles algorithm [SSK+13b]. Comparing to Decision Trees [Qui86], this algorithm proves to better prevent overfitting, and also efficiently works with limited memory constraints. The combination of the proposed feature learning algorithm and the proposed modification of Decision Jungles allows us to achieve state of the art results with modest training time, as described in section 4.3.2 in detail.

The main properties and contributions of the proposed method are summarized as follows:

- Transformation-invariant feature learning allows us to incorporate any types of transformation invariances as prior constraints. This method results in good generalization without enlarging the original data set size or the parameter space.

- Regularization is enforced in two different ways by the learning method and it serves the purpose of producing interpretable features. These features are easy to debug, since they are defined through convolutional kernels and they support visual inspection (see figure 4.8). Regularization also helps when the data set is small: as we show in the experimental section 4.3.3, TICJ can efficiently be applied for data sets starting from only tens of images.

- The final classification algorithm is computationally very efficient and, unlike many state of the art techniques, can be easily run on a single CPU within a modest training time. In our experiments, the proposed method is up to two orders of magnitude faster than, for example, Deep Neural Networks, while achieving state of the art classification performance (see section 4.3.3).

- The method has only few hyperparameters, and we show in section 4.3.2 how they can be efficiently tuned. This simplicity of the method makes it highly suitable for plug-and-play experiments in comparison to many other modern computer vision techniques.

We also propose a modification of the node clustering technique for Decision Jungles and thereby, we overcome the problem of global clustering that produced poor results as mentioned in the original Decision Jungles paper [SSK+13b]. This contribution is discussed in more details in section 4.3.2.

4.3.1 Related work

Predefined transformation-invariant features

One of the easiest ways to incorporate partial transformation-invariances is to use special types of predefined, often "hand-crafted" features, i.e., the scale-invariant feature transform (SIFT) [Low99] or its rotation-invariant modification RIFT (rotation-invariant feature transform) [LSP+04] have proved to boost performance in a broad range of image processing applications and imaging modalities.

These features are designed to be general-purpose and transformation-invariant, and they satisfactorily solve the task in many cases. However, they are limited in two ways: they only can incorporate very specific transformations, and they do not adapt to the task being solved.

One of the ways to overcome the second deficit is to design hand-crafted features for the specific task. Line filter transform [SB07] for blood vessel segmentation is one of the examples of domain-specific features, which is also rotation-invariant. But designing the features manually is time-consuming and expensive while not always possible.

The proposed method, in contrast to predefined features, not only learns the features in a supervised manner, but also allows one to incorporate any types of invariances.

Transformation-invariant feature learning

Instead of designing features for every different task, one could learn these features automatically such that for every data set a set of learned features would be the most representative and/or discriminative one. Four approaches that follow this idea are "bag of visual words" (BOVW) [PCI+08], Convolutional Decision Trees (CDT, introduced in the previous section 4.2), Sparse Coding [YYH10] and different types of Neural Networks.

Some of these classes can posses the property of transformation-invariance. For example, BOVW does not distinguish the positions in which the "visual word" occurs, and therefore it is a shift-invariant method. A modification of BOVW method, called RBOWV [YN10] also learns features invariant to rotation.

CDT trains the features as convolutional kernels, which resembles our proposal. However, CDT does not permit to incorporate transformation-invariance. Another key difference between the method proposed in this chapter and CTD is that we solve a convex approximation to the information gain formulation of the split objective. The approximation produces comparable results to the information gain maximization procedure, but allows us to find the solution significantly faster.

Sparse Coding algorithm can be also modified to learn overcomplete shift-invariant image representation as presented in the paper [YYH10]. We consider neural networks in the following section.

The proposed algorithm incorporates various ideas of the above methods to learn features from data and, simultaneously, to adapt to the specific task. But unlike both BOVW and Sparse Coding algorithms, it allows us to incorporate many different types of transformations.

Transformation-invariant neural networks

Deep Neural Network architectures are the richest model classes used nowadays in computer vision and they enable very impressive results

in many tasks. Because of their richness, many modifications can be implemented to incorporate different types of prior knowledge in the training process itself. Not surprisingly, transformation-invariance is also actively discussed in this field.

The most commonly used property of Convolutional Neural Networks that enables some transformation invariance is a subsampling layer [LB95] with *max-pooling*. Because the maximum is taken over the neighbouring pixels, local one-pixel shifts usually do not change the output of the subsampling layer. A more general pooling operation [RP99] allows one to also consider local rotation and scale changes. As usually many layers are stacked in a hierarchy on top of each other, the window size for local invariances increases.

Other techniques that support invariances to a rich transformation class include topographic filter maps [KRFL09], that learns features invariant to rotation, shift and scale changes, and the algorithm presented in [SL12], where local transformations that can be approximated as linear transformations. However, neither of these approaches can learn arbitrary set of transformations. Another simple approach that works without enlarging the data set and the model size is presented in [GM87] and [CMS12]. The idea is to train different models with the same topology but using different data sets: the original data set, and the transformed data sets (one model for every transformation considered). Then either the weights of these models are averaged to produce one new model, or the outputs of the networks vote for a majority, forming an ensemble of models. This last approach is widely used and we compare our algorithms to it as a baseline in one of our experiments (see section 4.3.3).

4.3.2 Method description

Notation

Let us consider an image classification data set with K classes that consists of N images. Let $X_i \in \mathbb{R}^{w \times w}$ be the i-th image in this data set represented by a square real-valued matrix of pixel intensities,

where $i = 1, \ldots, N$ and w is the size of the image. For simplicity we consider square images, however, the method naturally generalizes also to rectangular images. In homogeneous coordinates, the image X_i is described by a vector $x_i \in \mathbb{R}^{w^2+1}$ with $x_{i,1} \equiv 1$.

Every image X_i has an assigned class $y_i \in \{1, \ldots, K\}$. The task of an image classification algorithm is to return a class estimate $\hat{y}$ for a new unobserved image X.

Assume Φ to be a set of all considered transformations. $\Phi = \{\phi_1, \ldots, \phi_T\}$, where ϕ_t denotes a transformation function and T specifies the number of transformations. If X is an image, then $\phi(X)$ represents a transformed image of the same size $w \times w$. For simplicity of notation, $\phi(x)$ also denotes an extended vectorized representation of the transformed image $\phi(X)$. Φ always includes the identity transformation $\phi_0 : \phi_0(X) \equiv X$.

The reader should notice that ϕ can represent either one of the simple transformations shown in figure 4.1, or the combination of these transformations. For example, ϕ_3 could be the composition of ϕ_1 and ϕ_2: $\phi_3 = \phi_1 \circ \phi_2$ means that $\phi_3(\cdot) = \phi_1(\phi_2(\cdot))$.

Transformation-invariant feature definition

As discussed in section 4.3, we parametrize a feature with a convolutional kernel $\theta \in \mathbb{R}^{w^2+1}$. The value of the feature for an image X is given by:

$$f_\theta(x) = \max_{\phi \in \Phi} \theta^T \phi(x) \tag{4.8}$$

Because of the maximum operation, this equation in most cases gives exactly the same result $f_\theta(x)$ for the image X itself, and for the transformations of this image $\phi(X)$. Lemma 2 formulates the conditions on the set Φ for which this holds true.

Lemma 2. *The feature of the image X defined in equation 4.8 is transformation-invariant if the set Φ of all possible transformations forms a group, i.e. satisfies the axioms of closure, associativity, invertibility and identity.*

Proof. In order to prove this statement, the value of the feature has to be the same for all the transformations of the image. Since Φ always contains an identity transformation, we can compare the value of the feature with the value of the feature for the identity transformation ϕ_0. So we need to show that $f_\theta(\psi(X)) = f_\theta(\phi_0(X)) = f_\theta(X), \forall \psi \in \Phi$.

For any transformation $\psi \in \Phi$ the following holds:

$$f_\theta(\psi(x)) = \max_{\phi \in \Phi} \theta^T \phi(\psi(x)) = \max_{\varphi = \phi \circ \psi:\ \phi \in \Phi} \theta^T \varphi(x).$$

The closure axiom implies

$$\{\phi \circ \psi : \phi \in \Phi\} \subseteq \Phi. \tag{4.9}$$

On the other hand, invertibility axiom assure the existence of an inverse, $\forall \psi \in \Phi, \exists\ \psi^{-1}$. Furthermore, $\Phi \supseteq \{\phi \circ \psi^{-1}, \phi \in \Phi\} =: \Psi$ (as for Ψ we select only the elements of the set Φ that can be represented through a composition with ψ^{-1}).

Therefore,

$$\begin{gathered}\{\phi \circ \psi : \phi \in \Phi\} \supseteq \{\phi \circ \psi : \phi \in \Psi\} = \\ \{\phi \circ \psi^{-1} \circ \psi : \phi \in \Phi\} = \{\phi : \phi \in \Phi\} = \Phi.\end{gathered} \tag{4.10}$$

Equations 4.9 and 4.10 show that $\{\phi \circ \psi : \phi \in \Phi\} \equiv \Phi$ and therefore, the set over which the maximum is taken stays the same, which shows that $f_\theta(\phi(X)) \equiv f_\theta(X)$. □

The statement of the lemma is satisfied for many computer vision tasks: basically all the simple transformations shown in figure 4.1 as well as their compositions form a group. The most common examples of the transformation sets that do not satisfy this property include local shifts (jittering) and local rotations. For example, if one wants to consider only one pixel shifts, then the closure axiom of the group does not hold: one pixel shift applied twice gives two-pixel shift, which is not in a transformation set.

However, one can easily modify the definition of the feature such that it stays transformation-invariant with respect to local transformations:

$$f_\theta(x) = \max_{\phi,\psi\in\Phi} \theta^T \phi(\psi(x)) \tag{4.11}$$

This formulation allows us to relax the closure axiom of the whole set to the closure of only two elements of the set. Features defined by equation 4.11 are invariant to every transformation in Φ (but not in $\{\phi\circ\psi : \phi, \psi \in \Phi\}$) if the transformations set Φ contains all the inverse elements and the identity element. The proof of the last statement stays almost the same, but employs the notion of subgroup instead of the group.

Therefore, if one wants to consider only local transformations and lets Φ to contain, for example, one pixel shift to the left and to the right (together with an identity transformation), then the set over which the maximum should be taken includes shifts by one *or two* pixels.

In the following, we consider the definition 4.8 of a feature to simplify our notation.

Feature learning

Lemma 2 shows that the features formulated in equation 4.8 are transformation-invariant. However, one also needs to establish the procedure of learning the parameters θ of the feature.

Assume that we select two classes $c_1, c_2 \in [1, \dots, K]$ and we want to separate the images of these classes. We propose to find the parameter vector θ by solving the following optimization problem:

$$\theta = \arg\min_\theta E(\theta) = \arg\min_\theta \quad \lambda||\Gamma\theta||_2^2 + \sum_{i:\ y_i=c_1 \text{ or } y_i=c_2} \big(f_\theta(X_i) + [y_i = c_1] - [y_i = c_2]\big)^2 \tag{4.12}$$

Here $f_\theta(x)$ is a feature defined in 4.8 and $[\cdot]$ refers to Iverson brackets, that are equal to 1 if $\cdot$ is true and zero otherwise. Following this notation, $[y_i = c_1] - [y_i = c_2]$ is equal to 1 if $y_i = c_1$ and equal to -1 if

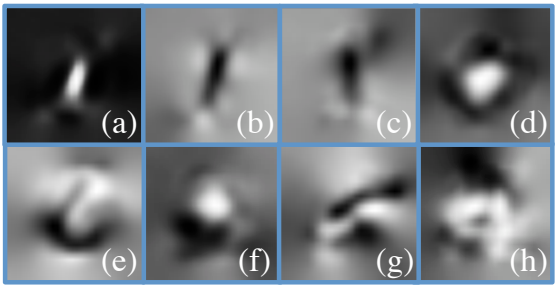

Figure 4.8: Examples of different kernels θ learned with TICJ algorithm applied to a neuronal segmentation data set. One could see that the features are relatively meaningful: features (a)–(c) detect direct lines (this correspond to straight membranes in the data set), (d) denotes the contrast of the center pixels comparing to the surroundings, (e) and (f) detect corners and curvatures (non-straight membranes), (g) and (h) – membrane conjunctions and textures of neuronal tissue (high-frequency features).

$y_i = c_2$. Matrix $\Gamma \in \mathbb{R}^{2w(w-1)\times(w^2+1)}$ is a matrix of a 2D differentiation operator in a vectorized space, that is a Tikhonov regularization matrix. Penalizing the gradient of the kernel enforces the kernel to be smooth. λ is a regularization parameter that controls the trade-off between the goodness of separation and the smoothness of the learned kernel.

Regularization serves two main goals. First of all, it ensures interpretability of the inferred kernels (see figure 4.8). Second, from an optimization point of view, a strictly concave and differentiable regularization term increases the convergence speed of the gradient descent optimization algorithm.

In order to efficiently find the minimum of $E(\theta)$, we also compute the subgradient of the functional 4.12:

$$\frac{dE}{d\theta} = \sum_{\substack{i:\ y_i=c_1 \\ \text{or } y_i=c_2}} 2\Big(f_\theta(X_i) + [y_i = c_1] - [y_i = c_2]\Big)\phi_i(X_i) + 2\lambda\Gamma^T\Gamma\theta$$

where $\phi_i = \arg\max_{\phi\in\Phi} \theta^T\phi(X_i)$ is a transformation that gives the maximum response for an input image X_i.

Based on the formulas 4.12 and 4.13 one can implement an optimization algorithm that finds the optimum value of θ for a given data set $\{X_i, y_i\}$ and for two selected classes c_1 and c_2. It is important to notice that the problem 4.12 is not continuously differentiable because of the maximum in the definition of $f_\theta(X)$, however, it is convex and therefore one is guaranteed to find the global optimum of the problem. In our experiments we selected the L-BFGS optimization subroutine [Noc80] as it always yielded the highest convergence speed for the tasks we consider. The constructive version of the algorithm is sketched in figure 4.9.

One important question is how one selects c_1 and c_2. We propose to take them at random with the probabilities p_c proportional to the presence of the class c in the data set: $p_c \sim |\{i : y_i = c\}|$. This choice assures that we try to separate the largest classes with high probability, but also leaves room for randomization of the algorithm,

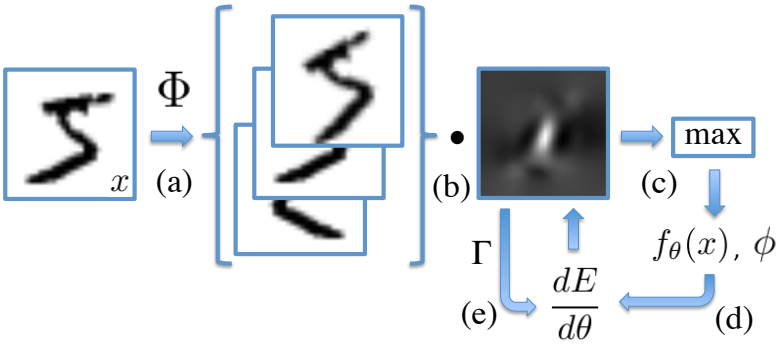

Figure 4.9: Partial visualization of the feature learning process. For all the images x, we compute the transformations $\phi(x), \forall \phi \in \Phi$ (a). Every image after transformation is convolved linearly with the current kernel vector θ (b). That gives the response for every transformation, from which we select then the maximum $f_\theta(x)$ and the corresponding transformation ϕ that gives maximum response (c). These values are then used to compute the functional value and the gradient of the functional value (d) when combined with the regularization term (e). The gradient step $\frac{dE}{d\theta}$ then updates the value of the feature parameters θ.

which is important for ensemble learning discussed in section 4.3.2. There, we also discuss the problem of selecting the regularization parameter λ.

TICT and TICJ: Transformation-Invariant Convolutional Trees and Jungles

Section 4.3.2 shows how to learn the parameters of a transformation-invariant feature that splits the data set into two subsets: one subset consists of the images $X_i : f_\theta(X_i) > 0$, another of images $X_i : f_\theta(X_i) \leq 0$. That means that the feature defines a split predicate on the space of images, and therefore can be used in algorithms such as decision trees.

This section discusses how to learn these features and combine them in an iterative feed-forward manner to build a final image classification algorithm.

Transformation-Invariant Convolutional Trees

Following the idea of decision trees [Qui86], we learn one feature and then split the whole data set into two subsets according to the predicate defined by this feature. To each of the subsets the same idea can be applied recursively until a termination criterion is satisfied. We call this algorithm Transformation-Invariant Convolutional Trees (TICT).

Formally, we start with a root node that accepts the whole data set $\{X_i, y_i : i = 1, \ldots, N\}$ as input, and trains θ to define a root feature. Then we split the training data set into subsets l_1 and l_2: $l_1 = \{i : f_\theta(X_i) > 0\}$, $l_2 = \{i : f_\theta(X_i) \leq 0\}$. For this first layer we define a set of leaves $L = \{l, r\}$.

Then each new layer is built recursively as follows.

- For every leaf $l \in L$ we train new feature parameters θ, but using only a subset of the original data set defined by indices in l.

- Then we split l to $l_1 = \{i : i \in l, f_\theta(X_i) > 0\}$ and $l_2 = \{i : i \in l, f_\theta(X_i) \leq 0\}$.
- If $|l_1| > 0$ and $|l_2| > 0$ (the split is non-trivial), then the new leaves set is defined as $L \cup \{l_{j,1}, l_{j,2}\} \setminus l_j$, otherwise it does not change.
- If $|l_1| = 0$ or $|l_2| = 0$, but $|\{c : \exists i \in l \text{ s.t. } y_i = c\}| > 1$ (this denotes that the leaf l contains objects of at least two different classes), that means that the features are not flexible enough to separate the data set and we increase its flexibility by decreasing the value of λ (for example, multiply it by $\frac{2}{3}$).

We add layers with the above procedure until the maximum number of iterations is reached. If at some iteration step $|\{c : \exists i \in l \text{ s.t. } y_i = c\}| = 1$ for every $l \in L$, then the training data set is perfectly separated and the algorithm terminates.

The classification of new image X with TICT is achieved in exactly the same way as with decision trees: we go from the root node following the splits to the leaves, and then return the majority class of the objects in this leaf. The proposed algorithm is similar to oblique decision trees [HKS93]. However, unlike oblique decision trees, the features in our case do not form a linear combination of all the pixel intensities, and therefore TICT does not belong to this category. We use TICT for comparison, but the final algorithm uses a modification of it inspired by Decision Jungles [SSK+13b].

Transformation-Invariant Convolutional Jungles

There are two main problems with the previous approach:

- first, it easily overfits the data if the number of splits is large and the sizes of the leaves are small;
- second, the size of TICT grows very fast, causing major efficiency and memory issues (if we add 20 layers, in worst case we need to train 2^{21} features).

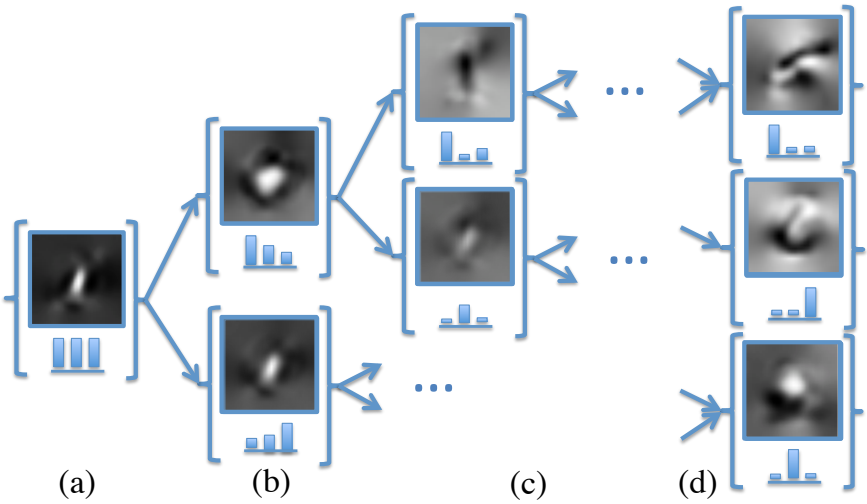

Figure 4.10: A visualization of TICJ training process. Each node is represented with feature parameters θ and a histogram h of input object classes (for simplicity we consider three classes here). (a) shows the root node, for which the whole data set is an input. Using the learned feature f_θ – the data set is split in two subsets to serve as input for two other nodes (b). The algorithm proceeds by splitting the data set until the maximum number width M is achieved (c). Then some of the data subsets can be joined together with a histogram clustering technique (d).

In order to overcome both of these issues, we propose TICJ algorithm based on the modification of the Decision Jungles algorithm. The idea of TICJ is very simple: after adding one layer, we perform the clustering of leaves in L and join similar leaves together where the similarity of leaves is measured as the similarity of the histograms of the classes present in a leaf. We merge leaves only if the leaves set size $|L|$ is greater than some constant M. We also generate a new layer in a feed-forward manner, so after joining the leafs, we do not retrain the features. That allows us to spend up to two times less training time, and produces very similar results to a two-step procedure in our experiments. The scheme of the algorithm is sketched in figure 4.10.

The second extension of the original decision jungles paper is how we perform clustering. The paper [SSK+13b] suggests two clustering technique: a global and a randomized greedy, and claims that a global clustering technique performs worse. We experienced very similar behaviour and discovered a possible reason behind that: very often global clustering joins the leafs that were separated just before with the feature learned. For example, quite a common case is that θ learned in one layer splits l into l_1 and l_2, and then the clustering algorithm groups l_1 and l_2 again together. That means that in the next layer θ will be trained again with the same data, and with high probability will yield the same results, getting the algorithm stuck in this loop.

To overcome this issue, we propose to forbid the clusters consisting of two leaves that were just split. That can be easily implemented by just setting the distance of these leaves to be infinite before executing the clustering algorithm. Formally we define the distance between leafs $D(l_j, l_i)$ as either $+\infty$ if l_j and l_i originate from one set l_k, or $D(l_j, l_i) = \frac{1}{2}\left(D_{KL}(h_j||h_i) + D_{KL}(h_i||h_j)\right)$ otherwise. Here $D_{KL}(\cdot||\cdot)$ stands for Kullback-Leibler divergence [Kai67], and $h_j = [|\{i \in l_j : y_i = 1\}|, \ldots, |\{i \in l_j : y_i = K\}|]$ – a histogram representation of the leaf classes present in a leaf l_j. Then we apply agglomerative clustering [GGG06] with the defined metric to get clusters $1, \ldots, M$ and redefine the leaf set to be $L = \{\cup_{l \in \text{ cluster } 1} l, \ldots, \cup_{l \in \text{ cluster } M} l\}$.

One can also construct an ensemble of trained jungles. Such an ensemble may slightly increases the recognition performance in some cases and reduces the variance of the resulting classifier. This improvement is caused by the randomness involved into constructing every instance of TICJ: classes for separation are taken at random. One can also use only a subset of objects or a subset of transformations for TICJ training to further diversify the trained models and benefit more from averaging their outputs.

Algorithm parameters discussions

In this section we demonstrate how to better understand the parameters and how to set them wisely without harming the performance and the efficiency of the algorithm. There are three main parameters in the proposed algorithm: (i) a set of transformations Φ, (ii) the regularization parameter λ considered during feature learning, and (iii) the maximum TICJ width M.

The maximum number of iterations is also a hyperparameter, but it is less important as it does not need to be specified in advance: if the performance on the validation data set is still improving, one can always add more layers.

Φ – a set of transformations – depends on the task being solved and almost always can be selected in advance, as discussed in section 4.3. We also want to note that Φ partly serves the regularization purpose. When the size of the set Φ increases, then the learned feature f_θ is expected to have less degrees of freedom. Therefore one should be careful when selecting a large set Φ as it can prevent flexible features from being learned.

This model selection choice is, however, not an issue when we determine the regularization parameter λ. One can start with a large value of λ to learn very smooth kernels corresponding to low-frequency features. As we discuss in section 4.3.2, if learning θ gives only trivial splits, we decrease the value of λ (usually just multiply it by $\frac{2}{3}$) and start to discover also high-frequency features. That gradually increases the complexity of the features as we go down the layers

hierarchy.

The maximum width of the jungle M is the only remaining parameter that defines the topology of TICJ. This parameter also significantly contributes to the control of the bias-variance trade-off: if it is small, it prevents overfitting, but the algorithm usually appears to be less flexible. If it is chosen too large then TICJ adapts to fluctuations. We propose to use the following heuristic:

- start with a small value of M (we usually take $M = 3K$, where K is the number of classes),
- train the algorithm by adding more layers and observe the validation error;
- if the validation error stops decreasing, enlarge M without retraining the whole TICJ and continue adding layers (just the new layers would be wider).

This process can be repeated until the algorithm starts overfitting, which is usually indicated by the increase in a validation set error.

4.3.3 Experiments

In this section we present the experimental results on two publicly available computer vision data sets: (i) the Yale face recognition data set [BHK97] and (ii) the Neuronal structures segmentation data set [CSP+10]. Both data sets include large intra class variabilities (see examples in figure 4.11), but also contain some transformation-invariances, which we exploit with the proposed algorithm.

In the face recognition benchmark we achieve slightly better results, than the state of the art algorithms: 0.3% increase in accuracy (here we consider only the methods that use no additional training data).

In the structure segmentation benchmark, we exactly match the performance of the current state of the art algorithm, which are Convolutional Neural Networks [CGGS12], but we obtain these results orders of magnitudes faster.

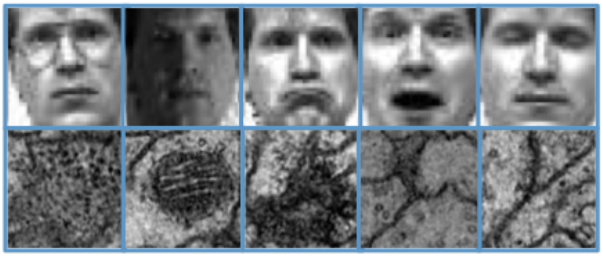

Figure 4.11: Example cropped images from Yale face recognition data set (top row) and example patches from Neuronal structures segmentation data set (bottom row). The Yale data set includes large variations in pose, facial expression, illumination and sometimes includes obstacles (glasses). Patches of neuronal tissue sometimes clearly indicate membranes (the last two images), but in many cases the images display very unclear and blurred structures that are hard to detect even for a trained human expert.

Face recognition

The original Yale face recognition data set contains 165 grayscale images of 15 individuals (and therefore has $K = 15$ classes). There are 11 images per subject, one per different facial expression (normal, happy, sad, sleepy, surprised, and wink) or configuration (left-light, center-light, right-light, with/without glasses).

We follow the most commonly adopted experimental setup [CHH+07, HVD07, SC08] and average the results over 50 random splits into training and test sets. Splits are performed independently for all images of a particular person. For training we select five images per person, and use the other six for testing. We also use the cropped version of the data set [CHH+07] since most publications with competing methods follow this protocol.

We run both TICT and TICJ with the set Φ of transformations that includes small shifts (up to two pixels in each side) and illumi-

Figure 4.12: An example implementation of a transformation that changes illumination. The original image is per-pixel divided by a blurred version of itself. This makes some originally dark regions a little lighter.

nation changes. We implement illumination changes simply through dividing the original image by the very blurred one (see figure 4.12 for example). For blurring we use Gaussian kernels with width 8 and 16. Other parameters are selected as described in section 4.3.2.

As baselines we select two state of the art methods that achieve the best results for this experimental setup: spatially smooth subspace learning [CHH+07] and orthogonal rank one tensor projections [HVD07]. The results are presented in table below. The table also includes neural networks on pretrained features [SC08] that performs better than the proposed TICJ. However, this method uses additional data for the feature learning process, and therefore a comparison is questionable. Apart from this method, TICJ achieves better results than state of the art methods using no additional data. TICT overfits the data and performs significantly worse.

Method	**Error (%)**
Cai et al. [CHH+07]	18.3
Cai et al. [CHH+07] (updated)	14.7
Hua et al. [HVD07]	13.2
Shan et al. [SC08]	8.2
TICT (ours)	18.4
TICJ ($\Phi = \varnothing$) (ours)	16
TICJ (ours)	12.9

Neuronal structure segmentation

We consider neuronal structure segmentation data set as an example of a medical imaging data set with intrinsic transformation-invariance. The task is to segment inner are of neurons from the membranes separating different neurons. From the neurological experts we know, that membrane appearance does not depend on the orientation of the membrane and, therefore, we can safely include 360° rotations in the set of transformations Φ. We sample rotations at every 15 degrees, resulting in 24 transformations considered. Further details of the data set are described in section 2.2.2.

To address segmentation tasks, we employ the commonly used patch classification strategy: instead of X_i being an image in classification task, we consider X_i to be a patch around pixel i, and y_i to be the corresponding pixel class (segment index). Then, the algorithm should return the class estimate for every pixel based on the appearance of the surrounding $w \times w$ pixel area. The patch size w is an application dependent parameter that we select by cross-validation.

We perform training on 50000 pixel patches X selected at random from the training images together with the labels of the corresponding pixel y. We select the patch size to be 31×31 pixels ($w = 31$).

As baselines, we select the methods that won the first and the second place in the challenge [cha]. The first method is an ensemble of convolutional neural networks (CNN) [CGGS12]. The second method is a random forest per-pixel classifier with a huge number of features and cross-image priors (RF) (introduced in chapter 2). We also compare with the previously proposed CDT, as they outperform RF.

Method	1–F-score (%)	Training time
RF (chapter 2)	7.9	1 hour
CDT (section 4.2)	6.8	8 hours (CPU)
CNN [CGGS12]	6.0	7 days (GPU)
TICT (ours)	6.7	2.5 hours (CPU)
TICJ (ours)	6.0	3 hours (CPU)

The results of the experiment are presented in table above: TICJ matches the state of the art results of Convolutional Neural Networks and consistently outperforms other methods. It is also very important to notice the highly significant speedup during training, where the training time is orders of magnitude smaller for TICJ then for CNN. CNN are reported to train for about one week on a GPU cluster, and the estimated time is one year on a single CPU. TICJ, on the contrary, can be trained in one CPU within three hours.

4.4 Transformation-Invariant Pooling

As we show in a previous section, TICJ effectively exploits transformation-invariance and, thereby, achieves results similar to Convolutional Neural Networks for some detection problems. However, its greedy manner can prevent it from performing as a state of the art algorithm for the majority of problems. While focusing on training speed can be very important for special problems, it can be unnecessary for others.

In this section we show how to apply the ideas of TICJ to Convolutional Neural Networks framework. The combination of the two techniques results in better accuracy and generalization capacity, while sacrificing some of the training speed of TICJ.

4.4.1 Transformation-invariance in deep learning

Recent advances in deep learning produced impressive results for various applications of machine learning and computer vision in different fields. These advances are largely attributed to the expressiveness of deep neural networks with many parameters, that are effectively able to approximate any decision function in the data space [LLPS93].

While this is true for all the neural network architectures with many layers and with sufficient number of parameters, the most impressive results are being achieved in the fields where deep architectures heavily rely on internal structure of the input data, such as speech recognition, natural language processing and image recog-

nition [LB95]. For example, convolutional neural networks [KSH12] learn kernels to be applied on images or signals reflecting the spatial or temporal dependencies between the neighbouring pixels or moments in time. This structural information serves for internal regularization through weight sharing in convolutional layers [LJB+95]. When combined with the expressiveness of multilayer neural networks, it enables to learn very rich feature representation of input data with little to no preprocessing.

Incorporating structural information permits to work with the inner dependencies in the representation of the data, but only few works have addressed the possible use of other structural prior information known about the data. For example, many data sets in computer vision contain some nuisance variations, such as rotations, shifts, scale changes, illumination variations, etc. These variations are in many cases known in advance from experts collecting the data and one can significantly improve the performance when being considered during training.

The effect is even more explicit when dealing with domain-specific problems. For example, in many medical imaging data sets, the rotation can be irrelevant due to the symmetric nature of some biological structures. At the same time, the scale is fixed during the imaging process and should not be considered as a nuisance factor. Moreover scale-invariance can even harm the performance if object size is at least somehow informative, for example, in case of classifying healthy cells from cancer cells [SFO+10]. We describe one example in detail in section 4.4.4.

The state of the art approach to deal with these variations and the most popular one in deep learning is *data augmentation* [VDM01] – a powerful technique that transforms the data point according to some predefined rules and uses it as a separate training sample during the learning procedure. The most common transformations being used in general computer vision are rotations, scale changes and random crops. This approach works especially good when applied with deep learning algorithms, because the models in deep learning are

extremely flexible and are able to learn the representation for the original sample and for the transformed ones and therefore are able to generalize also to the variations of the unseen data points [VDM01]. This approach, however, has some limitations listed below.

- The algorithm still needs to learn feature representations separately for different variations of the original data. For example, if a neural network learns edge-detecting features [CGGS12] under rotation-invariance setting, it still needs to learn separately vertical and horizontal edge detectors as separate paths of neuron activations.

- Some transformations of the data can actually result in the algorithm learning from noise samples or wrong labels. For example, random crops applied to the input image can capture only a non-representative part of the object in the image, or can fully cut the object out, in which case the algorithm can either overfit to the surrounding or learn from a completely useless representation.

- The more variations are considered in the data, the more flexible the model needs to be to capture all the variations in the data. This results in more data required, longer training times, less control over the model complexity and larger potential for overfitting.

On the other hand we use the approach inspired by the max-pooling operator [LB95] and by multiple-instance learning [WYHY15] to formulate convolutional neural network features to be transformation-invariant. We take the path of neuron activations in the network and feed it, in a similar manner to augmentation, with the original image and its transformed versions (input instances). But instead of treating all the instances as independent samples, we accumulate all of the responses and take the maximum of them (TI-POOLING operator). Because of the maximum, the response is independent from the variations and results in transformation-invariant features that are

further propagated through the network. At the same time this allows for more efficient data usage as it learns from only one instance, that already gives maximum response. We call these instances "canonical" and describe in more details in 4.4.3.

This topology is implemented as parallel siamese network [BBB+93] layers with shared weights and with inputs corresponding to different transformations, described in detail in section 4.4.3 and sketched in figure 4.13. We provide theoretical justification on why features learned in this way are transformation-invariant and elaborate on further properties of TI-POOLING in section 4.4.3.

Using TI-POOLING permits to learn smaller number of network parameters than when using data augmentation, and lacks a drawback of some data-points missing relevant information after the applied transformation: it only uses the most representative instance for learning and omits the augmentations that are not useful. We review other approaches dealing with nuisance data variations in section 4.4.2.

We evaluate our approach and demonstrate it's properties on three different data sets. The first two are variations of the original MNIST data set [LBBH98], where we significantly outperform the state of the art approaches (for the first variation) or match the current state of the art performance with significantly faster training (on the second variation). The third data set is a real-world biomedical segmentation data set with explicit rotation-invariance. On this benchmark we show that incorporating TI-POOLING operator increases the performance over the baselines with similar number of parameters, and also demonstrate the property of TI-POOLING to find canonical transformations of the input for more efficient data usage.

4.4.2 Related work

There exist many transformation-invariant features, such as SIFT [Low99], line filter transform [SB07] and rotational bag of visual words [YN10]. We discuss most of these features in section 4.3.1, and focus in this section mostly on comparison with previously introduced

TICJ, transformation-invariant deep learning architectures, and multiple instance learning.

Transformation-Invariant Convolutional Jungle (TICJ), introduced in section 4.3, while being fast, have the following limitations: (i) greediness in the feature learning process (only one kernel is learned at a time) and (ii) relatively low expressiveness of the combining machine learning algorithm. The algorithms that are usually able to overcome both of these limitations are neural networks.

Deep neural networks

Convolutional deep neural networks [KSH12] are known to learn very expressive features in an adaptive manner depending on the task. Moreover in many cases they resemble some transformation-invariant properties, as discussed in section 4.3.1. Here we will focus specifically on two relevant approaches: such as multi-column deep neural networks [CMS12] and spatial transformer networks [JSZK15].

The idea behind multi-column networks is to train different models with the same topology but using different data sets: the original data set, and the transformed data sets (one separate model is trained for every transformation considered). Then an average of the outputs of individual models is taken to form the final solution.

Spatial transformer networks (STN) follow a completely different idea of looking for a canonical appearance of the input data point. They introduce a new layer to the topology of the network, that transforms the input according to the rules of parametrized class of transformations. The key feature of this approach is that it learns the transformation parameters from the data itself without any additional supervision, except of a defined class of transformations.

The TI-POOLING approach in many ways has very similar properties to STN. As we demonstrate in section 4.4.3, our method also finds a canonical position of the input image. But instead of defining a *parametrized class* of transformations, we define a general *set* of transformations to be considered, not limited to any parametrized functions. In section 4.4.4 we show that we achieve similar to STN

results on a benchmark introduced by its authors [JSZK15], but with simpler model and with shorter training time.

Multiple instance learning

Multi-column networks with model averaging described above fall into a category of more general techniques called "multiple instance learning" (MIL) [WYHY15]. The area of applications of MIL is very broad, and it can also be applied to train the algorithms invariant to some variations defined as a set of transformations Φ.

Assume that we are given an algorithm $\mathcal{A}$ with some input x and scalar (for simplicity) output $\mathcal{A}(x)$. Then the multiple-instance learning approach suggests that the algorithm $\mathcal{B}(x)$ will be in many cases transformation-invariant if defined as

$$\mathcal{B}(x) = \max_{\phi \in \Phi} \mathcal{A}(\phi(x))$$

Instead of a maximum, many different operators can be used (such as averaging), but maximum proves to work best in most applications, so we also focus on it in this work.

While MIL algorithm as a whole can indeed be transformation-invariant, individual features are not required to be transformation-invariant. In a way, MIL tries to assemble a transformation-invariant algorithm from arbitrary features, which can sometimes limit both performance and accuracy. The main difference between our approach and MIL is that we propose to learn individual features to be transformation-invariant, and not the algorithm as a whole. Each of the features can then be learned in a way that is most optimal specifically for this feature, allowing different features to rely on different canonical instances and make the most of feature inter-dependencies. We describe this relation in more details in section 4.4.3. Overall our method significantly outperforms the standard MIL models as we show further in section 4.4.4.

Other approaches that are based on the ideas similar to the one

presented in this paper are rolling feature maps [1] and multi-view networks [SMKLM15]. The former explores a pooling over a set of transformations, but does not guarantee the transformation-invariance of the features learned. And the latter solves a problem of view invariance, not invariance to an expert-defined set of transformations.

4.4.3 Method description

Convolutional neural networks notation

Convolutional neural networks are usually represented as a sequence of convolutional and subsampling layers with one or more fully-connected layers before the outputs. In this section we for simplicity assume that the input image is two-dimensional (i.e. incorporate no colour channels), but the approach generalizes also for colored images. We also omit the explicit notation for activation functions, assuming activations to be incorporated in the specific form of an operator O defined below.

Assume that each neuron performs an operation on the input x, that we will refer to as an operator $O(x, \theta)$. It can be either a convolution operator, in which case θ is a vectorized representation of a convolutional kernel. Or it can be a subsampling operator, which is usually non-parametric, and has no parameters θ. The size of the output matrix $O(x, \theta)$ in each dimension is smaller than the size of x by the size of the kernel in case of a convolution operator, or two times smaller than the input x in case of a subsampling operator.

We refer to these operators applied in layer $l \in \{1, \ldots, L\}$ using superscript l on the parameters θ and we refer to a specific index of the operator within a layer as a subscript. For example, convolutional operations applied in the first layer of the network can be referred as $O(x, \theta_1^1), \ldots, O(x, \theta_{n_1}^1)$, where n_1 is the number of neurons in layer 1 (we define all the constants in table 4.1). The output of the neuron i in layer 2 is recursively defined as $O\big([O(x, \theta_1^1), \ldots, O(x, \theta_{n_1}^1)], \theta_i^2\big)$, $i = 1, \ldots, n_2$.

[1] http://benanne.github.io/2015/03/17/plankton.html

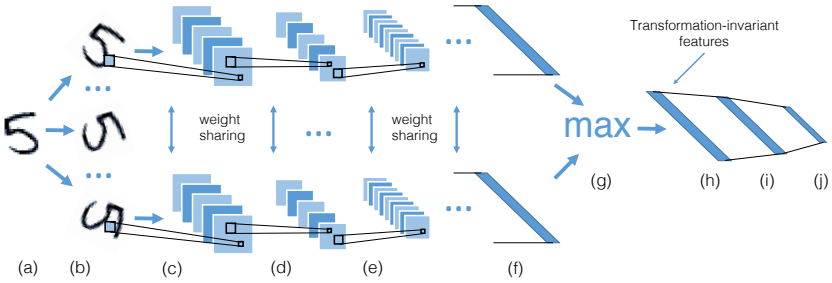

Figure 4.13: Network topology and pipeline description. First, input image x (a) is transformed according to the considered set of transformations Φ to obtain a set of new image instances $\phi(x), \phi \in \Phi$ (b). For every transformed image, a parallel instance of partial siamese network is initialized, consisting only of convolutional and subsampling layers (two copies are shown in the top and in the bottom of the figure). Every instance is then passed through a sequence of convolutional (c, e) and subsampling layers (d), until the vector of scalars is not achieved (e). This vector of scalars is composed of image features $f_k(\phi(x))$ learned by the network. Then TI-POOLING (element-wise maximum) (g) is applied on the feature vectors to obtain a vector of transformation-invariant features $g_k(x)$ (h). This vector then serves as an input to a fully-connected layer (i), possibly with dropout, and further propagates to the network output (j). Because of the weight-sharing between parallel siamese layers, the actual model requires the same amount of memory as just one convolutional neural network. TI-POOLING ensures that the actual training of each features parameters is performed on the most representative instance $\phi(x)$.

To simplify the notation and omit all the nested formulas, we will assume that the input to layer l is known, replacing this input with a $\cdot$ notation. Using this notation, the output to the neuron i in layer l is constructed as follows: $O(\cdot, \theta_i^l)$.

We refer to feature f_k of the input image x as an output of a neuron k in a layer that contains only scalar values, i.e. layer l such that $O(\cdot, \theta_i^l) \in \mathbb{R}^{1\times 1}$.

$$f_k(x) = O\left(\cdot, \theta_k^l\right), \text{ where } l : O(\cdot, \theta_i^l) \in \mathbb{R}^{1\times 1}. \tag{4.13}$$

On top of these features $f_k(x)$, fully-connected layers are usually stacked with some intermediate activation functions, and possibly with dropout masks [HSK+12] during learning. These are not directly relevant for this paper and therefore not described in detail.

Network topology

Features $f_k(x)$, introduced before, are very powerful when all the parameters θ are properly trained. They, however, lack a very important property of incorporating any prior information, such as invariance to some known nuisance variations in the data. We fix this property with a relatively easy trick, inspired by multiple-instance learning (MIL) and max-pooling operator.

Assume that, given a set of possible transformations Φ, we want to construct new features $g_k(x)$ in such a way that their output is independent from the known in advance nuisance variations of the image x. We propose to formulate these features in the following manner:

$$g_k(x) = \max_{\phi\in\Phi} f_k(\phi(x)) \tag{4.14}$$

We refer to this max-pooling over transformations as to transformation-invariant pooling or TI-POOLING. Because of the maximum being applied, every learned feature becomes less dependent on the variations being considered. Moreover, for some sets Φ we achieve full transformation-invariance, as we theoretically show in section 4.4.3.

As mentioned before and as we show in section 4.4.3, TI-POOLING ensures that we use the most optimal instance $\phi(x)$ for learning, and comparing to MIL models we allow every feature k to find its own optimal transformation ϕ of the input x: $\phi = \arg\max_{\phi \in \Phi} f_k(\phi(x))$.

The topology of the proposed model is also briefly sketched and described in figure 4.13.

Back-propagation. Let $\nabla f_k(x)$ be the gradient of the feature $f_k(x)$ defined in equations 4.13 with respect to the outputs $O(\cdot, \theta_j^{l-1})$ of the previous layer. This gradient is standard for convolutional neural networks and we do not discuss in detail how to compute it. From this gradient we can easily formulate the gradient $\frac{dg_k(x)}{df_k(x)}$ of the transformation-invariant feature $g_k(x)$ in the following manner:

$$\frac{dg_k(x)}{df_k(x)} = \nabla f_k(\phi(x)), \text{where } \phi = \arg\max_{\phi \in \Phi} f_k(\phi(x))$$

The gradient of the neurons of the following fully-connected layer with respect to the output of $g_k(x)$ stays exactly the same as for conventional network topology. Therefore, we have all the building blocks for a back-propagation parameter optimization [CR95], which concludes the description of TI-POOLING and of the proposed topology.

Theory and properties

Theoretical transformation-invariance. Lemma 3 is an adaptation of the lemma 2 from the previous section, formulates the conditions on the set Φ for which the features formulated in equation 4.14 are indeed transformation-invariant, i.e. give exactly the same output for both the original image x and every considered transformation $\phi(x), \phi \in \Phi$.

Lemma 3. *Let the function $g_k(\cdot)$ be defined as a maximum over transformations $\phi \in \Phi$ of some other function $f_k(\cdot)$. This function $g_k(\cdot)$ is transformation-invariant if the set Φ of all possible transformations forms a group, i.e. satisfies the axioms of closure, associativity, invertibility and identity.*

The proof essentially repeats the proof of lemma 2.

The statement of the lemma is satisfied for many computer vision tasks: simple transformations, such as rotations or non-linear distortions, as well as their compositions form a group. One common example that does not satisfy this property is local shifts (jittering). For example, if one wants to consider only one pixel shifts, then the closure axiom of the group does not hold: one pixel shift applied twice gives two-pixel shift, which is not in a transformation set.

Canonical position identification. From a practical point of view, however, the algorithm achieves approximate transformation-invariance even for local transformations. If the set Φ does not form a group, we often observe that the algorithm tries to find a canonical appearance of the image, and then maps a new transformed image to one of the canonical modes. This standardization allows us to preserve transformation-invariance in most practical cases with no limitations on Φ. Figure 4.14 shows some examples of neuronal structures oriented in the same manner to a canonical orientation for one of the features.

The canonical samples are useful for most problems as they permit for better use of input images. For example, learning discriminative features for every orientation of the image is of course possible with large and deep enough neural network. But assume that now features need to be learned only for canonical orientation (e.g. for membranes oriented in all the same direction).

First, for this, much simpler problem, smaller models can be used. Second, the algorithm sees many more examples of canonical vertical edges and therefore can better generalize from them. This brings the next important property of the algorithm.

Improved performance and convergence. Because of more representative examples being used for network training, we observe better performance and convergence rate, when compared with simple data augmentation. Figure 4.15 shows that the larger the transformation set Φ – the better usually the results achieved. This is most probably due to the fact that fewer canonical positions needs to be

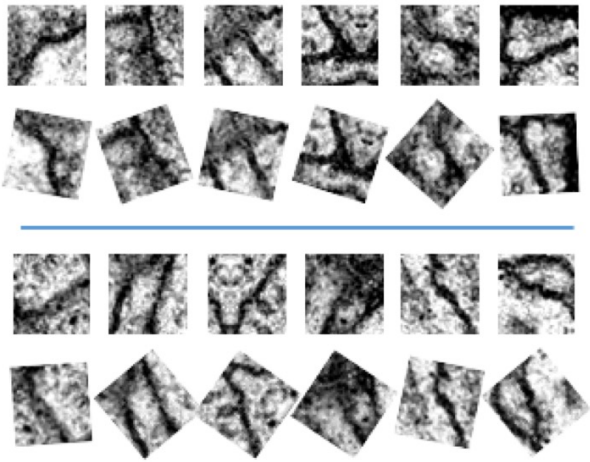

Figure 4.14: First and third rows show the input patches from neuronal segmentation data set. For this data set we consider Φ to be a set of rotations. We then apply a learned model to these patches x and record the angle at which the maximum is achieved for specific feature $g_k(x)$. Then we show the same patches rotated by this angle as shown in rows two and four. One could notice that in most cases the membranes (slightly darker elongated structures) are oriented in approximately the same direction. This means that the algorithm considers this orientation to be canonical for this specific feature and rotates new images to appear similarly.

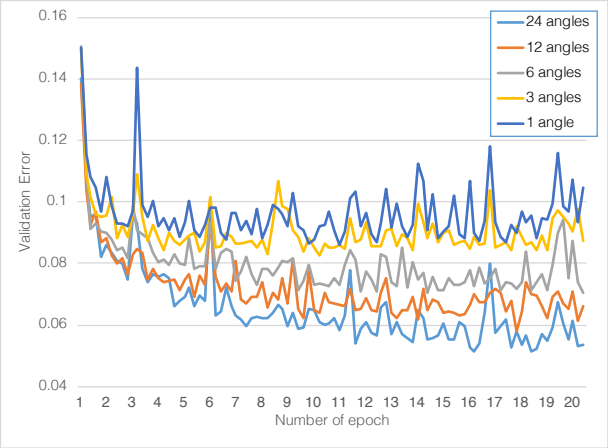

Figure 4.15: Validation error plot for the neuronal segmentation data set. Depending on how many angles we sample to form a transformation set Φ (from one, which is equivalent to data augmentation, up to 24) – the results improve significantly.

handled by the learning algorithm.

What TI-POOLING is doing to achieve that can be formulated as an exhaustive search over the transformed instances for an instance better corresponding to the current response of the feature. Then only this instance is used to even better improve the performance of the feature. On the other hand, we do not limit all the features to use the same canonical appearance, allowing features to better explore inter-dependencies between the outputs of network layers. We elaborate more on the results in section 4.4.4.

Any type of transformations. Another property of the technique, that is worth mentioning, is that it can work with a set of almost any arbitrary transformations. Many works, such as spatial transformer networks [JSZK15], focus on only limited class of transformations. Those classes can be very rich, e.g. include all the possible affine transformations or projections. But still, they need to be differentiable with respect to some defined parameters of the transformation, and, depending on the problem at hand, this can be not enough. TI-POOLING, on the contrary, does not rely on differentiability or on any properties of bijective functions or even on the parametrization itself. Examples of common transformations that can be used with our method, and not with [JSZK15] are reflections, most morphological operations and non-linear distortions.

Implementation details

We use Torch7 framework for model formulation and training [CKF11]. The easiest way to formulate a proposed model is to use parallel network notation with shared weights as described in figure 4.13. The whole model definition requires just few additional lines of code. An example in pseudo-lua code is provided below. Here `nPhi` is a size of the set Φ.

```
-- define first siamese layers
siamese = Sequential()
...
```

```
-- clone and share weights
parallel = Parallel(1, 3)
for phi = 1, nPhi do
   clone = siamese:clone()
   parallel:add(clone)
   clone:share(siamese, 'weight','bias',
               'gradWeight','gradBias')
end
-- formulate a full model
model = Sequential()
model:add(parallel)
model:add(SpatialMaxPooling(nPhi,1,1))
-- add fully-connected layers,
-- dropout and output layer
```

The only other modification is to the data: we increase the dimension of the input data tensor by one and stack input instances $\phi(x), \phi \in \Phi$ across the new dimension.

Computational complexity. It may seem like an exhaustive search in the space of possible transformations Φ significantly increases computational complexity of the pipeline. Indeed, instead of processing one image at a time, we forward-pass $|\Phi|$ images through almost the whole network. We can speed it up by sampling from the space of transformations, but in practice, even searching the full space appears to be more efficient than just data augmentation, because of the following reasons:

- Only partial forward pass is done multiple times for the same image, forward-pass through fully-connected layers and back-propagation are exactly the same computationally as for a standard convolutional neural network with the same number of parameters.

- Comparing to the data augmentation approach, we make use of every image and it's augmented versions in one pass. Standard convolutional neural network instead makes one pass for every

augmented sample, which in the end results in the number of passes equal to the number of augmentated samples to process one image. Due to the previous point, we actually perform it more than two times faster than before.

- Because we make use of the canonical appearance of the image, the proposed pipeline actually trains more efficiently than the standard neural network, and usually requires smaller number of overall parameters.

4.4.4 Experiments

In this section we present the experimental results on three computer vision data sets. The first two data sets are different variations of MNIST data set [LBBH98] designed to test artificially-introduced variations in the data. The third one is a neuronal structures segmentation data set [CSP+10], that demonstrates a real-world example of rotation invariance.

Rotated MNIST

Original MNIST data set [LBBH98] is a very typical toy data set to check the performance of new computer vision algorithms modifications. Two variations of MNIST exist to test the performance of different algorithms that are designed to be invariant to some specific variations, such as rotations.

For both the data sets we use the same topology, but slightly different sets Φ. The topology is described in table 4.1. We perform the training using tuning-free convergent adadelta algorithm [Zei12] with the batch size equal to 128 and dropout [HSK+12] for fully-connected layers.

mnist-rot-12k data set. The most commonly used variation of MNIST that is used for validating rotation-invariant algorithms is mnist-rot [LEC+07]. It consists of images from the original MNIST, rotated by a random angle from 0 to 2π (full circle). This data set

Layer	Parameters & channel size
input	size: 32x32
convolution	kernel: 3x3, channel: 40
relu	
max pooling	kernel: 2x2, stride: 2
convolution	kernel: 3x3, channel: 80
relu	
max pooling	kernel: 2x2, stride: 2
convolution	kernel: 3x3, channel: 160
relu	
max pooling	kernel: 2x2, stride: 2
linear	channel: 5120
relu	
TI-POOLING	transformations: Φ
dropout	rate: 0.5
linear	channel: 10
softmax	

Table 4.1: The topology of the network in the experiments.

Method	Error, %
ScatNet-2 [BM13]	7.48
PCANet-2 [CJG+15]	7.37
TIRBM [SL12]	4.2
TI-POOLING (ours)	**2.2**

Table 4.2: Results on mnist-rot-12k data set.

contains 12000 training images, which is significantly smaller, than in the original data set, and 50000 test samples.

For this data set we include a TI POOLING step over Φ containing 24 rotations sampled uniformly from 0 to 2π.

We train this network on a single GPU for 1200 epochs and compare the achieved test error with the best results published for this data set. The best approach by [SL12] employs restricted boltzmann machines and achieves 4.2% error, while we achieve 2.2% – the results almost two times better in terms of classification error. The final errors for the proposed and the state of the art results are present in the table 4.2. It can be seen that using TI-POOLING indeed leads to significant improvements with no significant effort of optimising topology and just by better exploiting the variations in the data.

Half-rotated MNIST data set. The second data set we consider is another rotational variation of MNIST data set, but with much more training images. This data set was introduced in [JSZK15]. There are two reasons why the authors decided to advance further from the original mnist-rot-12k. First, mnist-rot-12k is very small in size (five times less than training set in MNIST data set). And second, it has somewhat artificial limitation of images being rotated full circle. So they proposed to take full MNIST data set, use random angle in the range $[-\frac{\pi}{2}, \frac{\pi}{2}]$ (*half* the circle) and use the input images rotated by this angle as training samples. This data generation process makes the problem a little easier, but closer to real-world scenarios.

As discussed in section 4.4.2, the authors of spatial transformer

Method	Error, %
FCN	2.1
CNN	1.2
STN (general)	0.8
STN (affine)	0.7
TI-POOLING (ours)	0.8

Table 4.3: Results on half-rotated MNIST data set.

networks [JSZK15] propose an elegant way of optimising the transformation of the image while learning also the canonical orientation. Here we show that for some classes of transformations, we achieve comparable results with simpler model and shorter training time.

For this problem formulate a set of transformations Φ as a set of angles sampled uniformly from half a circle, to match the data set formulation, overall 13 angles. With this relatively simple model, we converge to the results of 0.8% error within 360 epochs, while STN was trained for 1280 epochs. Moreover, using TI POOLING does not require grid sampling and therefore each individual iteration is faster. With this we still match the performance of the most general STN model defined for a class of projection transformations. For more narrow class of transformations selected manually (affine transformations), our results are slightly worse (by 0.1%). However, we did not optimise with respect to the transformation classes, and therefore the comparison is not fully fair in this case. Table 4.3 shows further comparison with STN and other related baselines on this data set. Baseline fully-connected (FCN) and standard convolutional (CNN) neural networks are defined in [JSZK15] and tuned to have approximately the same number of parameters as the baseline STN.

Neuronal structures segmentation

From the neurological experts we know, that membrane appearance does not depend on the orientation of the membrane, and therefore

Method	Error, %
MIL over CNN [WYHY15]	8.9
CNN with augmentation [KSH12]	8.1
TI-POOLING - dropout	**7.4**
TI-POOLING + dropout	**7.0**

Table 4.4: Results on neuronal segmentation data set.

we can safely include $[0, 2\pi]$ rotations in the set of transformations Φ. We sample rotations every 15 degrees, resulting in 24 transformations considered.

Because this is a segmentation task, we extract patches around a pixel and classify those patches (here label of the patch is the label of the central pixel of the patch). We perform training on all the available pixel patches (balanced between classes). The patch is decided to be square and has the size of 46 pixel, but after the rotation we crop the patch, so the actual input to the network is a 32×32 patch. Some examples of the patches are present in figure 4.14.

For every algorithm we run for this data set, we select the same network topology, in order to better evaluate the improvement of the proposed TI-POOLING operator for rotation-invariant feature learning without incorporating any other effects such as model size. As our baselines, we select the following two algorithms, that are closely related to the proposed technique as discussed in sections 4.4.1 and 4.4.2:

- standard convolutional neural network with data augmentation, that is able ideally to learn features expressive enough to handle rotations in the data;
- multiple instance learning of convolutional neural networks, that is able to learn a transformation-invariant algorithm for a given set of transformations, but not the features.

For all the underlying networks we select the same topology as

described in table 4.1, except of TI-POOLING and the number of outputs (two classes for this data set). We also report the results with and without dropout, as discussed later.

Table 4.4 shows the pixel error achieved by all the algorithms after 16 epochs. For standard convolutional neural network with augmentation we record the results after $16 * 24 = 384$ epochs, so that the number of images "seen" by the algorithm is the same as in other algorithms (because for the proposed approach and for the MIL modification, we take the maximum over all the 24 rotations in one iteration). We also run MIL modification with no dropout, and compare the results with the version of our algorithm trained with no dropout. For both baselines we see the significant improvement for the same topology. From this we can conclude that the proposed TI-POOLING is indeed very helpful for real-world problems with nuisance variations.

4.5 Contributions

Invariance to different types of transformations is required in many domains of machine learning and computer vision. The prior knowledge about nuisance transformations that are reflected in visual data sets can be incorporated into a learning process to achieve better accuracy and higher generalization capacity.

In this chapter we propose three novel methods, applicable to a wide range of computer vision problems. We start with general methods for feature learning and segmentation, then building on top of them method that is able to include the information on transformation-invariance, and finally, we generalize it to the most complex and flexible models available.

The first method we introduce is CDT (Convolutional Decision Trees): a general purpose binary segmentation algorithm that represent every feature as a convolution kernel and learns its parameters by maximizing the regularized information gain. These features are then combined efficiently in an oblique decision tree.

- The key advantage of the proposed algorithm is its run-time;

it trains several orders of magnitude faster than regular CNNs which makes it possible to learn features without access to special hardware.

- A very nice properties of the method are the interpretability and robustness achieved by regularizing the derivative of the kernel.

- The method achieves state-of-the-art results on Weizmann Horse data set. On neuronal segmentation data set it shows the results slightly inferior to CNNs, but significantly outperforms the best algorithms with similar training time.

The second method we introduce is TICJ (Transformation-Invariant Convolutional Jungle) – a novel image classification and segmentation algorithm based on CDT, but with transformation-invariant features inspired by a pooling operation.

- To assure that these features are transformation-invariant, we take the maximum response value of the predicate in every split, over the transformations considered (see lemma 2). We show that incorporating transformation-invariance lead to better generalization.

- Regularization is enforced in TICJ through transformation-invariance constraints, gradient regularization and through the limitation on the maximum width of the final TICJ. These design constraints render the learned features interpretable and ensure satisfactory generalization even for small data sets.

- On Yale face recognition data set the method outperforms the competitors by at least 0.3%, if we consider the algorithms that do not use additional training data. For neuronal segmentation data set the method achieves the same F-score as Convolutional Neural Networks approach, but the training only requires 3 hours in a single CPU, comparing to about one week CNN training on a GPU.

The third approach that we introduce is the TI-POOLING operator. We assemble a set of transformations that should not affect the algorithm decision and generate multiple instances of the image according to these transformations. Those instances, instead of being used for training independently, are passed through initial layers of the network and through the TI-POOLING operator to form transformation-invariant features. These features are fully-trainable using back-propagation, they possess the rich expressiveness of standard convolutional neural network features, but at the same time they do not depend on the variations in the data.

- Convolutional neural networks with TI-POOLING have some theoretical guarantees of being transformation-invariant algorithm for variations common in computer vision problems. But more importantly, they show some convenient practical properties, e.g. they permit to learn from the most representative instances, that we call "canonical".

- Because of that the networks do not have to learn features separately for every possible variation of the data from augmented samples, but instead they learn only features that are relevant for one appearance of the image, and then they apply the features for all the variations.

- They also enable better use of the input data to learn these features: e.g. all the samples including edges participate in learning transformation-invariant edge detector feature, and no separate vertical or horizontal edge detector features are needed.

- We test the method on three data sets with explicitly defined variability. In all the experiments we either significantly outperform or at least match the performance of baseline state of the art techniques. Often we also show faster convergence rates than baselines with smaller yet smarter data-aware models.

- The proposed TI-POOLING operator can be used as a separate neuronal unit for most networks architectures with very little

effort to incorporate prior knowledge on nuisance factors in the data. But the range of its applications goes well beyond that, rendering it possibleÈJB commen to incorporate many types of prior information on the data and opening the opportunities for more robust expert-driven algorithms in combination with the powerful expressiveness of deep learning.

Chapter 5

Conclusion & discussion

5.1 Findings

In this thesis we have investigated how to incorporate different types of expert knowledge to design and enhance new computer vision algorithms. We show expert knowledge to be beneficial for general computer vision problems, but even more so for problems in more specialized fields, like medical imaging.

Three major types of expert knowledge that we consider include information on:

- data peculiarities and details of data acquisition process,
- expected and/or desired properties of the solution,
- how a trained human expert approaches the problem.

Motivated by real-world problems and scenarios, we consider how these types of expert information can be incorporated in various computer vision pipelines. This research results in the development of multiple novel expert-aware approaches. We report on these approaches by grouping them into three chapters.

- First, chapter 2 introduces the properties of *anisotropic data* and describes how an expert deals with this data. By incorporating this information, we are able to better resolve ambiguities in data representation, and consequently better solve membrane segmentation and image enhancement problems.

- Then, chapter 3 demonstrates how one can less-subjectively solve the amyloid plaque detection problem with no training data by incorporating biologically motivated priors. By stating the expected properties of the solution, we are able to tune the internal parameters of the algorithm.

- Finally, chapter 4 demonstrates how computer vision algorithms can benefit from the expert-defined knowledge on nuisance variations in the data. By introducing transformation-invariance, we are able to use available data highly efficiently, resulting in satisfactory accuracy, compact models and efficient training.

Overall, based on these ideas, we introduce multiple novel computer vision and image processing algorithms.

- A neuronal membrane segmentation method that uses dense correspondences across anisotropic sections.

- SuperSlicing – an image restoration and enhancement technique for anisotropic data, that also serves within the proposed segmentation pipeline.

- A robust and non-subjective pipeline for plaque distribution estimation, featuring a feedback-loop for an automated parameter tuning from biologically motivated priors.

- CDT (Convolutional Decision Trees) – a fast general purpose binary segmentation algorithm.

- TICJ (Transformation-Invariant Convolutional Jungle) – image classification and segmentation algorithm based on CDT with transformation-invariant features.

- Convolutional Neural Network topology with TI-POOLING operator, that combines the benefits of trainable CNN features with transformation-invariance.

The following research results compose the core achievements of this thesis work and are my personal highlights

- SUPERSLICING reconstruction method achieves on average 10% better peak signal-to-noise ratio than state of the art techniques. This improvement is achieved by modelling the physics of anisotropic data acquisition, defined by imaging experts.

- The feedback-loop framework renders solutions possible for problems that are defined in unsupervised or weakly supervised setting. It automatically and non-subjectively tunes algorithm parameters based purely on biological properties of the solution, defined by medical experts.

- Transformation-invariance, which is incorporated either through TICJ features or through the TI-POOLING operator, enables highly efficient learning by exploiting training on *canonical* samples. This improvement results in high accuracy and sufficient generalization capacity, that is achieved with the small and easily trainable models.

- From an application point of view, we develop a state-of-the-art recognition pipelines for the following two problems: neuronal structure segmentation for Drosophila ventral nerve cord ssTEM images, and amyloid plaque distribution estimation in mouse brains. We also demonstrate state of the art results in various natural image recognition benchmarks, such as: Yale face, rotated MNIST and Weizmann Horse data sets.

5.2 Future work

The most promising directions of work from my point of view include addressing the limitations of current methods, further generalizing

the approaches, and enabling highly specialized expert knowledge integration.

SuperSlicing. This method currently works with uniformly accumulated anisotropic data, i.e. data, where a section can be modelled as an average of hidden subsections. While this is a common case, it is not the only one. For example, most video cameras keep the shutter close for some time between frames, which results in computing average of only a subset of subframes. Another example is different types of microscopes that capture a section as a weighted sum of subsections, with weight decreasing with depth. All these cases can be very beneficial to implement in the current SuperSlicing framework.

Neuronal segmentation. The proposed algorithms for anisotropic data segmentation allow us to combine information from neighboring sections to resolve ambiguities. But in principle, neuronal structures are 3D shapes, and should not be segmented in 2D. Generalizing the segmentation approaches to 3D is a very promising direction of work. One way to do that would be to use higher-order correspondences: employ not only pairs of sections, but multiple sections at a time.

Amyloid plaque distribution estimation. The current pipeline is simple and fast, and feedback-loop enables efficient parameter tuning, but ultimately many individual components of this pipeline can be replaced with better ones. Atlas alignment could be significantly more accurate with cross-modal registration, and by incorporating both structural constraints and texture of tissue in different regions. Filtering can be more biology-aware, for example, analysing the shape of a connected component, and not only its size. And the feedback-loop itself can be accelerated by using smarter optimization techniques instead of binary search.

Transformation-invariance. Current implementation of both TI-pooling and TICJ features requires time proportional to the size of

the set of transformations. While it is comparable with augmentation in simple cases, considering multiple types of transformations can lead to exponential growth of time required to compute a feature. Sampling the space of transformations can be one solution, as discussed above, but probably more efficient search in space of transformations can be found. For example, feature value is usually highly non-convex with respect to transformations applied, but it is still smooth, so Bayesian optimization framework [SLA12] can be used to find the argmax of the feature value.

5.3 Concluding remarks

With the development of more flexible computer vision models, larger data sets to train these models and faster computations, the field of computer vision itself is changing. Many natural image recognition problems, that were considered very challenging before, are now solved good enough for many practical applications.

Ready-to-use *black box* solutions for classification, detection and segmentation problems are now available, that are able to solve many problems at hand. They only require large enough labeled training data sets, and some time to be trained, preferably on special hardware.

Unfortunately, this has very limited impact on many more specialized problem, where there is still need to develop custom solutions. Major examples include problems in medical imaging, where both the data is very different from natural imaging, and also labeling is much more expensive, as it can only be done by trained experts.

We show in this thesis, how these problems can benefit from modern computer vision approaches when combined with crucial domain-specific information, that only domain experts can provide. We consider multiple examples of this fusion, but the idea is applicable well beyond these examples – basically any computer vision problem can benefit from expert-aware algorithms, from very narrow and specialized fields, such as medical image analysis to fields as broad as natural image recognition.

This direction of research is broad and exciting, and we managed to only scratch the surface of it. But hopefully, the results achieved in this thesis will inspire further research. No doubt that many future breakthroughs in narrow domains of image analysis will be connected with expert-aware algorithms.

Bibliography

[ACTB+15] Ignacio Arganda-Carreras, Srinivas C Turaga, Daniel R Berger, Dan Ciresan, Alessandro Giusti, Luca M Gambardella, Jürgen Schmidhuber, Dmitry Laptev, Sarvesh Dwivedi, Joachim M Buhmann, et al. Crowdsourcing the creation of image segmentation algorithms for connectomics. *Frontiers in neuroanatomy*, 9, 2015. 2, 21, 29, 31, 41

[BAKF12] Carlos Becker, Karim Ali, Graham Knott, and Pascal Fua. Learning context cues for synapse segmentation in em volumes. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI*, pages 585–592. Springer, 2012. 69

[BBB+93] Jane Bromley, James W Bentz, Léon Bottou, Isabelle Guyon, Yann LeCun, Cliff Moore, Eduard Säckinger, and Roopak Shah. Signature verification using a siamese time delay neural network. *International Journal of Pattern Recognition and Artificial Intelligence*, 7(04):669–688, 1993. 112

[BFOS84] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. Classification and regression trees. wadsworth & brooks. *Monterey, CA*, 1984. 75, 76

[BH12] Jennifer N Bourne and Kristen M Harris. Nanoscale analysis of structural synaptic plasticity. *Current opinion in neurobiology*, 22(3):372–382, 2012. 16

[BHD11] Kevin L Briggman, Moritz Helmstaedter, and Winfried Denk. Wiring specificity in the direction-selectivity circuit of the retina. *Nature*, 471(7337):183–188, 2011. 16

[BHK97] Peter N. Belhumeur, João P Hespanha, and David Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):711–720, 1997. 105

[BHS+07] Gökhan Bakir, Thomas Hofmann, Bernhard Schölkopf, Alexander J. Smola, Ben Taskar, and S.V.N Vishwanathan. *Predicting Structured Data*. MIT Press, 2007. 12

[BK04] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(9):1124–1137, 2004. 12, 24, 28, 84

[BM13] Joan Bruna and Stéphane Mallat. Invariant scattering convolution networks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1872–1886, 2013. 126

[BP66] Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966. 12

[Bre01] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. 24, 27, 37

[BRRS13] Daniel J Bumbarger, Metta Riebesell, Christian Rödelsperger, and Ralf J Sommer. System-wide rewiring underlies behavioral differences in predatory and bacterial-feeding nematodes. *Cell*, 152(1):109–119, 2013. 16

[BSL+11] Simon Baker, Daniel Scharstein, J.P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92:1–31, 2011. 19

[BU02] Eran Borenstein and Shimon Ullman. Class-specific, top-down segmentation. In *European Conference in Computer Vision – ECCV 2002*, pages 109–122. Springer, 2002. 86

[CGGS12] Dan Ciresan, Alessandro Giusti, Luca M Gambardella, and Jürgen Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in neural information processing systems*, pages 2843–2851, 2012. 31, 72, 75, 84, 86, 105, 108, 111

[cha] Isbi challenge: Segmentation of neuronal structures in em stacks (http://brainiac2.mit.edu/isbi_challenge/). 21, 29, 108

[CHH+07] Deng Cai, Xiaofei He, Yuxiao Hu, Jiawei Han, and Thomas Huang. Learning a spatially smooth subspace for face recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–7. IEEE, 2007. 106, 107

[CHH14] Alessandro Canopoli, Joshua A Herbst, and Richard HR Hahnloser. A higher sensory brain region is involved in reversing reinforcement-induced vocal changes in a

songbird. *The Journal of Neuroscience*, 34(20):7018–7026, 2014. 16

[CJG+15] Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, and Yi Ma. Pcanet: A simple deep learning baseline for image classification? *IEEE Transactions on Image Processing (TIP 2015)*, 24, 2015. 126

[CKF11] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, EPFL-CONF-192376, 2011. 122

[CMS12] Dan Ciresan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642–3649. IEEE, 2012. 70, 93, 113

[CR95] Yves Chauvin and David E Rumelhart. *Backpropagation: theory, architectures, and applications*. Psychology Press, 1995. 118

[CSP+10] Albert Cardona, Stephan Saalfeld, Stephan Preibisch, Benjamin Schmid, Anchi Cheng, Jim Pulokas, Pavel Tomancak, and Volker Hartenstein. An integrated micro-and macroarchitectural analysis of the drosophila brain by computer-assisted serial section electron microscopy. *PLoS biology*, 8(10):e1000502, 2010. 12, 29, 75, 105, 124

[EVGW+10] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010. 2

[Fij] Trainable weka segmentation (fiji plugin) (http://imagej.net/trainable_weka_segmentation). 18, 26

[gal] Galaxy zoo - the galaxy challenge (kaggle) (https://www.kaggle.com/). 2

[GGG06] Jacob Goldberger, Shiri Gordon, and Hayit Greenspan. Unsupervised image-set clustering using an information theoretic framework. *Image Processing, IEEE Transactions on*, 15(2):449–458, 2006. 103

[GM87] C Lee Giles and Tom Maxwell. Learning, invariance, and generalization in high-order neural networks. *Applied optics*, 26(23):4972–4978, 1987. 93

[Gra06] Leo Grady. Random walks for image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(11):1768–1783, 2006. 59, 60

[GS08] Alex Graves and Juergen Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *Advances in Neural Information Processing Systems*, pages 545–552, 2008. 72

[GYR+11] Juergen Gall, Angela Yao, Nima Razavi, Luc Van Gool, and Victor Lempitsky. Hough forests for object detection, tracking, and action recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(11):2188–2202, 2011. 73, 86

[HBT+13] Moritz Helmstaedter, Kevin L Briggman, Srinivas C Turaga, Viren Jain, H Sebastian Seung, and Winfried Denk. Connectomic reconstruction of the inner plexiform layer in the mouse retina. *Nature*, 500(7461):168–174, 2013. 16

[HKS93] David Heath, Simon Kasif, and Steven Salzberg. Induction of oblique decision trees. *International Joint Conference on Artificial Intelligence, IJCAI*, 1993. 72, 75, 76, 101

[HNIV+12] Tao Hu, Juan Nunez-Iglesias, Shiv Vitaladevuni, Lou Scheffer, Shan Xu, Mehdi Bolorizadeh, Harald Hess, Richard Fetter, and Dmitri Chklovskii. Super-resolution using sparse representations over learned dictionaries: Reconstruction of brain structure using electron microscopy. *CoRR*, abs/1210.0564, 2012. 19

[HS91] Robert M Haralock and Linda G Shapiro. *Computer and robot vision*, volume 1. Addison-Wesley Longman Publishing Co., Inc., 1991. 60

[HSB+04] Jan Huisken, Jim Swoger, Filippo Del Bene, Joachim Wittbrodt, and Ernst HK Stelzer. Optical sectioning deep inside live embryos by selective plane illumination microscopy. *Science*, 305:1007–1009, 2004. 49

[HSK+12] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012. 117, 124

[HVD07] Gang Hua, Paul A Viola, and Steven M Drucker. Face recognition using discriminatively trained orthogonal rank one tensor projections. In *Computer Vision and Pattern Recognition CVPR 2007. IEEE Conference on*, pages 1–8. IEEE, 2007. 106, 107

[IG13] Saadia Iftikhar and Afzal Godil. Feature measures for the segmentation of neuronal membrane using a machine learning algorithm. In *Sixth International Conference on Machine Vision (ICMV 13)*, pages 90670V–

90670V. International Society for Optics and Photonics, 2013. 31

[Jac13] Steven L Jacques. Optical properties of biological tissues: a review. *Physics in medicine and biology*, 58(11):R37, 2013. 49

[JBB+10] G Allan Johnson, Alexandra Badea, Jeffrey Brandenburg, Gary Cofer, Boma Fubara, Song Liu, and Jonathan Nissanov. Waxholm space: an image-based reference for coordinating mouse brain research. *Neuroimage*, 53(2):365–372, 2010. 55, 66

[JBR+10] Viren Jain, Benjamin Bollmann, Mark Richardson, Daniel R. Berger, Moritz N. Helmstaedter, Kevin L. Briggman, Winfried Denk, Jared B. Bowden, John M. Mendenhall, Wickliffe C. Abraham, Kristen M. Harris, Narayanan Kasthuri, Ken J. Hayworth, Richard Schalek, Juan Carlos Tapia, Jeff W. Lichtman, and H. Sebastian Seung. Boundary learning by optimization with topological constraints. In *CVPR*, pages 2488–2495, 2010. 29, 41

[JBWB+15] Nina Jahrling, Klaus Becker, Bettina M Wegenast-Braun, Stefan A Grathwohl, Mathias Jucker, and Hans-Ulrich Dodt. Cerebral β-amyloidosis in mice investigated by ultramicroscopy. *PLoS ONE*, 10(5), 05 2015. 57

[JS01] Mark Jenkinson and Stephen Smith. A global optimisation method for robust affine registration of brain images. *Medical image analysis*, 5(2), 2001. 61

[JST10] Viren Jain, H Sebastian Seung, and Srinivas C Turaga. Machines that learn to segment images: a crucial technology for connectomics. *Current opinion in neurobiology*, 20(5):653–666, 2010. 17

[JSZK15] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. *Advances in Neural Information Processing Systems (NIPS 2015)*, 2015. 113, 114, 122, 126, 127

[JWB+12] Travis A Jarrell, Yi Wang, Adam E Bloniarz, Christopher A Brittin, Meng Xu, J Nichol Thomson, Donna G Albertson, David H Hall, and Scott W Emmons. The connectome of a decision-making neural network. *Science*, 337(6093):437–444, 2012. 16

[JWG+13] Elizabeth Jurrus, Shigeki Watanabe, Richard J Giuly, Antonio RC Paiva, Mark H Ellisman, Erik M Jorgensen, and Tolga Tasdizen. Semi-automated neuron boundary detection and nonbranching process segmentation in electron microscopy images. *Neuroinformatics*, 11(1):5–29, 2013. 31

[Kai67] Thomas Kailath. The divergence and bhattacharyya distance measures in signal selection. *Communication Technology, IEEE Transactions on*, 15(1):52–60, 1967. 103

[KBL+16] Daniel Kirschenbaum, Olivier Bichsel, Dmitry Laptev, Michael B. Smith, and Adriano Aguzzi. Rapid electrophoretic tissue clearing and molecular labelling in whole-mount mouse brain. *manuscript is in preparation*, 2016. 49, 64

[KDMR+03] Kenneth Kreutz-Delgado, Joseph F Murray, Bhaskar D Rao, Kjersti Engan, Te-Won Lee, and Terrence J Sejnowski. Dictionary learning algorithms for sparse representation. *Neural computation*, 15(2):349–396, 2003. 69, 72

[KF11] Verena Sabine Kaynig-Fittkau. *Machine learning approaches for neuron geometry extraction and synapse*

detection in electron microscopy images. PhD thesis, ETH Zurich, Diss. Nr. 19559, 2011. 13

[KFB10a] Verena Kaynig, Thomas J. Fuchs, and Joachim M. Buhmann. Geometrical consistent 3d tracing of neuronal processes in sstem data. In *MICCAI 2010*, pages 209–216. Springer Berlin / Heidelberg, 2010. 18, 39, 41

[KFB10b] Verena Kaynig, Thomas J. Fuchs, and Joachim M. Buhmann. Neuron geometry extraction by perceptual grouping in sstem images. In *CVPR*, pages 2902–2909. IEEE, 2010. 18, 22, 27, 28

[KGZ+14] Jinseop S Kim, Matthew J Greene, Aleksandar Zlateski, Kisuk Lee, Mark Richardson, Srinivas C Turaga, Michael Purcaro, Matthew Balkam, Amy Robinson, Bardia F Behabadi, et al. Space-time wiring specificity supports direction selectivity in the retina. *Nature*, 509(7500):331, 2014. 16

[KHB+15] Narayanan Kasthuri, Kenneth Jeffrey Hayworth, Daniel Raimund Berger, Richard Lee Schalek, José Angel Conchello, Seymour Knowles-Barley, Dongil Lee, Amelio Vázquez-Reina, Verena Kaynig, Thouis Raymond Jones, et al. Saturated reconstruction of a volume of neocortex. *Cell*, 162(3):648–661, 2015. 16

[KRFL09] Koray Kavukcuoglu, M Ranzato, Rob Fergus, and Yann LeCun. Learning invariant features through topographic filter maps. In *Computer Vision and Pattern Recognition CVPR 2009. IEEE Conference on*, pages 1605–1612. IEEE, 2009. 93

[KRP10] Ritwik Kumar, Amelio V. Reina, and Hanspeter Pfister. Radon-like features and their application to connectomics. In *MMBIA*. IEEE, 2010. 22, 27

[KS+80] Ross Kindermann, James Laurie Snell, et al. *Markov random fields and their applications*, volume 1. American Mathematical Society Providence, RI, 1980. 12

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 110, 113, 128

[LB95] Yann LeCun and Yoshua Bengio. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995. 1, 12, 72, 93, 110, 111

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 71, 112, 124

[LEC+07] Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th international conference on Machine learning*, pages 473–480. ACM, 2007. 124

[LGTB97] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, and Andrew D Back. Face recognition: A convolutional neural-network approach. *Neural Networks, IEEE Transactions on*, 8(1):98–113, 1997. 69, 72

[Liu09] Ce Liu. *Beyond pixels: exploring new representations and applications for motion analysis*. PhD thesis, Massachusetts Institute of Technology Cambridge, MA, USA, 2009. 32, 35

[LJB+95] Yann LeCun, LD Jackel, L Bottou, A Brunot, C Cortes, JS Denker, H Drucker, I Guyon, UA Muller, E Sackinger, et al. Comparison of learning algorithms for handwritten digit recognition. In *International conference on artificial neural networks*, volume 60, pages 53–60, 1995. 110

[LLPS93] Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6):861–867, 1993. 109

[Low99] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. IEEE, 1999. 26, 39, 69, 91, 112

[LSA+10] Aurélien Lucchi, Kevin Smith, Radhakrishna Achanta, Vincent Lepetit, and Pascal Fua. A fully automated approach to segmentation of irregularly shaped cellular structures in em images. In *MICCAI*, pages 463–471, 2010. 27

[LSD12] Alex Levinshtein, Cristian Sminchisescu, and Sven Dickinson. Optimal image and video closure by superpixel grouping. *International journal of computer vision*, 100(1):99–119, 2012. 86, 88

[LSP+04] Svetlana Lazebnik, Cordelia Schmid, Jean Ponce, et al. Semi-local affine parts for object recognition. In *British Machine Vision Conference (BMVC'04)*, pages 779–788, 2004. 91

[LTSG14] Dmitry Laptev, Alexey Tikhonov, Pavel Serdyukov, and Gleb Gusev. Parameter-free discovery and recommendation of areas-of-interest. In *22nd ACM SIGSPATIAL*

International Conference on Advances in Geographic Information Systems, pages 113–122. ACM, 2014. 57

[LYT11] Ce Liu, Jenny Yuen, and Antonio Torralba. Sift flow: Dense correspondence across scenes and its applications. *Trans. Pattern Anal. Mach. Intell.*, 33(5):978–994, 2011. 22, 24, 26, 35

[mat] Matlab image processing toolbox (http://www.mathworks.com/help/images/). 69

[MBP+08] Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro, and Andrew Zisserman. Supervised dictionary learning. *arXiv preprint arXiv:0809.3083*, 2008. 69

[Mey94] Fernand Meyer. Topographic distance and watershed lines. *Signal processing*, 38(1):113–125, 1994. 52

[MMS98] Patrick D. Meek, E. Kristin McKeithan, and Glen T. Schumock. Economic considerations in alzheimer's disease. *Pharmacotherapy: The Journal of Human Pharmacology and Drug Therapy*, 18(2P2):68–73, 1998. 48

[Mor05] Greg Mori. Guiding model search using segmentation. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1417–1423. IEEE, 2005. 12

[MTS+13] Albert Montillo, J Tu, Jamie Shotton, John Winn, Juan Iglesias, DN Metaxas, and Antonio Criminisi. Entanglement and differentiable information gain maximization. In *Decision Forests for Computer Vision and Medical Image Analysis*, pages 273–293. Springer, 2013. 77

[MVS99] Frederik Maes, Dirk Vandermeulen, and Paul Suetens. Comparative evaluation of multiresolution optimization

strategies for multimodality image registration by maximization of mutual information. *Medical image analysis*, 3(4):373–386, 1999. 60

[NNY94] Yurii Nesterov, Arkadii Nemirovskii, and Yinyu Ye. *Interior-point polynomial algorithms in convex programming*, volume 13. SIAM, 1994. 35

[Noc80] Jorge Nocedal. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980. 80, 98

[Par62] Emanuel Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962. 52

[PBC+09] Yan Ping, Adam W. Bero, John R. Cirrito, Qingli Xiao, Xiaoyan Hu, Yan Wang, Ernesto Gonzales, David M. Holtzman, and Jin-Moo Lee. Characterizing the appearance and growth of amyloid plaques in app/ps1 mice. *The Journal of Neuroscience*, 29(34), 2009. 48, 57, 63

[PCI+08] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 92

[Qui86] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986. 90, 100

[RBK+06] Rebecca Radde, Tristan Bolmont, Stephan A Kaeser, Janaky Coomaraswamy, Dennis Lindau, Lars Stoltze, Michael E Calhoun, Fabienne Jäggi, Hartwig Wolburg, Simon Gengler, Christian Haass, Bernardino Ghetti, Christian Czech, Christian Hölscher, Paul M Mathews,

and Mathias Jucker. Aβ42-driven cerebral amyloidosis in transgenic mice reveals early and robust pathology. *EMBO reports*, 7(9):940–946, 2006. 64

[RDS+15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 1, 2

[RP99] Maximilian Riesenhuber and Tomaso Poggio. Hierarchical models of object recognition in cortex. *Nature neuroscience*, 2(11):1019–1025, 1999. 89, 93

[SB07] Kristian Sandberg and Moorea Brega. Segmentation of thin structures in electron micrographs using orientation fields. *Journal of structural biology*, 157(2):403–415, 2007. 27, 28, 39, 69, 91, 112

[SC08] Honghao Shan and Garrison W Cottrell. Looking around the backyard helps to recognize faces and digits. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008. 106, 107

[Seu12] Sebastian Seung. Connectome: How the brain's wiring makes us who we are. *Houghton Mifflin Harcourt*, 2012. 13, 15

[SFI11] Oded Shahar, Alon Faktor, and Michal Irani. Space-time super-resolution from a single video. In *CVPR*, pages 3353–3360, 2011. 41

[SFO+10] Peter J Schüffler, Thomas J Fuchs, Cheng Soon Ong, Volker Roth, and Joachim M Buhmann. Computational tma analysis and cell nucleus classification of re-

nal cell carcinoma. In *Pattern Recognition*, pages 202–211. Springer, 2010. 70, 110

[SL12] Kihyuk Sohn and Honglak Lee. Learning invariant representations with local transformations. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 1311–1318, 2012. 93, 126

[SLA12] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012. 137

[SLC04] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: A local svm approach. In *ICPR*, 2004. 42

[SM80] Jack Sklansky and Leo Michelotti. Locally trained piecewise linear classifiers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2:101–111, 1980. 75, 76

[SMKLM15] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 945–953, 2015. 115

[SOSS10] Mihoko Shimano, Takahiro Okabe, Imari Sato, and Yoichi Sato. Video temporal super-resolution based on self-similarity. In *ACCV*, pages 93–106, 2010. 20

[SSK+13a] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013. 2

[SSK+13b] Jamie Shotton, Toby Sharp, Pushmeet Kohli, Sebastian Nowozin, John Winn, and Antonio Criminisi. Decision jungles: Compact and rich models for classification. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 234–242. Curran Associates, Inc., 2013. 90, 91, 101, 103

[TBL+13] Shin-ya Takemura, Arjun Bharioke, Zhiyuan Lu, Aljoscha Nern, Shiv Vitaladevuni, Patricia K Rivlin, William T Katz, Donald J Olbris, Stephen M Plaza, Philip Winston, et al. A visual motion detection circuit suggested by drosophila connectomics. *Nature*, 500(7461):175–181, 2013. 16

[TJHA05] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. In *Journal of Machine Learning Research*, pages 1453–1484, 2005. 12

[TS12] Xiao Tan and Changming Sun. Membrane extraction using two-step classification and post-processing. In *Proceedings of ISBI*, 2012. 31

[TYHD14] Raju Tomer, Li Ye, Brian Hsueh, and Karl Deisseroth. Advanced clarity for rapid and high-resolution imaging of intact tissues. *Nature protocols*, 9(7):1682–1697, 2014. 49

[VDM01] David A Van Dyk and Xiao-Li Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 10(1), 2001. 110, 111

[VFB11] Alexander Vezhnevets, Vittorio Ferrari, and Joachim M Buhmann. Weakly supervised semantic segmentation with a multi-image model. In *Computer Vision (ICCV),*

2011 IEEE International Conference on, pages 643–650. IEEE, 2011. 12

[VLBM08] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008. 69

[VRHG+11] Amelio Vazquez-Reina, Daniel Huang, Michael Gelbart, Jeff Lichtman, Eric Miller, and Hanspeter Pfister. Segmentation fusion for connectomics. In *ICCV*. IEEE, 2011. 18

[VS01] Alexander Vasilevskiy and Kaleem Siddiqi. Flux maximizing geometric flows. *Trans. Pattern Anal. Mach. Intell.*, 24:1565–1578, 2001. 27, 28

[WSTB86] John G. White, Eileen Southgate, J. Nichol Thomson, and Sydney Brenner. The structure of the nervous system of the nematode caenorhabditis elegans: the mind of a worm. *Phil. Trans. R. Soc. Lond*, 314:1–340, 1986. 16

[WYHY15] Jiajun Wu, Yinan Yu, Chang Huang, and Kai Yu. Deep multiple instance learning for image classification and auto-annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3460–3469, 2015. 111, 114, 128

[YJHN07] Jun Yang, Yu-Gang Jiang, Alexander G Hauptmann, and Chong-Wah Ngo. Evaluating bag-of-visual-words representations in scene classification. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, pages 197–206. ACM, 2007. 69

[YN10] Yi Yang and Shawn Newsam. Bag-of-visual-words and spatial extensions for land-use classification. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 270–279. ACM, 2010. 92, 112

[YYH10] Jianchao Yang, Kai Yu, and Thomas Huang. Supervised translation-invariant sparse coding. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3517–3524. IEEE, 2010. 92

[ZCY10] Long Zhu, Yuanhao Chen, and Alan Yuille. Learning a hierarchical deformable template for rapid deformable object parsing. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(6):1029–1043, 2010. 86, 88

[Zei12] Matthew D Zeiler. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012. 124

[ZF03] Barbara Zitova and Jan Flusser. Image registration methods: a survey. *Image and vision computing*, 21(11):977–1000, 2003. 18

[ZYCA06] Qiang Zhu, Mei-Chen Yeh, Kwang-Ting Cheng, and Shai Avidan. Fast human detection using a cascade of histograms of oriented gradients. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1491–1498. IEEE, 2006. 69

Curriculum Vitae

Name	Dmitry Laptev
Date of birth	February 03, 1989
09/2004 – 06/2006	High school studies, Advanced Educational Scientific Center A.N. Kolmogorov School (AESC MSU), Moscow, Russian Federation
09/2006 – 07/2011	Diploma studies, Lomonosov Moscow State University, Moscow, Russian Federation, Department of Computer Science and Applied Mathematics, Supervisor: Prof. Dr. Dmitry Vetrov
12/2011 – 12/2016	Doctoral studies, ETH Zürich, Zürich, Switzerland, Department of Computer Science, Institute for Machine Learning, Information Science and Engineering group, Supervisor: Prof. Dr. Joachim M. Buhmann