Fully Unsupervised Image Denoising, Diversity Denoising and Image Segmentation with Limited Annotations

BY

MANGAL PRAKASH
Born Jan 28, 1992, India

PhD THESIS

Submitted in fulfillment of the requirements
for the degree of Doktor-Ingenieur (Dr.-Ing) in Computer Science
in the Faculty of Computer Science at the
Technische Universität Dresden, 2022

Dresden, Sachsen, DE

Doctoral advisor: Dr. Florian Jug

Doctoral Committee:

Prof. Dr. Stefan Gumhold, Committee Chair
Prof. Dr. Raimund Dachselt, Committee Member
Prof. Dr. Ivo Sbalzarini, Reviewer and Committee Member
Dr. Jan Funke, Reviewer and Committee Member
Prof. Dr. Carsten Rother, Fachreferent and Committee Member

To my parents and my sister

For their unwavering support and valuable life lessons at all times when going got tough.

ACKNOWLEDGMENTS

This thesis marks a gratifying culmination of four years of my PhD research. Although I am penning down this thesis alone, I feel that there are many individuals all along my PhD journey who enabled the research presented here and helped me in shaping this work directly or indirectly. I would like to express my gratitude and sincerely thank all these people.

First of all, I wish to express my sincerest gratitude to my PhD supervisor, Dr. Florian Jug for his guidance, support and wisdom not only throughout my PhD research but also for helping me grow in my personal as well as professional life. I remember the time when I had just joined his group and did not have any experience with either coding or computer vision/machine learning or even life sciences. Florian always encouraged me to explore and learn new things and taught me not to be shy of asking critical or even trivial questions. In hindsight, I can say that he always had a vision to indirectly shape my research without imposing his ideas and thoughts. In the beginning, he gave me constant support and shared his ideas while towards the end he gave me enough freedom to realize my own ideas and interests. All throughout I felt that he was there to guide me onto the right path whenever it appeared that my research may veer into a potentially unproductive direction. I am also very grateful to him for holding my hand and guiding me through paper writing process, introducing me to his international network of researchers and facilitating my path through the research arena. Besides research, Florian is someone who I have always looked as an inspiration in my personal life as well. I have learnt from him how to be cheerful even in the most difficult times, how to constantly strive for perfection and how to maintain healthy professional and personal relationships. I am very thankful to him for always believing in me and also helping me with strong references, valuable advice and mock interview preparations throughout my job search. I could not have asked for a better PhD supervisor.

I also wish to sincerely thank Dr.Alexander Krull, who along with Florian, has inspired me to take up research in the field of Computer Vision. I have worked so closely with Alex that I feel he is my second supervisor. The numerous stimulating and engaging discussions that I have had with Alex gave me impetus to keep pursuing my research in a meaningful way. I learnt from Alex how to think methodically about any research problem and also communicate my ideas to others in a cohesive manner. He always entertained any research ideas that me and my other colleagues had and was willing to engage in thoughtful discussions to shape our ideas into useful finished products. I appreciate his quality of always being available for his junior researchers at all times and I hope to imbibe this quality in my future career. I have spent countless hours with him working on our projects and trying to answer many interesting questions. Alex provided critical advice through all my projects and this work would not have materialized without his key insights and supervision. Finally, I am also grateful to Alex for helping me with paper writing process and providing me with strong references during my job search.

I would like to acknowledge the Jug lab members for the great peer support, advice and fun times which made this journey so memorable for me. I had the pleasure of working together on some of the projects and co-authoring papers with my PhD colleagues Tim-Oliver Buchholz and

Manan Lalit. They were both kind to be always available to discuss. We collaboratively explored our ideas and shared our excitements and disappointments when these ideas materialized or failed. I enjoyed the company of my labmates Anna Goncharova, Nuno Pimpão Martins, Tobias Pietzsch, Matthias Arzt, Tom Burke, Joran Deschamps, Deborah Schmidt and Gabriella Turek on white board discussions as well as during our regular lunch brakes.

My Thesis Advisory Committee members Dr. Pavel Tomancak, Prof. Dr. Ivo Sbalzarini and Dr. Lutz Brusch were very instrumental in keeping my research projects on track and giving encouraging ideas during our regular discussions. I also express my gratitude to my thesis reviewers: Florian, Prof. Dr. Ivo Sbalzarini and Prof. Dr. Carsten Rother for reading my thesis and giving valuable feedback.

My collaborators in Tomancak lab, Rother lab, Vermot lab, Goldstein lab and Tabler lab are all amazing human beings and helped me understand the basics of biology and discrete optimization which allowed me to contribute to these areas which I am not an expert in. Specially, I would like to thank Dr. Bogdan Savchynskky, Stefan Haller and Prof. Dr. Carsten Rother for our fruitful collaborations in cell segmentation and cell tracking. I will be forever grateful to Diana Afonso for discussions with all my research projects and providing emotional support during the PhD. Apart from them, the whole staff of MPI-CBG has been very welcoming and friendly which made my stay here a pleasurable experience.

Finally, I am forever indebted to my parents and my sister who were always supportive of me and believed in my abilities. There were several sacrifices at their end which allowed me to focus on research. Their contributions to my life and consequently to my research are innumerable and I do not have enough words to express my gratitude to them. My school friends Shashi and Mithilesh have been pillars of support for me, helping me to get through the most difficult times in my life.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

CHAPTER PAGE

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (Continued)

FIGURE **PAGE**

LIST OF ABBREVIATIONS

AP	Average Precision
CARE	Content-Aware Image Restoration
CNN	Convolutional Neural Network
CT	Computed Tomography
DIP	Deep Image Prior
DL	Deep Learning
DN	DivNoising
FBP	Filtered Backprojection
FCN	Fully Convolutional Network
FT	Finetune
FT Seq.	Finetune Sequential
GAN	Generative Adversarial Network
GDD	Generative Diversity Denoising
GMM	Gaussian Mixture Model
GT	Ground Truth
HDN	Hierarchical DivNoising
ILP	Integer Linear Program
MAP	Maximum A Posteriori
MMSE	Minimum Mean Squared Error
MRI	Magnetic Resonance Imaging
MSE	Mean Squared Error
NM	Noise Model
N2N	Noise2Noise
N2V	Noise2Void
OCR	Optical Character Recognition
PSNR	Peak Signal-to-Noise Ratio
PN2V	Probabilistic Noise2Void
RF	Receptive Field
Seq.	Sequential

LIST OF ABBREVIATIONS (Continued)

SNR	Signal-to-Noise Ratio
SOTA	State-of-the-Art
SSIM	Structural Similarity
S2S	Self2Self
VAE	Variational Autoencoder

ABSTRACT

Understanding the processes of cellular development and the interplay of cell shape changes, division and migration requires investigation of developmental processes at the spatial resolution of single cell. Biomedical imaging experiments enable the study of dynamic processes as they occur in living organisms. While biomedical imaging is essential, a key component of exposing unknown biological phenomena is quantitative image analysis.

Biomedical images, especially microscopy images, are usually noisy owing to practical limitations such as available photon budget, sample sensitivity, etc. Additionally, microscopy images often contain artefacts due to the optical aberrations in microscopes or due to imperfections in camera sensor and internal electronics. The noisy nature of images as well as the artefacts prohibit accurate downstream analysis such as cell segmentation. Although countless approaches have been proposed for image denoising, artefact removal and segmentation, supervised Deep Learning (DL) based content-aware algorithms are currently the best performing for all these tasks.

Supervised DL based methods are plagued by many practical limitations. Supervised denoising and artefact removal algorithms require paired corrupted and high quality images for training. Obtaining such image pairs can be very hard and virtually impossible in most biomedical imaging applications owing to photosensitivity and the dynamic nature of the samples being imaged. Similarly, supervised DL based segmentation methods need copious amounts of annotated data for training, which is often very expensive to obtain. Owing to these restrictions, it is imperative to look beyond supervised methods. The objective of this thesis is to develop novel unsupervised alternatives for image denoising, and artefact removal as well as semi-supervised approaches for image segmentation.

The first part of this thesis deals with unsupervised image denoising and artefact removal. For unsupervised image denoising task, this thesis first introduces a probabilistic approach for training DL based methods using parametric models of imaging noise. Next, a novel unsupervised diversity denoising framework is presented which addresses the fundamentally non-unique inverse nature of image denoising by generating multiple plausible denoised solutions for any given noisy image. Finally, interesting properties of the diversity denoising methods are presented which make them suitable for unsupervised spatial artefact removal in microscopy and medical imaging applications.

In the second part of this thesis, the problem of cell/nucleus segmentation is addressed. The focus is especially on practical scenarios where ground truth annotations for training DL based segmentation methods are scarcely available. Unsupervised denoising is used as an aid to improve segmentation performance in the presence of limited annotations. Several training strategies are presented in this work to leverage the representations learned by unsupervised denoising networks to enable better cell/nucleus segmentation in microscopy data. Apart from DL based segmentation methods, a proof-of-concept is introduced which views cell/nucleus segmentation from the perspective of solving a label fusion problem. This method, through

limited human interaction, learns to choose the best possible segmentation for each cell/nucleus using only a pool of diverse (and possibly faulty) segmentation hypotheses as input.

In summary, this thesis seeks to introduce new unsupervised denoising and artefact removal methods as well as semi-supervised segmentation methods which can be easily deployed to directly and immediately benefit biomedical practitioners with their research.

CHAPTER 1

INTRODUCTION

Image is an international language.

Marjane Satrapi

Images contain a wealth of information. Images provide an immersive experience by adding context to a description, evoking emotions, capturing moments in time and often communicating complex ideas clearly and concisely. From the cave paintings drawn by our ancestors thousands of years ago to the digital photos taken by our modern day cameras, we have always used images as a reliable source to convey information. Not surprisingly, what makes images so useful is the innate ability of humans to parse them.

Humans excel at the task of extracting useful information from images and a considerable body of research has been dedicated to understanding this extraordinary feat of human visual system [1–3]. We can extract semantically meaningful data even from previously unseen images rather effortlessly. No wonder why Aristotle, in as early as 4th century B.C., proclaimed vision as the most important of the basic senses necessary for us to experience the world and survive. Despite being blessed with excellent visual system and skills, manually analyzing large amounts of images can become mundane and can also induce errors.

In the wake of modern scientific imaging advances in application areas such as microscopy, astronomy, medicine, optical character recognition, robotics, machine vision, etc., high volume of image data is generated on a regular basis. For instance, in life science research, imaging using modern microscopy modalities such as light sheet fluorescence, confocal, electron microscopy, etc. can generate terabytes of images in a single imaging session spanning few hours. Similarly, time lapse microscopy can produce microscopy movies consisting of hundreds and thousands of frames. Microscopy images need to be analyzed to understand the processes of cellular development and the interplay of cell shape changes, division and migration at the spatial resolution of single cells. The sheer volume of images generated everyday makes it near impossible for humans to manually analyze them accurately and efficiently. Thus, automated quantitative image analysis is a necessity for exposing unknown biological phenomena.

Automated analysis of microscopy images spans many tasks. Examples include the automatic classification of different cell types within large cell cultures [4], segmentation of cells/nuclei to study the morphological characteristics [5], tracking of cells/nuclei to study tissue development and cell fate changes [6], denoising and artefact removal for intensity quantification and improving downstream analyses [7, 8], image registration for alignment of multimodal data [9], object detection for localising biomarkers for medical prognosis [10] as well as many others.

The focus of this thesis is limited to image restoration (denoising and artefact removal) and image segmentation. Although countless Machine Learning and Computer Vision based

approaches have been proposed thus far for these tasks, existing tools still leave plenty of room for improvements. Better image restoration methods and accurate segmentation tools are thus an undeniable necessity. This thesis addresses some voids present in available image restoration and segmentation approaches by introducing new machine learning algorithms and software tools. The following sections provide a brief overview of image restoration and image segmentation tasks, shed light on the challenges associated with automating these tasks in the context of microscopy and finally briefly introduce the new methods that are presented in this thesis to alleviate these challenges.

1.1 Image Restoration

The objective of image restoration is to restore/recover clean image(s) given corresponding degraded image(s). A wide range of degradations can give rise to image corruptions during either the acquisition process or preprocessing/postprocessing stages. For instance, unwanted movement of camera or the scene during image acquisition gives rise to motion blur [11], defocus blur often occurs in microscopy due to optical aberrations [12] or the insufficient focusing accuracy of the autofocusing system [13], different types of noise often corrupt images in natural and biomedical domains due to insufficient lighting/photon budget [14], digital noise/artefacts can occur due to imperfections in camera sensor and internal electronics [15], quantization artefacts such as JPEG artefacts arise during image compression [16].

This thesis deals with the restoration of images corrupted with pixel noises (Gaussian and Poisson noises) and spatial artefacts in biomedical image domain. In biomedical applications such as microscopy, photon budget for imaging is often limited due to the health of the living sample being imaged. Thus, a compromise has to be made between imaging speed, exposure time and signal-to-noise ratio. This is manifested as shot noise (Poisson noise). Another common source of pixel noise in microscopy is readout noise (Gaussian noise) which occurs on account of sensor characteristics and electronic imperfections. Thus, pixel noises are inevitable in microscopy images. Additionally, spatially correlated noise such as line artefacts are common in microscopy owing to column defects in sensors used for imaging [17] while in medical imaging such as Computed Tomography (CT), spatial streaking artefacts are common due to employed image reconstruction mechanism [18]. I present novel methods and algorithms for pixel noise and spatial artefact removal in the context of biomedical images. Still, most of the ideas presented in this thesis are generic enough and occasionally I present their applications to natural image domain as well.

1.2 Image Segmentation

The goal of image segmentation is to partition a given image into semantically meaningful parts. Image segmentation is a key component in many real-world applications such as face recognition [19], autonomous driving [20,21], video surveillance [22,23], robot navigation [24,25], biomedical image analysis [26, 27], smart homes [28, 29] and many others. However, there is no one method that solves all kinds of segmentation problems in practice and hence specialized

segmentation algorithms have been designed and employed depending on the downstream task at hand.

This thesis focuses on automatic segmentation of cells/nuclei in microscopy images. The automatic segmentation of microscopy images poses many challenges. Firstly, for many datasets, high cell/nucleus density and cluster formation makes it difficult to separate all touching/overlapping instances correctly. Secondly, there may be heterogeneity in sizes, shapes and intensities of cells/nuclei in the same dataset. Thirdly, noise and artefacts present in microscopy images reduce the signal-to-noise ratio and induce low contrast, thereby making the segmentation task extremely challenging. Last but not least, the sheer volume of images generated by modern microscopes simply makes the task of manual segmentation impossible. Hence, accurate (semi-)automatic image segmentation tools in microscopy are an undeniable necessity. I present novel tools for segmentation to tackle these problems and show that the proposed algorithms scale well for automatic image segmentation for large datasets without much manual effort.

1.3 Contributions of the Thesis

This thesis proposes several methods that establish new state-of-the-art in image restoration and segmentation. A brief summary of the main contributions are listed below:

- **A parametric probabilistic unsupervised pixel-wise noise removal framework** is introduced which remedies some deficiencies of the existing state-of-the-art denoising method Probabilistic Noise2Void and leads to significant improvements for blind image denoising.
- **A diversity denoising framework using Variational Autoencoders** is introduced to provide multiple restored solutions for a given noisy image at hand. This approach answers a long standing problem of generating multiple plausible restored solutions for a corrupted image. Furthermore, this approach is effective for not only pixel-wise noises but also spatial artefacts such as line artefacts in microscopy and streaking artefacts in computed tomography.
- **Frameworks for utilizing unsupervised denoising to perform Deep Learning based segmentation with few ground truth annotations**. In this thesis, the effectiveness of transfer learning between denoising and segmentation tasks is explored and multiple novel schemes will be proposed to exploit the representations learned by unsupervised denoising networks to perform high quality segmentation using very limited quantity of segmentation ground truth annotations.
- **Discrete optimization based semi-supervised label fusion framework** for image segmentation is presented. I present a proof-of-concept for harnessing the diversity present in multiple poor quality segmentations (which are easy to obtain) from different segmentation methods to estimate a high quality segmentation with very less human effort.

1.4 Thesis Overview

This thesis can be broadly divided into two parts with Chapters 1-3 dealing with image denoising and artefact removal and Chapters 4-7 tackling the problem of image segmentation. The upcoming chapters of this thesis are organized as follows.

- Chapter2 talks about an existing deep learning based unsupervised denoising method called Probabilistic Noise2Void and points out some deficiencies of this method. Then, I present a modification to this approach to counter these drawbacks and show the effectiveness of the modified approach for blind image denoising. This chapter corresponds to the paper "Fully Unsupervised Probabilistic Noise2Void" (Prakash, Lalit, Tomancak, Krull, Jug, 2020) which was presented at ISBI 2020.
- Chapter3 presents a novel unsupervised diversity denoising framework called DivNoising to account for inherent ambiguities in noisy images by producing multiple plausible denoising solutions. This research was presented at ICLR 2021 as "Fully Unsupervised Diversity Denoising with Convolutional Variational Autoencoders" (Prakash, Krull, Jug, 2021).
- Chapter4 contains an extension of DivNoising presented in Chapter 3. Here, I present Hierarchical DivNoising which alleviates some of the drawbacks of DivNoising. Additionally, the scope of diversity denoising frameworks is expanded in this chapter to deal with spatial artefacts as well. This work is currently under review as "Removing Pixel Noises and Spatial Artefacts with Generative Diversity Denoising Methods" (Prakash, Delbracio, Milanfar, Jug, 2021).
- Chapter 5 investigates if Deep Learning based segmentation networks can benefit from prior denoising especially when limited quantity of segmentation annotations are available for training the segmentation network. This research has appeared at ISBI 2020 as "Leveraging Self-Supervised Denoising for Image Segmentation" (Prakash, Buchholz, Lalit, Tomancak, Jug, Krull, 2020).
- Chapter 6 builds on the findings in the previous chapter and puts forward a joint denoising and segmentation framework and illustrates the benefits of exploiting the representations learned by unsupervised denoising for high quality image segmentation. This research was presented at BioImage Computing workshop at ECCV 2020 as "DenoiSeg: Joint Denoising and Segmentation" (Buchholz, Prakash, Schmidt, Krull, Jug, 2020).
- Chapter 7 tackles the segmentation problem from a label fusion perspective and introduces a proof of concept algorithm called MetaSeg to obtain high quality segmentation results given only poor quality segmentations from multiple segmentation sources.
- Chapter 8 summarizes the ideas and results presented in this dissertation and also suggests possible directions for future work.

In the rest of the thesis, the pronoun "we" refers to the author and the reader.

Part I

Image Restoration

I-A
Unsupervised Non-Diversity Denoising

CHAPTER 2

FULLY UNSUPERVISED PROBABILISTIC NOISE2VOID

Image acquisition forms the core of modern research in many disciplines such as astronomy [30], microscopy [31], medicine [32], materials science [33], defense [34], remote sensing [35] and many others. However, images are often damaged by undesirable corruptions such as noise, blur and other artefacts. The practical limitations of imaging pipelines dictate that these corruptions cannot be avoided entirely.

Consider the example of microscopy where noise is typically the most common cause of image degradation. For acquiring good quality microscopy images which can lend insights into biological processes, we want to optimize for spatial resolution, temporal resolution, achievable fluorophore density, phototoxicity and total duration of experiment. But all these aspects cannot be optimized at the same time and a compromise needs to be made. For instance, we are often forced to image biological samples at low signal intensities resulting in hard to analyze noisy images with low signal-to-noise ratio (SNR). The SNR can be easily improved by either increasing the exposure time or laser power but these settings are detrimental to the health of the sample being imaged and hence we are forced to decrease the total duration of the imaging experiment. Thus, noisy low SNR images cannot be precluded in microscopy if long time lapse images need to be acquired.

The most abundant sources of noise in microscopy are pixel noises such as Gaussian readout noise and Poisson shot noise. Gaussian noise arises out of electronic components which convert photons to digital signal whereas Poisson noise is a manifestation of randomness of photons coming on to the detector. Both these noises are called pixel noises since Gaussian noise is added uniformly to each pixel and Poisson noise is independent at each pixel given the clean pixel value. In order to extract useful information from noisy micrographs, postprocessing steps such as image denoising are often employed.

Although there exists a huge body of literature aimed at addressing image denoising in general, the field of microscopy image denoising has recently taken rapid strides [36–42] with the advent of Deep Learning (DL). The goal of this chapter is to introduce existing state-of-the-art DL based methods for denoising images corrupted with pixel noises, highlight their limitations and propose new methods to overcome these limitations.

2.1 The Denoising Task and Models of Imaging Noise

Let us begin by formalizing the task of image denoising. An image $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ is the noisy version of a clean image (signal) $\mathbf{s} = (\mathbf{s}_1, \ldots, \mathbf{s}_n)$ where $\mathbf{x}_i$ and $\mathbf{s}_i$ refer to the respective

---

Parts of this chapter are taken from the publication *Prakash, Lalit, Tomancak, Krull, Jug*, IEEE ISBI 2020 and *Krull, Vičar, Prakash, Lait, Jug*, Frontiers in Computer Science 2020.

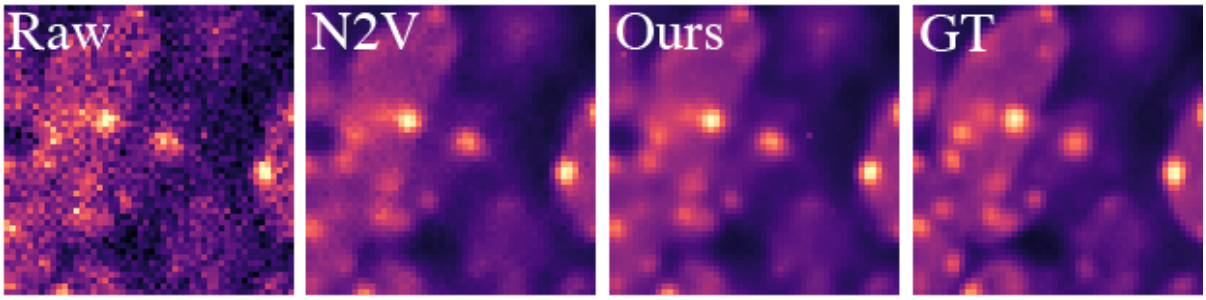

Figure 2.1: **Fully unsupervised PN2V with GMM based noise model for pixel-wise noise removal.** Our proposed GMM bootstrapping approach does not require paired training or calibration data, but achieves superior results compared to other fully unsupervised methods.

pixel intensities. Image denoising seeks to recover the original signal $\mathbf{s}$ from $\mathbf{x}$. The noisy $\mathbf{x}$ is thought to be drawn from a probability distribution $p(\mathbf{x}|\mathbf{s})$. As mentioned earlier, the focus of this chapter is on denoising images that suffer from insufficient illumination (*i.e.*, Poisson noise) and detector/camera imperfections (*i.e.*, Gaussian noise). For these noises, $p(\mathbf{x}|\mathbf{s})$ is usually thought to factorize as a product of pixels [37], implying that the corruption, given the underlying signal, is occurring independently in each pixel as

$$p(\mathbf{x}|\mathbf{s}) = \prod_{i=1}^{n} p(\mathbf{x}_i|\mathbf{s}_i). \tag{2.1}$$

We will refer to the probability $p(\mathbf{x}_i|\mathbf{s}_i)$ of observing a particular noisy value $\mathbf{x}_i$ at a pixel i given clean signal $\mathbf{s}_i$ as the *observation likelihood* or *pixel noise model*. As we will see later, this noise model can usually be computed from a simple stack of calibration images. It does not depend on the content of the images that are to be denoised. We will also show later that this noise model may even be bootstrapped from noisy images themselves.

In what follows, we will briefly review the most prominent existing denoising methods in Section 2.2 and discuss their limitations in Section 2.3. Section 2.4 will introduce our modifications to an existing state-of-the-art denoising method which addresses some of these limitations. In Section 2.5, we will present experimental results of our proposed methods and finally we close this chapter with a discussion of our results in Section 2.6.

2.2 Related Work

Classical Denoising Methods. To compute the denoised estimate for a particular pixel, many popular classical methods employ the idea of weighted averaging of other pixels in the image. For instance, bilateral filtering [43] performs such averaging in every pixel's local neighborhood. The weights for different pixels in the neighborhood are determined by their distances in image and

color space to the pixel being denoised. Another popular algorithm called Non-Local Means (NLM) [44] computes each pixel's estimate as an average of multiple similar pixels located anywhere in the image. Pixel weights in this case are determined based on the similarity of surrounding image patches. Arguably, the most popular classical denoising method is block-matching and 3D filtering (BM3D) [45]. Building on the idea of NLM, BM3D groups similar patches of the image in a 3D volume before applying filtering. These filtered patches are then projected back into the original image. A more detailed exposition of classical denoising methods is covered in [46]. All these methods do not require training data of any kind. In the past few years, classical denoising methods have been outperformed by deep learning based systems [36, 47–49].

Deep Learning Based Denoising. The field of image denoising has taken rapid strides with the advent of Deep Learning (DL). DL methods have surpassed the classical denoising methods and are currently the best performing ones [40, 47, 48, 50–53]. These methods, unlike classical denoising methods, are *content aware*, *i.e.*, they learn a strong prior on the visual nature of the data to be reconstructed [36, 37, 40, 42, 50, 52, 54, 55]. The core idea behind all these methods is that they learn, from training data, a function which takes a noisy image $\mathbf{x}$ and produces an image $\hat{\mathbf{s}}$ such that $\hat{\mathbf{s}} \approx \mathbf{s}$.

Recently, DL methods based on Convolutional Neural Network (CNN) [56] architecture have seen remarkable success for image denoising (see *e.g.* [36, 50, 52, 54]). In this setup every predicted output pixel $\hat{\mathbf{s}}_i$ depends only on a limited receptive field $\mathbf{x}_{\mathrm{RF}(i)}$, *i.e.*, a patch of input pixels surrounding it. CNN based image denoising in fact produces independent predictions $\hat{\mathbf{s}}_i = g(\mathbf{x}_{\mathrm{RF}(i)}; \boldsymbol{\theta}) \approx \mathbf{s}_i$ for each pixel i, depending only on $\mathbf{x}_{\mathrm{RF}(i)}$ instead of on the entire image. The prediction is parametrized by the weights $\boldsymbol{\theta}$ of the network.

Supervised DL based methods. In traditional supervised training [36, 50], $\boldsymbol{\theta}$ are learnt from pairs of noisy $\mathbf{x}^j$ and corresponding clean training images $\mathbf{s}^j$. $(\mathbf{x}^j_{\mathrm{RF}(i)}, \mathbf{s}^j_i)$ consisting of noisy input patches $\mathbf{x}^j_{\mathrm{RF}(i)}$ and their corresponding clean target values $\mathbf{s}^j_i$. The parameters $\boldsymbol{\theta}$ are traditionally tuned to minimize an empirical risk function such as the average squared distance

$$\arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \sum_{j=1}^{m} (\hat{\mathbf{s}}^j_i - \mathbf{s}^j_i)^2 \tag{2.2}$$

over all training images j and pixels i. The need for paired high quality and low quality data is very restrictive as it is impractical or at times impossible to obtain clean ground truth images. For instance, in live cell imaging in microscopy, such pairs cannot be obtained because the samples being imaged are dynamic and hence obtaining a perfectly registered noisy and clean ground truth image of the same scene is virtually impossible. Additionally, even if the sample being imaged is not very dynamic, acquiring a high quality image may not be possible owing to phototoxicity considerations.

To circumvent this requirement, NOISE2NOISE [54] training has been proposed which relaxes the problem somewhat by only needing two noisy image instances of the same scene. NOISE2NOISE uses pairs of corresponding noisy training images $\mathbf{x}^j$ and $\mathbf{x}'^j$, which are based on the same signal $\mathbf{s}^j$, but are corrupted independently by noise (see Equation 2.1). Such pairs can for example be acquired by imaging a static sample twice. NOISE2NOISE uses training examples $(\mathbf{x}^j_{\mathrm{RF}(i)}, \mathbf{x}'^j_i)$, with the input patch $\mathbf{x}^j_{\mathrm{RF}(i)}$ cropped from the first image $\mathbf{x}^j$ and the noisy target $\mathbf{x}'^j_i$ extracted from the patch center in the second one $\mathbf{x}'$. It is of course impossible for the network to predict the noisy pixel value $\mathbf{x}'^j_i$ from the independently corrupted input $\mathbf{x}^j_{\mathrm{RF}(i)}$. However, assuming the noise is zero centered (such as Gaussian noise and Poisson noise), *i.e.*, $\mathbb{E}\left[\mathbf{x}^j_i\right] = \mathbf{s}^j_i$, the best achievable prediction is the clean signal $\mathbf{s}^j_i$ and the network will learn to denoise the images it is presented with.

NOISE2NOISE can lead to virtually the same results as those obtained with traditional supervised methods trained with high quality and low quality image pairs. But unfortunately, even acquiring two noisy copies of the same image content is not possible in many applications such as cryo-electron microscopy [38].

Unsupervised DL based methods. Contrary to supervised methods, unsupervised DL based methods offer a promising alternative and show how even the requirement for a second noisy image can be avoided. These methods can train directly on the body of data to be denoised, making them extremely useful for practical applications.

The pioneering work by Ulyanov *et al.* [57] truly pushed the boundaries of DL based image restoration and ushered the era of unsupervised DL based image denoising. The authors discovered that the structure of CNNs "resonates" with natural images and is inherently suitable for the task of denoising and image restoration. Their method, *Deep Image Prior* (DIP), trains a network to predict a single noisy image from a constant random input while using the noisy data itself as target. They find that when the training process is stopped at a suitable time, the network can produce a noise free version of the given image. However, the training needs to be done separately for each noisy input image in the training set, making this approach computationally rather expensive. Furthermore, training has to be stopped after a suitable but a priori unknown number of training steps.

Krull *et al.* [40] and Batson *et al.* [55] concurrently proposed NOISE2VOID (N2V) and Noise2Self respectively to train CNNs for blind image denoising by excluding/masking the center (blind-spot) of the network's receptive fields. Unlike DIP, these methods have the advantage that they can be trained with all noisy images at the same time and the training does not have to be terminated manually at some intermediate time. These methods assume that the noise is pixel-wise independent (conditioned on clean signal) and that the true intensity of a pixel can be predicted from local image context, excluding before-mentioned blind-spots. For many applications, especially in the context of microscopy images, the first assumption is fulfilled, but the second assumption offers room for improvements [42]. Hence, these self-supervised methods are known to perform less well than models trained using paired training data [40, 41]. More

recently, a number of other unsupervised methods [42, 53, 58–62] have been proposed to bridge the gap between unsupervised and supervised methods.

Of all these methods, a recent work called Probabilistic NOISE2VOID (PN2V) [53] which builds on N2V is the most relevant for us in the context of this chapter. In what follows, we elaborate on N2V and PN2V in greater details to highlight their strengths and limitations.

NOISE2VOID. In NOISE2VOID, Krull *et al.* [40] use single noisy images to extract input and target for their networks. If this was done naively, the network would simply learn the identity transformation, directly outputting the value at the center of each pixel's receptive field. Krull *et al.*address the issue by effectively removing the central pixel from the network's receptive field. To achieve this, they mask the pixel during training, replacing it with a random value from the vicinity. Thus, a NOISE2VOID trained network can be seen as a function $\hat{\mathbf{s}}_i = \tilde{g}(\tilde{\mathbf{x}}_{\mathrm{RF}(i)}; \boldsymbol{\theta}) \approx \mathbf{s}_i$, making a prediction for a single pixel based on the modified patch $\tilde{\mathbf{x}}_{\mathrm{RF}(i)}$ that excludes the central pixel. Such a network can no longer describe the identity, and can be trained from single noisy images.

However, this ability comes at price. The accuracy of the predictions is reduced, as the network has to exclude the central pixel of its receptive field, thus having less information available. To allow efficient training of a CNN with NOISE2VOID, Krull *et al.*simultaneously mask multiple pixels in larger training patches and jointly calculate their gradients.

Probabilistic Noise2Void (PN2V). PN2V builds on the idea of masking pixels [40] to obtain a prediction from the modified receptive field $\tilde{\mathbf{x}}_{\mathrm{RF}(i)}$. However, instead of directly predicting an estimate for each pixel value, PN2V trains a CNN to describe a probability distribution

$$p(\mathbf{s}_i | \tilde{\mathbf{x}}_{\mathrm{RF}(i)}; \boldsymbol{\theta}), \tag{2.3}$$

referred to as *prior*, as it describes our knowledge of the pixel's signal considering only its surroundings, but not the observation at the pixel itself $\mathbf{x}_i$, since it has been excluded from $\tilde{\mathbf{x}}_{\mathrm{RF}(i)}$.

Remembering that the observed pixel's values are drawn independently (Equation 2.1) according to the noise model, we can combine Equation 2.3 with the noise model, and obtain the joint distribution

$$p(\mathbf{x}_i, \mathbf{s}_i | \tilde{\mathbf{x}}_{\mathrm{RF}(i)}; \boldsymbol{\theta}) = p(\mathbf{s}_i | \tilde{\mathbf{x}}_{\mathrm{RF}(i)}; \boldsymbol{\theta}) p(\mathbf{x}_i | \mathbf{s}_i). \tag{2.4}$$

By integrating over all possible clean signals, we have

$$p(\mathbf{x}_i | \tilde{\mathbf{x}}_{\mathrm{RF}(i)}; \boldsymbol{\theta}) = \int_{-\infty}^{\infty} p(\mathbf{s}_i | \tilde{\mathbf{x}}_{\mathrm{RF}(i)}; \boldsymbol{\theta}) p(\mathbf{x}_i | \mathbf{s}_i) d\mathbf{s}_i, \tag{2.5}$$

the probability of observing the pixel value $\mathbf{x}_i$, given we know its surroundings $\tilde{\mathbf{x}}_{\mathrm{RF}(i)}$. The CNN training can now be seen as an unsupervised learning task. Following the maximum likelihood approach, $\boldsymbol{\theta}$ are tuned to minimize

$$\arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} -\ln\left(\int_{-\infty}^{\infty} p(\mathbf{s}_i|\tilde{\mathbf{x}}_{\mathrm{RF}(i)};\boldsymbol{\theta}) p(\mathbf{x}_i|\mathbf{s}_i) d\mathbf{s}_i\right). \tag{2.6}$$

Note that in order to improve readability, we from here on omit the index j, and refrain from explicitly referring to the training image.

A sample based representation for prior is chosen for efficient optimization, *i.e.*, for every pixel i, PN2V directly predicts K output values $\mathbf{s}_i^k$, which are interpreted as independent samples, drawn from $p(\mathbf{s}_i|\tilde{\mathbf{x}}_{\mathrm{RF}(i)};\boldsymbol{\theta})$. We can now approximate Equation 2.6 as

$$\arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} -\ln\left(\frac{1}{K}\sum_{k=1}^{K} p(\mathbf{x}_i|\mathbf{s}_i^k)\right), \tag{2.7}$$

which forms the training loss for PN2V.

During inference, the MMSE estimate for any pixel i is computed as

$$\mathbf{s}_i^{\mathrm{MMSE}} = \arg\min_{\hat{\mathbf{s}}_i} \mathbb{E}_{p(\mathbf{s}_i|\mathbf{x}_{\mathrm{RF}(i)})}\left[(\hat{\mathbf{s}}_i - \mathbf{s}_i)^2\right] = \mathbb{E}_{p(\mathbf{s}_i|\mathbf{x}_{\mathrm{RF}(i)})}\left[\mathbf{s}_i\right], \tag{2.8}$$

where $p(\mathbf{s}_i|\mathbf{x}_{\mathrm{RF}(i)})$ is the *posterior* distribution of the signal given the complete surrounding patch. The posterior is proportional to the joint distribution given in Equation 2.4. $\mathbf{s}_i^{\mathrm{MMSE}}$ is approximated by weighing the predicted samples with the corresponding observation likelihood and calculating their average

$$\mathbf{s}_i^{\mathrm{MMSE}} \approx \frac{\sum_{k=1}^{K} p(\mathbf{x}_i|\mathbf{s}_i^k)\mathbf{s}_i^k}{\sum_{k=1}^{K} p(\mathbf{x}_i|\mathbf{s}_i^k)}. \tag{2.9}$$

2.3 Limitations of PN2V

The key component of PN2V is the noise model $p(\mathbf{x}_i|\mathbf{s}_i)$ which characterizes the distribution of noisy pixel $\mathbf{x}_i$ around their respective ground truth signal value $\mathbf{s}_i$. In the context of PN2V, noise models are described as a collection of histograms.

Histogram based noise models are built from a stack of calibration images $\mathbf{x}^1, \ldots, \mathbf{x}^m$. The imaged structures in this sequence can be arbitrary but must be static. Such images can, for example, be recorded by imaging the back illuminated half opened field diaphragm (see Figure 2.2). In order to minimize the effects of vibrations and sample drift, it is recommended to acquire calibration data in defocus. We call the average signal $\mathbf{s} = \frac{1}{m}\sum_{j=1}^{m} \mathbf{x}^j$ ground truth (GT). It is an established protocol to average multiple static but noisy acquisitions to obtain a

corresponding GT image [37]. By discretizing each GT pixel signal s_i and corresponding noisy observations x_i^j, a histogram can be created for each GT signal covering all corresponding noisy observations. The normalized set of histograms constitutes the camera noise model used in PN2V [41], describing the distribution of noisy pixel values $p(\mathbf{x}_i|\mathbf{s}_i)$ that are to be expected for each GT signal.

Histogram based noise models suffer from some practical limitations. Firstly, bin width plays an important role while constructing histogram based noise models. A small bin width may result in a noisy histogram whereas a large bin width may hide information about the variability in the distribution and may potentially mask the true underlying distribution. There do not exist guidelines on how to choose the appropriate bin width for any dataset for constructing a histogram based noise model in PN2V. Secondly, it is important that the calibration data covers all signal intensities which we may expect to be present in the data to be denoised. If this is not the case, then the noise model for some signal intensity $\mathbf{s}_i$ may not exist (See Figure 2.4(d)). Additionally, histogram based noise models may fail to mimic the true underlying noise distribution in cases where only a small amount of calibration data is available or if the calibration data is not adequately acquired (See Figure 2.4(c)). Hence, acquiring good quality calibration data is essential.

Acquiring calibration data can be tricky. Although acquisition of calibration images is a straightforward procedure for most light microscopy setups, it requires additional time and manual effort nonetheless. Moreover, if we wish to denoise images for which we may not have information about the microscope with which they were acquired, we cannot acquire calibration data on that microscope and hence, a noise model can simply not be constructed. For such case, PN2V training is not an option.

2.4 Proposed Approaches and Methods

Here we address the limitations of PN2V discussed in the previous section. We make two important contributions. Our first contribution is the introduction of parametric noise models based on Gaussian Mixture Models (GMM). We later demonstrate in Section 2.5 that GMM based noise models solve the problems associated with histogram based noise models and hence lead to superior denoising performance when used in the PN2V framework. Secondly, we show how suitable noise models can be created even in the absence of calibration data. This is a major step since it actually renders PN2V fully unsupervised.

GMM based noise models describe the distribution of noisy observations $\mathbf{x}_i$ for a GT signal $\mathbf{s}_i$ as the weighted average of L normal distributions:

$$p(\mathbf{x}_i|\mathbf{s}_i) = \sum_{l=1}^{L} \alpha_l(\mathbf{s}_i) f\big(\mu_l(\mathbf{s}_i), \sigma_l^2(\mathbf{s}_i)\big), \tag{2.10}$$

where $f\big(\mu_l(\mathbf{s}_i), \sigma_l^2(\mathbf{s}_i)\big)$ is the probability density function of the the normal distribution. We define each component's weight $\alpha_l(\mathbf{s}_i)$, mean $\mu_l(\mathbf{s}_i)$, and variance $\sigma_l^2(\mathbf{s}_i)$ as a function of the signal $\mathbf{s}_i$. To ensure all weights are positive and sum to one we define

$$\alpha_l(\mathbf{s}_i) = \exp\big(g_l^\alpha(\mathbf{s}_i)\big) / \sum_{l'=1}^{L} \exp\big(g_{l'}^\alpha(\mathbf{s}_i)\big), \tag{2.11}$$

where $g_{l'}^\alpha(\mathbf{s}_i)$ is a polynomial of degree d. To ensure that our distributions are always centered around the true signal $\mathbf{s}_i$, we define

$$\mu_l(\mathbf{s}_i) = \mathbf{s}_i + g_l^\mu(\mathbf{s}_i) - \sum_{l'=1}^{L} \alpha_{l'}(\mathbf{s}_i) g_{l'}^\mu(\mathbf{s}_i), \tag{2.12}$$

where $g_l^\mu(\mathbf{s}_i)$ is again a polynomial of degree d. Finally, to ensure numerical stability, we define the variance

$$\sigma_l^2(\mathbf{s}_i) = \max(g_l^\sigma(\mathbf{s}_i), c), \tag{2.13}$$

where $c = 50$ is a constant, and $g_l^\sigma(\mathbf{s}_i)$ is again a polynomial of degree d. Hence, our GMM based noise model is fully described by the $3 \times L \times d$ long vector of polynomial coefficients $\boldsymbol{a}$. We use a maximum likelihood approach to fit the parameters to our calibration data, optimizing for

$$\arg\max_{\boldsymbol{a}} \sum_{i,j} \log p(\mathbf{x}_i^j | \mathbf{s}_i), \tag{2.14}$$

where $p(\mathbf{x}_i^j | \mathbf{s}_i)$, is the GMM as described in Equation 2.10. We use numerical optimization, see Section 2.5.

GMM based noise models are inherently smooth and hence, unlike histogram based noise models, can interpolate/extrapolate for missing signals in the calibration data (see Fig. Figure 2.4).

Bootstrapped PN2V. In order to address the scenarios where no calibration data is available, *e.g.*, data that was acquired without denoising in mind, we propose a bootstrapping procedure to obtain a noise model. First, we train and apply the unsupervised N2V [40] on the body of available noisy images $\mathbf{x}^j$. Then, we treat the resulting denoised images $\hat{\mathbf{s}}^j$ as if they were the GT, henceforth calling them pseudo GT. We can now use the corresponding noisy $\mathbf{x}_i^j$ and denoised $\hat{\mathbf{s}}_i^j$ pixel values to either construct a histogram or learn a GMM based noise model as described earlier.

2.5 Experiments and Results

Datasets. We acquired three datasets (Figure 2.2) which are made publicly available: (*i*) *Convallaria* data, available online as part of PN2V consisting of 100 calibration images (diaphragm

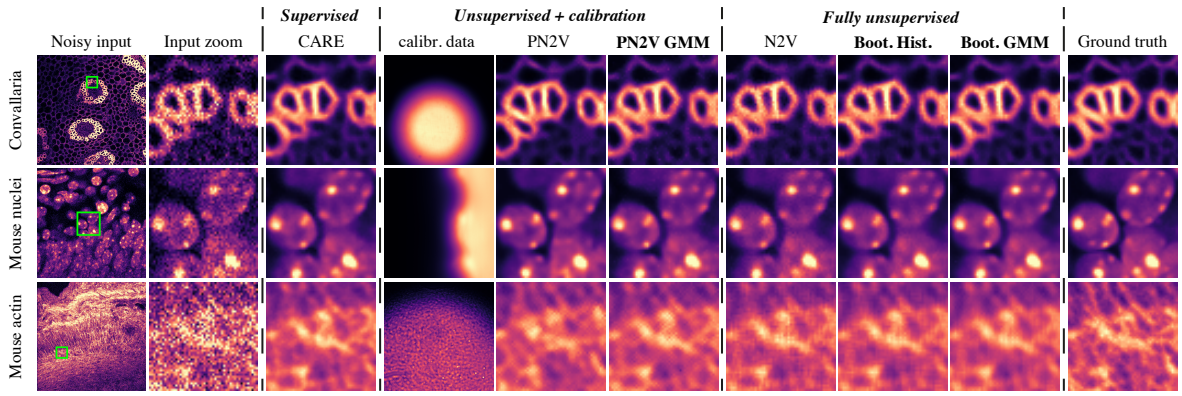

Figure 2.2: **A visual comparison of results obtained by CARE, N2V, PN2V, and our proposed methods**. We distinguish three families of methods: fully supervised (CARE), unsupervised but requiring additional calibration data (PN2V, our PN2V GMM), and fully unsupervised (N2V, PN2V using our bootstrapped histogram and GMM based noise models). The leftmost column in the unsupervised+calibration category shows the average of all available calibration images used for PN2V and PN2V GMM (see main text). Note that results of our fully unsupervised methods reach very similar quality to methods requiring either clean GT, or additional calibration data.

images, as previously explained) and 100 noisy images of a *Convallaria* section, (*ii*) mouse skull nuclei dataset consisting of 500 calibration images (showing the edge of a fluorescent slide) and 200 noisy realizations of the same static mouse skull nuclei, and (*iii*) mouse actin data consisting of 100 calibration images (diaphragm images with only the sample mounting medium in field of view) and 100 noisy realizations of the same static actin sample. The *Convallaria* and mouse actin datasets are acquired on a spinning disc confocal microscope while the mouse skull nuclei dataset is acquired with a point scanning confocal microscope.

Implementation and training details. All evaluated training schemes are based on the implementation from [41] and use the same network architecture: a U-Net [63] with depth 3, 1 input channel, and 64 feature channels in the first layer. All networks are trained with ADAM [64] with initial learning rate of 0.001, a patch size of 100, a batch size of 1, a virtual batch size of 20 and the standard learning rate scheduler as used in [41]. Training is done for 200 epochs, each consisting of 5 steps. We use the N2V and CARE (traditional supervised training) implementations from [41] as baselines.

With PN2V, we will refer to the version trained with the original histogram based noise model, derived form the available calibration data and we will use this as additional baseline. As in [53], for each dataset, we use a $B \times B$ bin discretization, where B is an integer determined in an empirically optimal manner for which the denoising performance (PSNR) of histogram based PN2V is maximized. The minimum and maximum bins are set to the minimum and maximum values present in the data to be denoised.

Methods	*Convallaria*	Mouse nuclei	Mouse actin
CARE	**36.71±0.026**	**36.58±0.019**	**34.20±0.021**
PN2V	**36.51±0.025**	36.29±0.007	33.78±0.006
PN2V GMM	36.47±0.031	**36.35±0.018**	**33.86±0.018**
N2V	35.73±0.037	35.84±0.015	33.39±0.014
Boot. Hist.	36.19±0.016	36.31±0.013	33.61±0.016
Boot. GMM	**36.70±0.012**	**36.43±0.014**	**33.74±0.012**

TABLE 2.1: **Comparision of the denoising performance of all tested methods.** Mean PSNR and ±1 standard error over five repetitions of each experiment are shown. Names of our proposed methods are shown in bold. Bold numbers indicate the best performing method in its respective category (supervised, unsupervised + calibration, and fully unsupervised; from top to bottom, separated by dashed lines).

Gaussians	Two coefficients	Three coefficients
1	36.56±0.022	36.34±0.040
2	36.48±0.020	36.35±0.014
3	36.47±0.031	36.31±0.022

TABLE 2.2: **Testing a variety of GMM hyper-parameters.** We tested GMMs using one, two, and three Gaussians, each using linear ($d = 2$) and quadratic ($d = 3$) parametrizations (see Section 2.4), to denoise the *Convallaria* data. The table always shows the mean PSNR and standard error over 5 repetitions.

Whenever we use our proposed GMM noise model, we will label results with PN2V GMM. As long as not stated differently, all GMM noise models use $L = 3$ Gaussians and $d = 2$ coefficients per parameter, and are trained on the available calibration data. Starting from a random initialization, optimization is performed using ADAM with learning rate 0.1, using a batch size of 25000 and 4000 iterations for mouse skull nuclei and mouse actin datasets, and a batch size of 250000 and 2000 iterations for the *Convallaria* data.

For bootstrapped PN2V (histogram and GMM based), we use the same setup as for PN2V but naturally taking the bootstrapped noise models instead. They are referred to as Boot. Hist. and Boot. GMM respectively. For the latter, we disregard the top and bottom 0.5% percentile of the pseudo GT pixels during noise model training, as we empirically observe that their N2V predictions can be often unreliable.

All datasets, results, and code are publicly available [1].

[1] github.com/juglab/ppn2v

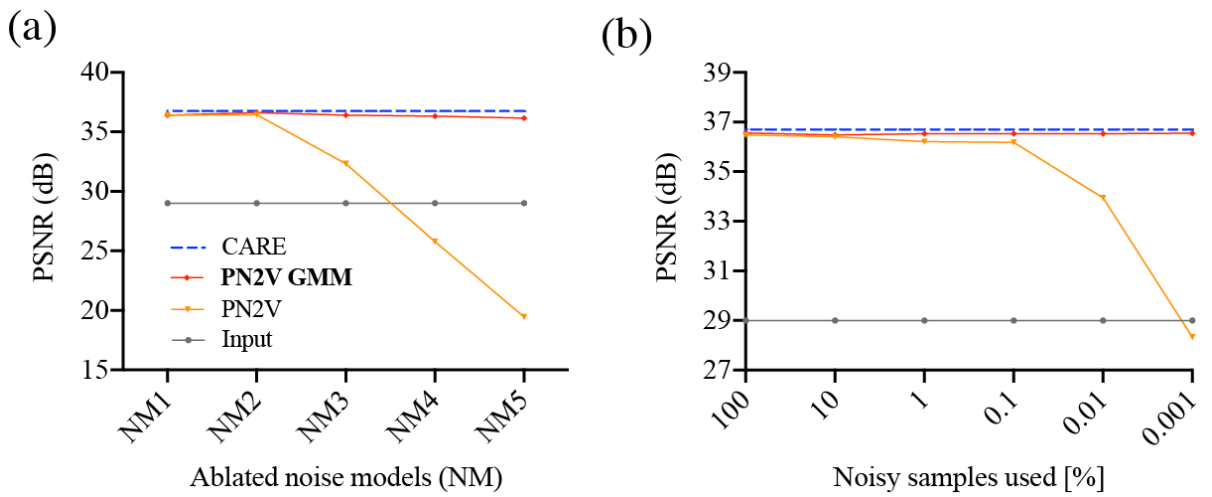

Figure 2.3: **Ablation studies on *Convallaria* data**. Denoising performance of PN2V with histogram and linear GMM noise models is shown. (a) The five noise models we tested are deduced from subsets of the available calibration data, such that: the entire range of signals used in the *Convallaria* data is covered (NM1), only the lower 40% are covered (NM2), the lower 25% (NM3), 15% (NM4) and 9% (NM5). (b) This case is obtained by reducing the fraction of available noisy calibration pixels from NM1, via random subsampling.

Gaussians	NM1	NM2	NM3	NM4	NM5
1	36.56	36.03	35.98	35.85	35.78
3	36.47	36.58	36.37	36.20	36.08

TABLE 2.3: **Denoising performance of PN2V GMM with linear noise models using one versus three Gaussians.** For each case, five noise models were derived from different subsets of the available calibration data (see Figure 2.3). We report the mean PSNR over 5 repetitions for each setup.

Comparing different training schemes. For each dataset and denoising method, we repeated each experiment 5 times and then compared the denoised images in terms of peak-signal-to-noise-ratio (PSNR) to available GT images. Results can be seen in Figure 2.2, as well as Table 2.1. We also evaluated the structural similarity (SSIM) score for all datasets and made them available at `github.com/juglab/ppn2v/wiki`.

Naturally, the fully supervised CARE networks, trained on clean ground truth images, show the best performance on all datasets. On the mouse actin dataset, PN2V using our GMM based noise model derived from high quality calibration data outperforms all other methods. Notably,

on the other two datasets, our fully unsupervised bootstrapped approach provides superior results and is remarkably close to CARE. For a discussion of these results, see Section 2.6.

Ablation and parameter study. Next we compare the robustness of histogram and GMM based noise models with respect to increasingly imperfect calibration data, using the *Convallaria* dataset as an example. These *ablation studies* consist of two scenarios, where (*i*) the available calibration data covers less and less of the range of signals in the data to be denoised, and (*ii*) the amount of available calibration pixels decreases successively. Figure 2.4 (c,d) shows example noise models that are derived from ablated calibration data. Evidently, for both ablation tests PN2V GMM performance is more robust compared to PN2V (see Figure 2.3).

We also investigated the sensitivity of GMM noise models with respect to the chosen hyper parameters. We performed a parameter study, varying the number of Gaussian kernels L and polynomial coefficients d, using the *Convallaria* dataset with full available calibration data. Results are summarized in Table 2.2. While these tests suggest that the simple linear model (one Gaussian, two coefficients) is slightly preferable, the performance of all configurations remains superior to N2V (see Table 2.1). We additionally measured the performance of a linear noise model using 1 Gaussian and 3 Gaussians with imperfect calibration data (see Table 2.3). We observe that a noise model with 3 Gaussians leads to more stable results.

2.6 Discussion

We presented a GMM based variation of PN2V noise models and showed that they can achieve higher reconstruction quality even with imperfect calibration data (Figure 2.3). Additionally, we introduced a novel bootstrapping scheme, which allows PN2V to be trained fully unsupervised using only the data to be denoised (Figure 2.4(b)). Our results (Table 2.1) show that the denoising quality of bootstrapped PN2V is quite close to fully supervised CARE [36] and significantly outperforms N2V [40]. Hence, if calibration data for a given microscope is unavailable, bootstrapping offers an excellent alternative.

Interestingly, at times, bootstrapped GMM based noise models even outperform models derived from calibration data. A possible reason for such good performance is that the distribution of pseudo GT signals used in bootstrapping corresponds well to the distribution of signals in the data to be denoised. The distribution of GT signals in the calibration data however, can be quite different.

GMM noise models, trained according to Equation 2.14, prioritize signals that are abundant in the (pseudo) GT and provide a better fit in these regions compared to others. Figure 2.4(b) corroborates that our bootstrapped GMM fits well to the true noise distribution for lower signals, which frequently occur in the *Convallaria* data, but fails for higher signals. However, the GMM trained on calibration data (Figure 2.4(a)), prioritizes its fit for higher signals, which are frequent in the calibration data, but barely present in the *Convallaria* dataset.

We strongly believe that the methods we propose will help to make high quality DL based denoising an easily applicable tool that does not require the acquisition of paired training data or calibration data. This would facilitate a plethora of projects in cell biology, where the processes

to be imaged are very photosensitive or so dynamic that suitable training image pairs cannot be obtained.

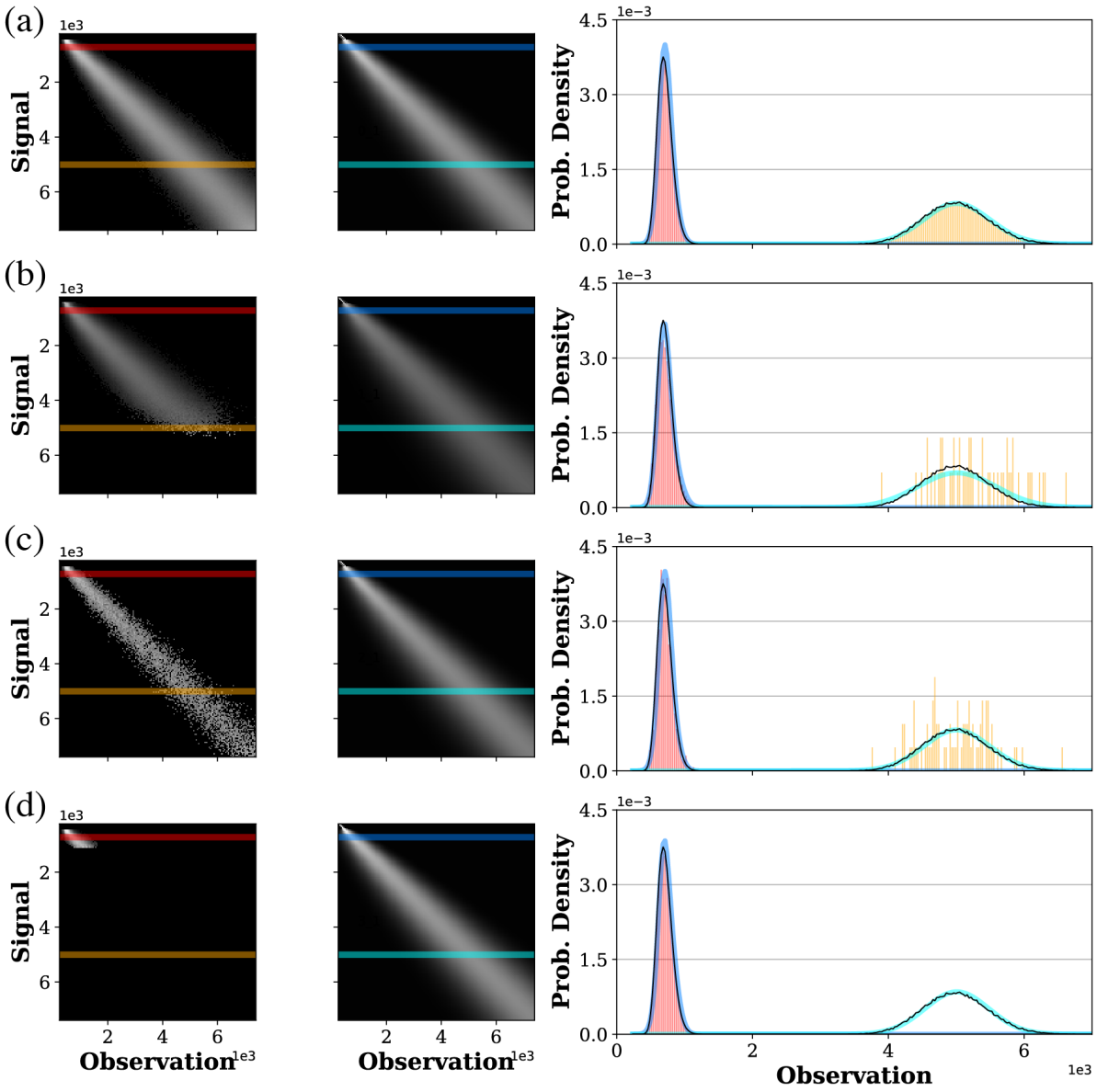

Figure 2.4: **Noise models for *Convallaria* data**. Left column shows histogram based noise models, the center column their respective GMM based equivalent. Rightmost column shows noise models for specific signals (colors, histograms shown as vertical lines). For comparison, full calibration data histogram is always included as black curve. (a) Noise models trained on full calibration data. (b) Bootstrapped noise models. (c) Noise models trained on sub-sampled (0.1%) calibration data (Figure 2.3b). (d) Noise models trained on reduced available calibration data (NM5 from Figure 2.3a).

I-B
Unsupervised Diversity Denoising

CHAPTER 3

FULLY UNSUPERVISED DIVERSITY DENOISING WITH CONVOLUTIONAL AUTOENCODERS

Noisy images are ambiguous owing to the fact that distortions degrade some of the information content in images. This makes it impossible to fully recover the desired clean signal with certainty. Hence, image denoising an ill-posed inverse problem which admits potentially many solutions and even an ideal method cannot know which of many possible clean images really has given rise to the degraded observation at hand.

All existing denoising approaches introduced in Chapter 2 have a common flaw. These methods, depending on their objective function, make a compromise between possible solutions when predicting a restored image and only provide a single denoised solution. For instance, in case of methods employing Minimum Mean Squared Error (MMSE) estimators such as fully supervised CARE [36] or unsupervised methods such as Noise2Void [40] introduced in Chapter 2, this compromise is the solution which minimizes the expected quadratic error.

Generating multiple plausible diverse denoising solutions is of much importance for ambiguity removal and uncertainty quantification. For instance, in noisy microscopy images with high cell density, a denoising solution may join the boundary between two nearby/touching cells. This will introduce errors in downstream applications such as segmentation and cell counting. Having multiple diverse denoising solutions will allow biomedical practitioners to make correct analyses.

Generative models, such as VAEs, are a canonical choice when a distribution over a set of variables needs to be learned. Still, so far VAEs have been overlooked as a method to solve unsupervised image denoising problems. This might also be due to the fact that vanilla VAEs [65, 66] show sub-par performance on denoising problems (see Section 3.5).

In this chapter, we look at the problem of diversity denoising and introduce DivNoising (DN), a principled approach to incorporate explicit models of the imaging noise distribution in the decoder of a VAE. Such noise models can be either measured or derived (bootstrapped) from the noisy image data alone [53, 58] as discussed in Chapter 2. Additionally here we propose a way to co-learn a suitable noise model during training, rendering DN fully unsupervised. We show on 13 datasets that fully convolutional VAEs, trained with our proposed DN framework, yield competitive results, in 8 cases actually becoming the new state-of-the-art (see Figure 3.4 and Table 3.1). Still, the key benefit of DN is that this method does not need to commit to a single prediction, but is instead capable of generating diverse samples from an approximate posterior of possible true signals. (Note that point estimates can still be inferred if desired, as shown in Figure 3.5.) Other unsupervised denoising methods only provide a single solution (point estimate) of that posterior [40, 54, 55] or predict an independent posterior distribution

---

Parts of this chapter is taken from the published article *Prakash, Krull, Jug*, ICLR 2021.

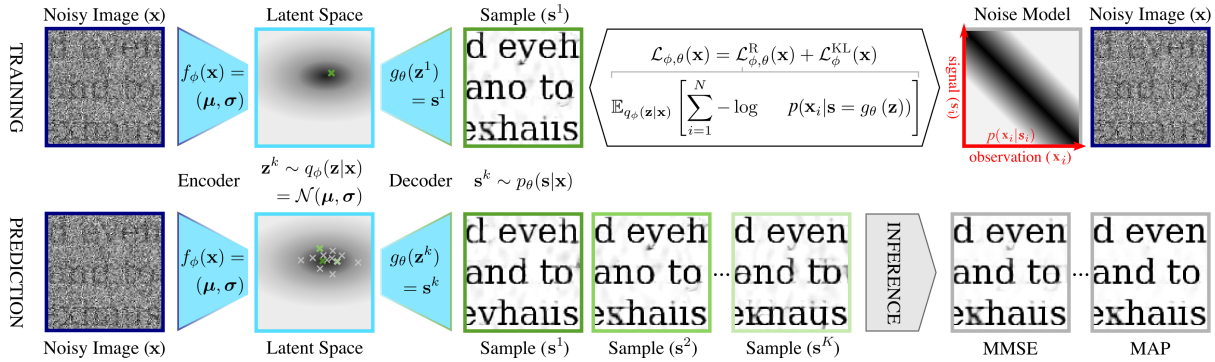

Figure 3.1: **Training and prediction/inference with DivNoising.** ***(top)*** A DivNoising VAE can be trained fully unsupervised, using only noisy data and a (measured, bootstrapped, or co-learned) pixel noise model $p(\mathbf{x}_i|\mathbf{s}_i)$ (see main text for details). ***(bottom)*** After training, the encoder can be used to sample multiple $\mathbf{z}^k \sim q_\phi(\mathbf{z}|\mathbf{x})$, giving rise to diverse denoised samples $\mathbf{s}^k$. These samples can further be used to infer consensus point estimates such as a MMSE or a MAP solution.

of intensities per pixel such as PN2V [53, 58] introduced in Chapter 2. Hence, DN is the first method that learns to approximate the posterior over meaningful structures in a given body of images.

Finally, we discuss the utility of diverse denoising results for OCR and showcase it for a ubiquitous analysis task in biology – the instance segmentation of cells in microscopy images (see Figure 3.6). Hence, DN has the potential to be useful for many real-world applications and will not only generate state-of-the-art (SOTA) restored images, but also enrich quantitative downstream processing.

3.1 Related Work

We discussed most of the relevant literature in the context of pixel wise noise removal in Chapter 2. Here we discuss some additional works relevant particularly in the context of this chapter.

Self2Self. Recently, [59] proposed an interesting method called Self2Self which trains a U-Net like architecture requiring only single noisy images. The key idea of this approach is to use blind spot masking, similar to Noise2Void [40], together with *dropout* [67], which avoids overfitting and allows sampling of diverse solutions. Similar to DN, the single denoised result is obtained by averaging many diverse predictions. Diverse results obtained via dropout are generally considered to capture the so called *epistemic* or *model uncertainty* [68, 69], *i.e.*the uncertainty arising from the fact that we have a limited amount of training data available. In contrast, DN combines a VAE and a model of the imaging noise to capture what is known

as *aleatoric* or *data uncertainty* [70, 71], *i.e.*the unavoidable uncertainty about the true signal resulting from noisy measurements. Like in [57], also Self2Self trains separately on each image that has to be denoised. While this renders the method universally applicable, it is computationally prohibitive when applied to large datasets. The same is true for real time applications such as facial denoising. DN, on the other hand, is trained only once on a given body of data. Afterwards, it can be efficiently applied to new images.

Denoising (Variational) Autoencoders. Despite the suggestive name, *denoising variational autoencoders* [72] are not solving denoising problems. Instead, this method proposes to add noise to the input data in order to boost the quality of encoder distributions. This, in turn, can lead to stronger generative models. Other methods also follow a similar approach to improve overall performance of autoencoders [73–75].

VAEs for Diverse Solution Sampling. Although not explored in the context of unsupervised denoising, VAEs are designed to sample diverse solutions from trained posteriors. The probabilistic U-Net [76, 77] uses conditional VAEs to learn a conditional distribution over segmentations. [78] improve the diversity of segmentation samples by introducing a hierarchy of latent variables to model segmentations at multiple resolutions. Unlike DN, both methods rely on paired training data. [79] employ VAEs to learn the distribution of incomplete and heterogeneous data in a fully unsupervised manner. [80] build upon a VAE style framework to predict multiple plausible future frames of videos conditioned on given context frames. A variational inference approach was used by [81] to generate multiple deprojected samples for images and videos collapsed in either spatial or temporal dimensions. Unlike all these approaches, we address the uncertainty introduced by common imaging noise and show how denoised samples can improve downstream processing.

3.2 The Denoising Task

Here we briefly recapitulate the denoising task introduced in Chapter 2. As discussed in Chapter 2, image restoration is the task of estimating a clean signal $\mathbf{s} = (\mathbf{s}_1, \ldots, \mathbf{s}_n)$ from a corrupted observation $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, where $\mathbf{s}_i$ and $\mathbf{x}_i$, refer to the respective pixel intensities. The corrupted $\mathbf{x}$ is thought to be drawn from a probability distribution $p(\mathbf{x}|\mathbf{s})$, which we call the *observation likelihood* or the *noise model*. Similar to Chapter 2, the scope of this chapter is restricted to restoring images that suffer from insufficient illumination (*i.e.*Poisson noise) and detector/camera imperfections (*e.g.*Gaussian or impulse noises).

Contrary to existing methods, DN is designed to capture the inherent uncertainty of the denoising problem by learning a suitable posterior distribution. Formally, the posterior we are interested in is $p(\mathbf{s}|\mathbf{x}) \propto p(\mathbf{x}|\mathbf{s})p(\mathbf{s})$ and depends on two components: the *prior* distribution $p(\mathbf{s})$ of the signal as well as the observation likelihood $p(\mathbf{x}|\mathbf{s})$. While the prior is a highly complex distribution, the likelihood $p(\mathbf{x}|\mathbf{s})$ of a given imaging system (camera/microscope) can be described analytically [53].

Again similar to Chapter 2, the noise model is thought to factorize as a product of pixels, implying that the corruption, given the underlying signal, is occurring independently in each pixel as

$$p(\mathbf{x}|\mathbf{s}) = \prod_{i}^{N} p(\mathbf{x}_i|\mathbf{s}_i). \tag{3.1}$$

In this work, we follow the Gaussian Mixture Model (GMM) based noise model description as introduced in Chapter 2, the parameters of which can be estimated whenever pairs $(\mathbf{x}', \mathbf{s}')$ of corresponding noisy and clean calibration images are available or can be estimated by a bootstrapping approach [58] in case where no calibration data can be acquired,. In this chapter, we additionally show how a suitable noise model can be co-learned during training of DN from noisy images themselves.

3.3 The Variational Autoencoder (VAE) Setup

Here, we want to give a very brief overview of the VAE approach introduced by Kingma *et al.*. in [65]. A more complete discussion can be found in [82]. VAEs are generative models, capable of learning complex distributions over classes of images $\mathbf{x}$, such as hand written digits [65] or faces [83]. To achieve this, VAEs use a latent variable $\mathbf{z}$ and describe

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z}. \tag{3.2}$$

Like conventional autoencoders, they consist of two components: an *encoder* network $f_\phi(\mathbf{x})$, which takes an observed image and maps it to the latent space, and a *decoder* network $g_\theta(\mathbf{z})$ that implements the inverse operation. By ϕ and θ we denote network parameters of the encoder and decoder, respectively. Next, we will take a closer look at the model from Equation 3.2.

The Generative Model. In VAEs the latent variable $\mathbf{z}$ is defined to follow a multivariate unit normal distribution $p(\mathbf{z})$. The decoder describes the conditional probability $p_\theta(\mathbf{x}|\mathbf{z})$ of observing $\mathbf{x}$, given the latent variable $\mathbf{z}$. It is assumed to factorize as

$$p_\theta(\mathbf{x}|\mathbf{z}) = \prod_{i=1}^{n} p_\theta(\mathbf{x}_i|\mathbf{z}), \tag{3.3}$$

where $p_\theta(\mathbf{x}_i|\mathbf{z})$ is the distribution of a pixel intensity given the latent vector. It is often modelled as a normal distribution, with the parameters for each pixel i predicted by the decoder network. Together with $p(\mathbf{z})$, our decoder completely describes the model in Equation 3.2.

Training. The goal is to find decoder parameters, such that Equation 3.2 well describes the distribution of training images $\mathbf{x}^1 \dots \mathbf{x}^M$. To enable this training task, the VAE relies on its

encoder. The encoder describes a distribution $q_\phi(\mathbf{z}|\mathbf{x})$, aiming to approximate the distribution of the latent variable given the image, as it is implicitly defined by the decoder

$$q_\phi(\mathbf{z}|\mathbf{x}) \approx p_\theta(\mathbf{z}|\mathbf{x}) = \frac{p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{\int p_\theta(\mathbf{x}|\mathbf{z}=\mathbf{z}')p(\mathbf{z}=\mathbf{z}')d\mathbf{z}'}. \tag{3.4}$$

The encoder describes its distribution as a product over the D-dimensional latent variable

$$q_\phi(\mathbf{z}|\mathbf{x}) = \prod_{i=1}^{D} q_\phi(\mathbf{z}_i|\mathbf{x}), \tag{3.5}$$

with $q_\phi(\mathbf{z}_i|\mathbf{x}) = \mathcal{N}(\mu_i, \sigma_i)$ denoting the distribution for an element i of the latent space. The means and standard deviations for each element are predicted by the encoder network $f_\phi(\mathbf{x}) = (\boldsymbol{\mu}, \boldsymbol{\sigma}) = (\mu_1 \dots \mu_D, \sigma_1 \dots \sigma_D)$. Based on this setup, both networks are trained jointly by optimizing the loss

$$\mathscr{L}_{\phi,\theta}(\mathbf{x}) = \mathscr{L}^{\mathrm{R}}_{\phi,\theta}(\mathbf{x}) + \mathscr{L}^{\mathrm{KL}}_{\phi}(\mathbf{x}), \tag{3.6}$$

where

$$\mathscr{L}^{\mathrm{R}}_{\phi,\theta}(\mathbf{x}) = \mathbb{E}\left[-\log p_\theta(\mathbf{x}|\mathbf{z})\right] q_\phi(\mathbf{z}|\mathbf{x}) = \mathbb{E}\left[\sum_{i=1}^{n} -\log p_\theta(\mathbf{x}_i|\mathbf{z})\right] q_\phi(\mathbf{z}|\mathbf{x}), \tag{3.7}$$

and $\mathscr{L}^{\mathrm{KL}}_{\phi}(\mathbf{x})$ is the KL divergence $\mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right)$. While $\mathscr{L}^{\mathrm{KL}}_{\phi}(\mathbf{x})$ can be computed analytically, the expected value in Equation 3.7 is approximated by drawing a single sample $\mathbf{z}^1$ from $q_\phi(\mathbf{z}|\mathbf{x})$ and using the *reparametrization trick* [65] for gradient computation. It has been shown in [65] that minimizing Equation 3.6 simultaneously achieves two goals: (*i*) the model defined by the decoder well describes the distribution of the observed data, and (*ii*) the encoder gives a good approximation of Equation 3.4.

Image Generation. Once trained, images from $p_\theta(\mathbf{x})$ can be generated by drawing samples $\mathbf{z}^k \sim p(\mathbf{z})$ from the normal distribution in latent space and processing them with the decoder to sample the pixel values from $\mathbf{x}_i^k \sim p_\theta(\mathbf{x}_i|\mathbf{z}^k)$.

Fully Convolutional Architecture. Originally, VAEs were using fully connected network architectures, or a mixture of convolutional and fully connected layers. Such architectures can only process fixed size inputs. However, recently the use of fully convolutional VAE architectures was proposed [84]. In our work, we also use a fully convolutional architecture (see Section 3.5), enabling us to process arbitrarily sized input images.

3.4 DivNoising (DN)

In DN, we build on the VAE setup but interpret it from a denoising-specific perspective. We assume that images have been created from a clean signal $\mathbf{s}$ via a known noise model, i.e., $\mathbf{x} \sim p(\mathbf{x}|\mathbf{s})$. To account for this within the VAE setup, we replace the generic normal distribution over pixel intensities in Equation 3.3 with a known noise model $p(\mathbf{x}|\mathbf{s})$ (see Equation 3.1). We

get $p_\theta(\mathbf{x}|\mathbf{z}) = p(\mathbf{x}|\mathbf{s}) = \prod_i^n p(\mathbf{x}_i|\mathbf{s}_i)$, with the decoder now predicting the signal $g_\theta(\mathbf{z}) = \mathbf{s}$. Together with $p(\mathbf{z})$ and the noise model, the decoder now describes a full joint model for all three variables, including the signal:

$$p_\theta(\mathbf{z}, \mathbf{x}, \mathbf{s}) = p(\mathbf{x}|\mathbf{s})p_\theta(\mathbf{s}|\mathbf{z})p(\mathbf{z}), \tag{3.8}$$

where we assume that $p(\mathbf{x}|\mathbf{s}, \mathbf{z}) = p(\mathbf{x}|\mathbf{s})$. For a given $\mathbf{z}^k$, as for standard VAEs, the decoder describes a distribution over noisy images $p(\mathbf{x}|\mathbf{z})$. The corresponding clean signal $\mathbf{s}^k$, in contrast, is deterministically defined. Hence, $p_\theta(\mathbf{s}|\mathbf{z})$ is a Dirac distribution centered at $g_\theta(\mathbf{z})$.

Training. Considering Equation 3.1, the reconstruction loss becomes

$$\mathscr{L}^{\mathrm{R}}_{\phi,\theta}(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\sum_{i=1}^{n} -\log p(\mathbf{x}_i|\mathbf{s} = g_\theta(\mathbf{z}))\right]. \tag{3.9}$$

Apart from this modification, we can follow the standard VAE training procedure, just as described in Section 3.3. Since we have only modified how the decoder distribution is modeled, we can assume that the training procedure still produces (i) a model describing the distribution of our training data, while (ii) making sure that the encoder distribution well approximates the distribution of the latent variable given the image. A complete derivation of the DN loss (from probability model perspective) can be found in Appendix A.8.

Prediction. While we can use the trained VAE to generate images from $p_\theta(\mathbf{x})$, here we are mainly interested in denoising. Hence, we desire access to the posterior $p(\mathbf{s}|\mathbf{x})$, *i.e.*the distribution of possible clean signals $\mathbf{s}$ given a noisy observation $\mathbf{x}$. Assuming the encoder and decoder are sufficiently well trained, samples $\mathbf{s}^k$ from an approximate posterior can be obtained by (i) feeding the noisy image $\mathbf{x}$ into our encoder, (ii) drawing samples $\mathbf{z}^k \sim q_\phi(\mathbf{z}|\mathbf{x})$, and (iii) decoding the samples via the decoder to get $\mathbf{s}^k = g_\theta(\mathbf{z}^k)$.

Inference. Given a set of posterior samples $\mathbf{s}^k$ for a noisy image $\mathbf{x}$, we can infer different consensus estimates (point estimates). We can, for example, approximate the MMSE estimate (see Figure 3.4), by averaging many samples $\mathbf{s}^k$. Alternatively, we can attempt to find the *maximum a posteriori* (MAP) estimate, *i.e.*the most likely signal given the noisy observation $\mathbf{x}$, by finding the mode of the posterior distribution. For this purpose, we iteratively use the mean shift algorithm [85] with decreasing bandwidth to find the mode of our sample set (see Figure 3.5 and Appendix A.4).

Fully Unsupervised DivNoising. So far we explained our setup under the assumption that the noise model can either be measured with paired calibration images, or bootstrapped from noisy data [58] as described in Chapter 2. Here, we propose yet another alternative approach of co-learning the noise model directly from noisy data during training. More concretely, this is enabled by a simple modification to the DN decoder. We assume that the noise at each pixel

i follows a normal distribution with its variance being a linear function of $\mathbf{s}_i$, *i.e.*, $\sigma_i^2 = a\mathbf{s}_i + b$. Linearity is motivated by noise properties in low-light settings [86, 87]. The learnable network parameters a and b are co-optimized during training. Since variances cannot be negative, we additionally constrain the predicted values for σ_i^2 to be positive (see Appendix A.3 for details).

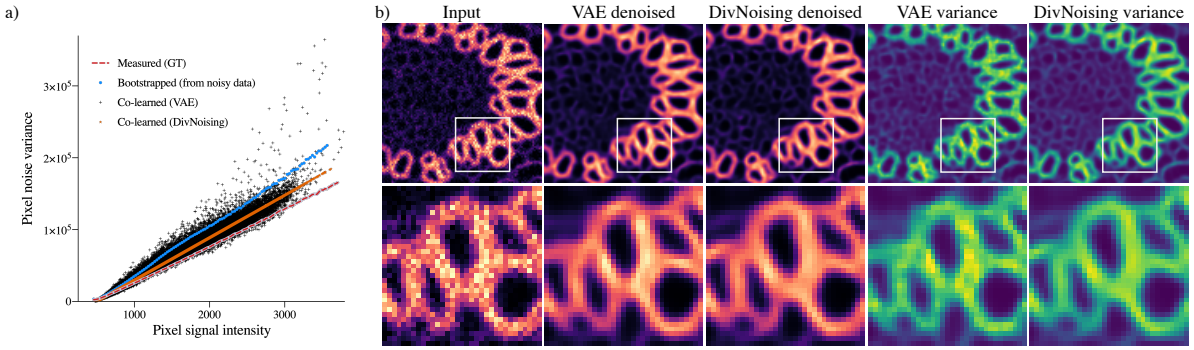

Figure 3.2: **Comparison of noise models and variance maps predicted by the vanilla VAE and DivNoising. (a)** For each predicted signal intensity (x-axis), we show the variance of noisy observations (y-axis). The plot is generated from experiments on the *Convallaria* dataset. The dashed red line shows the true noise distribution (measured from pairs of noisy and clean calibration data). We compare the noise model co-learned by the vanilla VAE and DivNoising with ground truth noise model and a noise model bootstrapped from noisy data alone as described in [58]. Clearly, the noise model learnt by unsupervised DivNoising is a much better approximation to the ground truth noise model compared to the noise models learned/obtained by other methods. **(b)** We visually compare the denoising results and show how the predicted variance varies across the image. As a consequence of the approximately linear relationship between signal and noise variance, both variance images closely resemble the denoised results. However, the result of the vanilla VAE additionally contains artefacts.

Denoising with Vanilla VAEs. While not originally intended for denoising tasks, we wish to see how vanilla VAEs perform when applied to these problems. Just like fully unsupervised DN, also the vanilla VAE does not require a noise model. It does, instead, directly predict per-pixel mean and variance (see Section 3.3), leaving the possibility open that the right values could be learned. However, here the decoder is not restricted to make each pixel's variance a function of

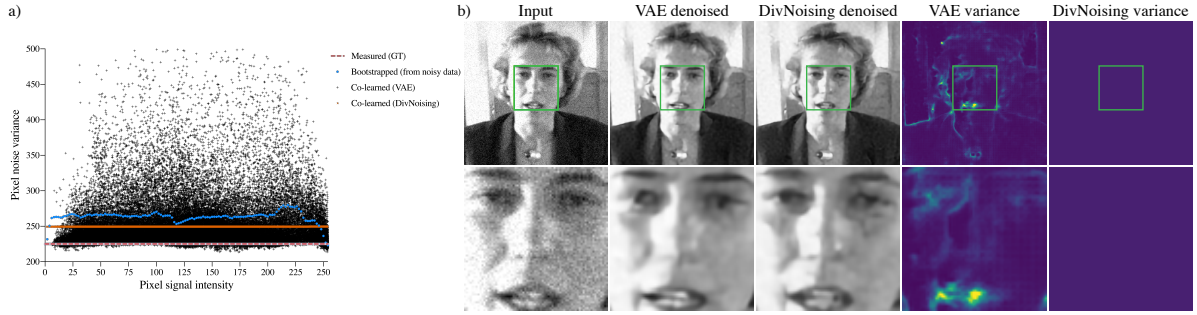

Figure 3.3: **Comparison of noise models and variance maps predicted by the vanilla VAE and DivNoising. (a)** For each predicted signal intensity (x-axis), we show the variance of noisy observations (y-axis). The plot is generated from experiments on the *BioID Face* dataset. The dashed red line shows the true noise distribution (Gaussian noise with $\sigma^2 = 225$). The noise model created via bootstrapping, and the noise model we co-learned with DivNoising, correctly show (approximately) constant values across all signal intensities. The implicitly learned noise model of the vanilla VAE has to independently predict the noise variance for each pixel. Its predictions clearly deviate from the true constant noise variance. **(b)** We visually compare the denoising results and show how the predicted variance varies across the image. While the variance predicted by the implicitly co-learned vanilla VAE model varies depending on the image content, the variance predicted by the co-learned DivNoising model correctly remains flat.

predicted signal. We investigate the denoising performance of the vanilla VAE in Section 3.5 and show in Figure 3.2 and Figure 3.3 that the predicted variances significantly diverge from ground truth noise distributions.

Signal Prior in DN. Classical denoising methods often explicitly model the image/signal prior $p(\mathbf{s})$ *e.g.*as smoothness priors [88, 89], non-local similarity priors [44, 45], sparseness priors [90] etc., assuming specific properties of the images at hand. They effectively assign the same probability to all images/signals sharing *e.g.*the same level of smoothness. However, the true distribution $p(\mathbf{s})$ of clean signals (*e.g.*for a particular experimental setup in a fluorescence microscope) is generally more complex. Instead of explicitly modelling $p(\mathbf{s})$, DN only implicitly describes $p_\theta(\mathbf{s}) = \int p_\theta(\mathbf{s}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$ as integral over all possible values of $\mathbf{z}$. We recall that the prior $p(\mathbf{z})$ is assumed to be the unit Gaussian distribution and the conditional distribution $p_\theta(\mathbf{s}|\mathbf{z})$ is learned by the decoder network as the Dirac distribution centered at $g_\theta(\mathbf{z})$. Depending on its parameters θ, the network will implement the function differently, leading to a different $p_\theta(\mathbf{s}|\mathbf{z})$, and ultimately to a different $p_\theta(\mathbf{s})$. This implicit distribution is quite powerful and can capture complex structures.

		Fully Unsupervised			Unsup. (p_{NM} requ.)		Supervised
	Dataset	N2V	Vanilla VAE	DN	PN2V	DN	CARE
FU-PN2V	Convallaria	35.73	36.57	*36.78*	36.47	**36.90**	36.71
	↳ Bootstrapped				**36.70**	36.64	
	Mouse Act.	33.39	33.46	*33.82*	33.86	**33.99**	34.20
	Mouse Nuc.	35.84	35.84	*36.05*	**36.35**	36.26	36.58
W2S	Ch.0 (avg1)	***34.59***	33.02	34.24	-	34.13	35.22
	Ch.1 (avg1)	32.11	31.36	***32.22***	-	**32.22**	32.88
	Ch.2 (avg1)	35.04	33.72	***35.24***	32.79	35.18	35.91
	Ch.0 (avg16)	39.01	39.27	*39.45*	39.36	**39.63**	42.35
	Ch.1 (avg16)	37.91	38.33	*38.41*	**38.46**	38.39	39.64
	Ch.2 (avg16)	40.30	40.24	***40.56***	40.36	40.41	42.03
DenoiSeg	Mouse	33.84	*34.06*	*34.06*	**34.19**	34.13	35.11
	Flywing	24.79	24.88	*24.92*	24.85	**25.02**	25.79
	Mouse s&p	32.98	23.62	*35.19*	29.67	**36.21**	37.03
	BioID Face	32.34	32.58	*33.02*	**33.76**	33.12	35.06

TABLE 3.1: **Quantitative results.** For all experiments, we compare all results in terms of mean Peak Signal-to-Noise Ratio (PSNR in dB) over 5 runs. Overall best performance indicated by being underlined, best unsupervised method in bold, and best fully unsupervised method in italic. For many datasets, DivNoising is the unsupervised SOTA, typically not being far behind the supervised CARE results.

3.5 Data, Experiments, Results

We quantitatively evaluated the performance of DN on 13 publicly available datasets (see Appendices A.1 and A.2 for data details), 9 of which are subject to high levels of intrinsic (real world) noise. To 4 others we synthetically added noise, hence giving us full knowledge about the nature of the added noise.

Denoising Baselines. We choose state-of-the-art baseline methods to compare DN against, namely, supervised CARE [36] and unsupervised methods Noise2Void (N2V) [40] and Probabilistic Noise2Void (PN2V) [40]. All baselines use the available implementations of [53] and, as long as not specified otherwise, make use of a depth 3 U-Net with 1 input channel and 64 channels in the first layer. As an additional baseline, we choose vanilla VAEs with the same network architecture as DN, but predicting per pixel mean and variance independently. Training is performed using the ADAM [64] optimizer for 200 epochs with 10 steps per epoch with a batch size of 4 and a virtual batch size of 20 for N2V and CARE and a batch size of 1 and a virtual batch size of 20 for PN2V, an initial learning rate of 0.001, and the same basic learning rate scheduler as in [53]. All baselines use on the fly data augmentation (flipping and rotation) during training.

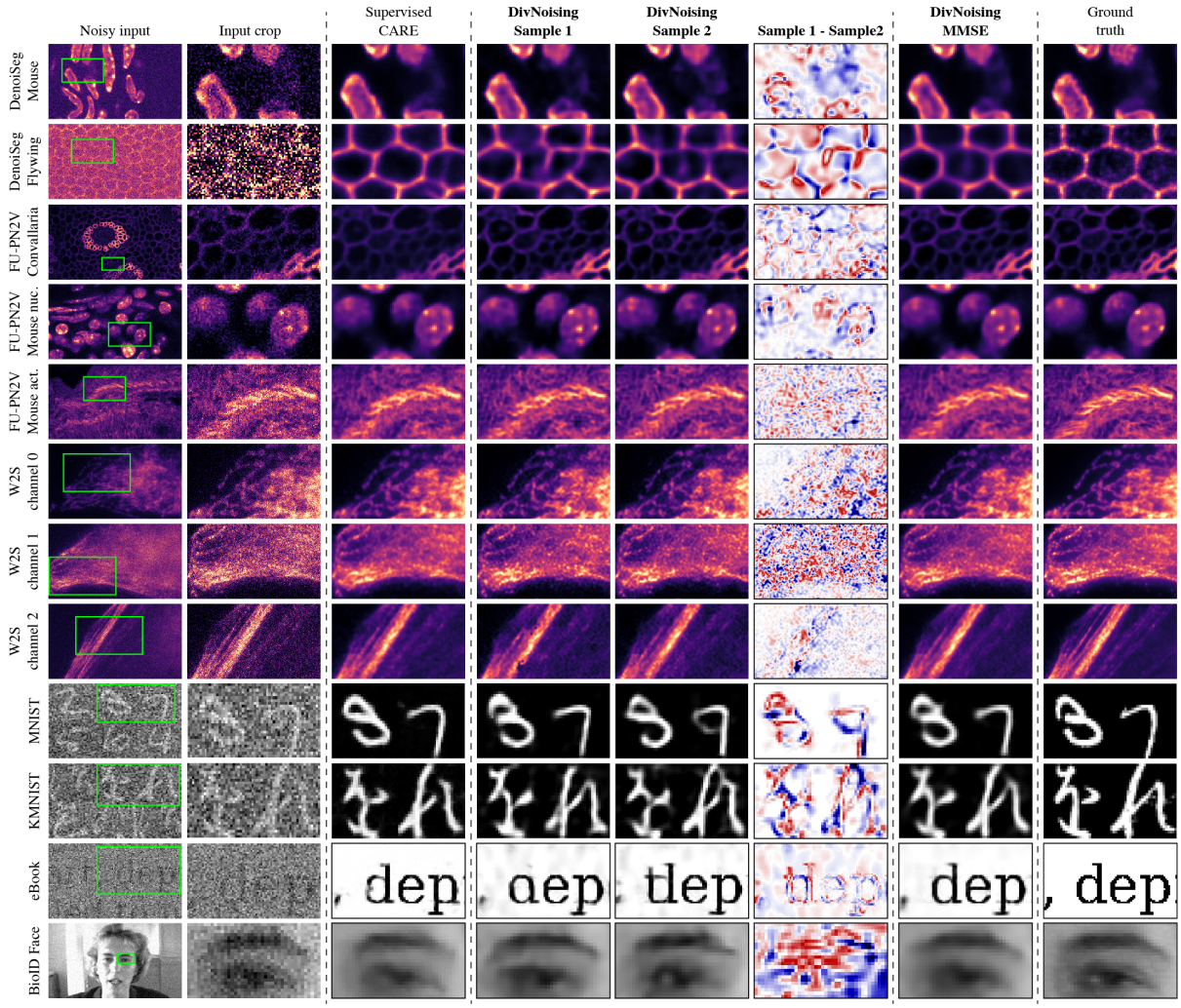

Figure 3.4: **Qualitative denoising results.** We compare two DivNoising samples, the MMSE estimate (derived by averaging 1000 sampled images), and results by the supervised CARE baseline. The diversity between individual samples is visualized in the column of difference images.

Training Details. In all experiments we use rather small, fully convolutional VAE networks, with either about 200k or 713k parameters (see Appendix A.3). For all experiments on intrinsically noisy microscopy data, validation and test set splits follow the ones described in the respective publication. In contrast to the synthetically noisy data, no apriori noise model is known for microscopy datasets. For these datasets, we used GMM-based noise models [58, 61],

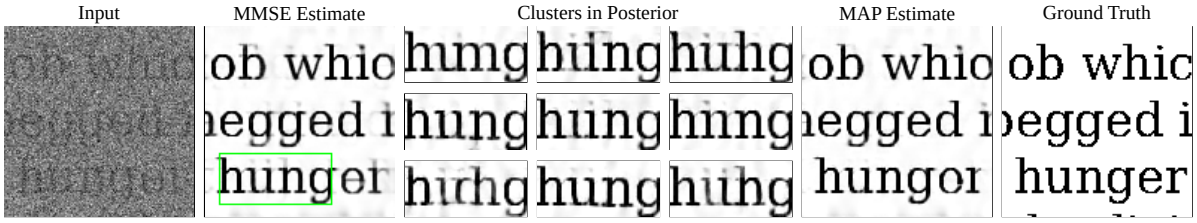

Figure 3.5: **Exploring the learned posterior.** The MMSE estimate (average of 10k samples) shows faintly overlaid letters as a consequence of ambiguities in noisy input. Among these samples from the posterior, we use mean shift clustering (on smaller crops) to identify diverse and likely points in the posterior. We show 9 such cluster centers in no particular order. We also obtain an approximate MAP estimate (see Appendix A.4), which has most artefacts of the MMSE solution removed.

which are measured from calibration images, as well as co-learned noise models. For the *W2S* datasets, no dedicated calibration samples to create noise models are available. Hence, for this dataset, we use the available clean ground truth images and all noisy observations of the training data to learn a GMM-based noise model. All GMM noise models use 3 Gaussians and 2 coefficients each. Find more training details in Appendix A.3. The code is publicly available[1].

Denoising Results. In Table 3.1, we report denoising performance of all experiments we conducted in terms of peak signal-to-noise ratio (PSNR) with respect to available ground truth images. The DN results (using the MMSE estimate from 1000 averaged samples) are typically either on par or even beyond the denoising quality reached by the baselines in the 'fully unsupervised' category, as well as the 'unsupervised with noise model' category.

Note that sampling is very efficient. For all presented experiments sampling 1000 images consistently took less than 7 seconds.

DN MMSE is typically, as expected, slightly behind the performance of the fully supervised baseline CARE [36]. Additionally, on *FU-PN2V Convallaria* we have demonstrated that a suitable noise model for DN can be created via bootstrapping [58, 61]. We also compare against Deep Image Prior (DIP) on *DenoiSeg Flywing* dataset as it has smallest number of test images and DIP has to be trained for each image. DIP achieves PSNR of 24.67dB compared to 25.02dB with DN.

The performance on the natural image benchmark dataset BSD68 citeroth2005fields is discussed in Appendix A.7. For natural image datasets, we find that the performance of DN is not so good and it is slightly worse than Noise2Void.

[1] github.com/juglab/DivNoising

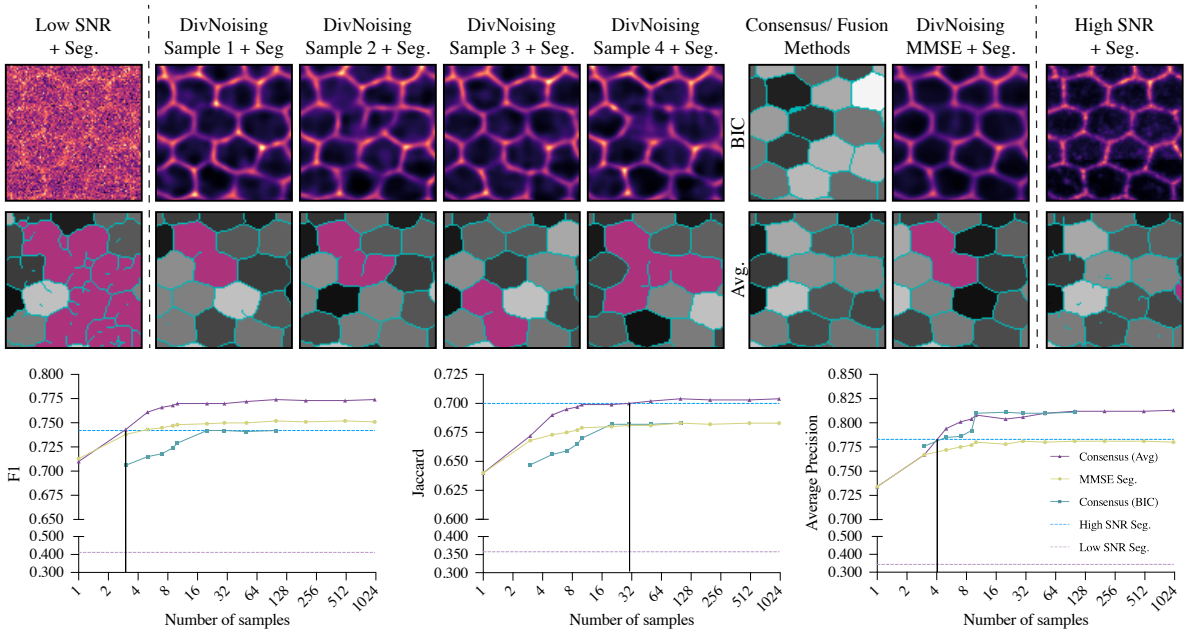

Figure 3.6: **DivNoising enables downstream segmentation tasks.** ***(top)*** We show various cell segmentation results as well as the corresponding denoised images they were produced from. Cells that were segmented incorrectly (merged or split) are indicated in magenta. Sampled DivNoising results give rise to diverse segmentation hypotheses. ***(bottom)*** We quantitatively evaluate the quality of the different segmentation results using the F1 score [91], Jaccard score [92] and Average Precision [93]. On the x-axis, we depict the number of samples/labels used by any given method. The performance of BIC is only evaluated up to 100 samples because of run time constraints (setting an upper time limit of 30 minutes). Remarkably, *Consensus (Avg)*, even using as little as 30 DivNoising segmentation labels, outperforms segmentations obtained from high SNR images on all available metrics.

Downstream Processing: OCR. In Figure 3.5 we show how Optical Character Recognition (OCR) applications might benefit from diverse denoising. While regular denoising approaches predict poor compromises that would never be seen in clean text, DN can generate a diverse set of rather plausible denoised solutions. While our MAP estimates clean up most such problems, occasional mistakes cannot be avoided, *e.g.*changing "hunger" to "hungor" (see Figure 3.5). Diverse denoising solutions obtained by clustering typically correspond to plausible alternative interpretations. It stands to reason that OCR systems can benefit from having access to diverse interpretations.

Downstream Processing: Instance Cell Segmentation. We demonstrate how diverse denoised images generated with DN can help to segment all cells in the *DenoiSeg Flywing* data. While methods to generate diverse segmentations do exist [76, 77], they require ground

truth segmentation labels during training. In contrast, we use a simple and fast downstream segmentation pipeline $c(\mathbf{x})$ based on local thresholding and skeletonization (see Appendix A.5 for details) and apply it to individual samples $(\mathbf{s}^1 \dots \mathbf{s}^K)$ predicted by DN to derive segmentations $(\mathbf{c}^1 \dots \mathbf{c}^K)$. We explore two label fusion methods to combine the individual results and obtain an improved segmentation. We do: (*i*) use *Consensus (BIC)* [94] and (*ii*) create a pixel-wise average of $(\mathbf{c}^1 \dots \mathbf{c}^K)$, followed by again applying our threshold based segmentation procedure on this average, calling it *Consensus (Avg)*.

For comparison, we also segment (*i*) the low SNR input images, (*ii*) the original high SNR images, and (*iii*) the MMSE solutions of DN. Figure 3.6 show all results of our instance segmentation experiments. It is important to note that segmentation from even a single DN prediction outperforms segmentations on the low SNR image data quite substantially. We observe that label fusion methods can, by utilizing multiple samples, outperform the MMSE estimate, with *Consensus (Avg)* giving the best overall results.

3.6 Discussion and Conclusion

We have introduced DivNoising (DN), a novel unsupervised denoising paradigm that allows us, for the first time, to generate diverse and plausible denoising solutions, sampled from a learned posterior. We have demonstrated that the quality of denoised images is highly competitive, typically outperforming the unsupervised state-of-the-art, and at times even improving on supervised results.[1]

DN uses a lightweight fully convolutional architecture. The success of Deep Image Prior [57] has shown that convolutional neural networks are inherently suitable for image denoising. The works by Yokota *et al.* [95] reinforce this idea and Tachella *et al.* [96] additionally hypothesize that a possible reason for the success of convolutional networks is their similarity to non-local patch based filtering techniques. However, the overall performance of DN is not merely a consequence of its convolutional architecture. We believe that the novel and explicit modeling of imaging noise in the decoder plays an essential role. This becomes evident when comparing our results to other convolutional baselines (including Deep Image Prior and fully convolutional VAEs), which do not perform as well as DN on any of the datasets we used. Additionally, we observe that incorrect noise models consistently lead to inferior results (see Appendix A.6).

We find that DN is suited particularly well for microscopy data or other applications on a limited image domain. In its current form it works less well on collections of natural images (see Appendix A.7). This might not be very surprising, as we are training a generative model for our image data and would not expect to be capturing the tremendously diverse domain of natural photographic images with the comparatively tiny networks used in our experiments (see Appendix A.3). For microscopy data, instead, the diversity between datasets can be huge. Images of the same type of sample, acquired using the same experimental setup, however, contain many resembling structures of lesser overall diversity (they are from a limited image domain).

---

[1] Supervised methods using perfect GT will outperform DN, but GT data is at times not perfect.

Nevertheless, the stunning results we achieve suggest that DN will also find application in other areas where low SNR limited domain image data has to be analyzed. Next to microscopy, we can think of astronomy, medical imaging, or limited domain natural images such as faces or street scenes. Additionally, follow up research will explore larger and improved network architectures, able to capture more complex DN posteriors on datasets covering larger image domains.

While we constrained ourselves to the standard per-pixel noise models in this paper, the DN approach could in principle also work with more sophisticated higher level image degradation models, as long as they can be probabilistically described. This might include diffraction, blur, or even compression and demosaicing artefacts.

Maybe most importantly, DN can not only produce competitive and diverse results, but these results can also be leveraged for downstream processing. We have seen that cell segmentation can be improved and that clustering our results provides us with meaningful alternative interpretations of the same data (see Figure 3.5). We believe that this is a highly promising direction for many applications, as it provides us with a way to account for the uncertainty introduced by the imaging process. We are looking forward to see how DN will be applied and extended by the community, showcasing the true potential and limitations of this approach.

CHAPTER 4

REMOVING PIXEL NOISES AND SPATIAL ARTEFACTS WITH GENERATIVE DIVERSITY DENOISING METHODS

In the previous chapter, we looked at our diversity denoising framework called DivNoising which provides multiple plausible diverse denoised solutions corresponding to any noisy image corrupted with pixel-noises. Additional to producing state-of-the-art (SOTA) results on many denoising benchmarks [97], DivNoising can generate diverse denoised solutions, giving users access to samples from a distribution of sensible denoising results.

There are two aspects of DivNoising that were not discussed in detail in the previous chapter.

(*i*) The most impressive results for DivNoising are, however, observed on limited image domains such as microscopy datasets and photographs of human faces. On richer domains, *e.g.*natural images, the expressivity of the employed generative models becomes a bottleneck [97].

(*ii*) Additionally, DivNoising was only explored in the context of pixel-wise noise removal (Gaussian and Poisson). However, images are often subjected to unwanted artefacts and structured noises[1] as well.

In this chapter, we build upon DivNoising presented in the previous chapter and introduce a more expressive architecture for diversity denoising which achieves excellent results for more complex natural image domains as well as on microscopy images. In addition, we investigate the structured noise and artefact removal capability of DivNoising and other VAE based generative models.

4.1 Introduction

From regular photographs to microscopy or medical images, all image data are subject to unwanted noise and artefacts. Often the most dominant sources of noise are pixel-noises such as Gaussian and Poisson noise [53]. We distinguish pixel-noises from structured noises, *i.e.*all signal corruptions or artefacts that affect groups of pixels in correlated ways.

Supervised image restoration methods based on Deep Learning (DL) can address pixel-wise and structured noises but require paired training data that allows the trained network to distinguish wanted signal from unwanted corruption patterns [36,38,50,54,98]. The applicability of unsupervised methods is much broader since they do not require paired training data [40, 42, 53, 55, 57, 59], but the downsides of all unsupervised methods introduced in Chapters 2 and 3 are that (*i*) most of them are only capable of removing pixel-noises, and (*ii*) they typically show weaker overall performance than their supervised alternatives.

Parts of this chapter is taken from the *Prakash, Delbracio, Milanfar, Jug*, ICLR 2022.

[1]We define structured noises in Section 4.1

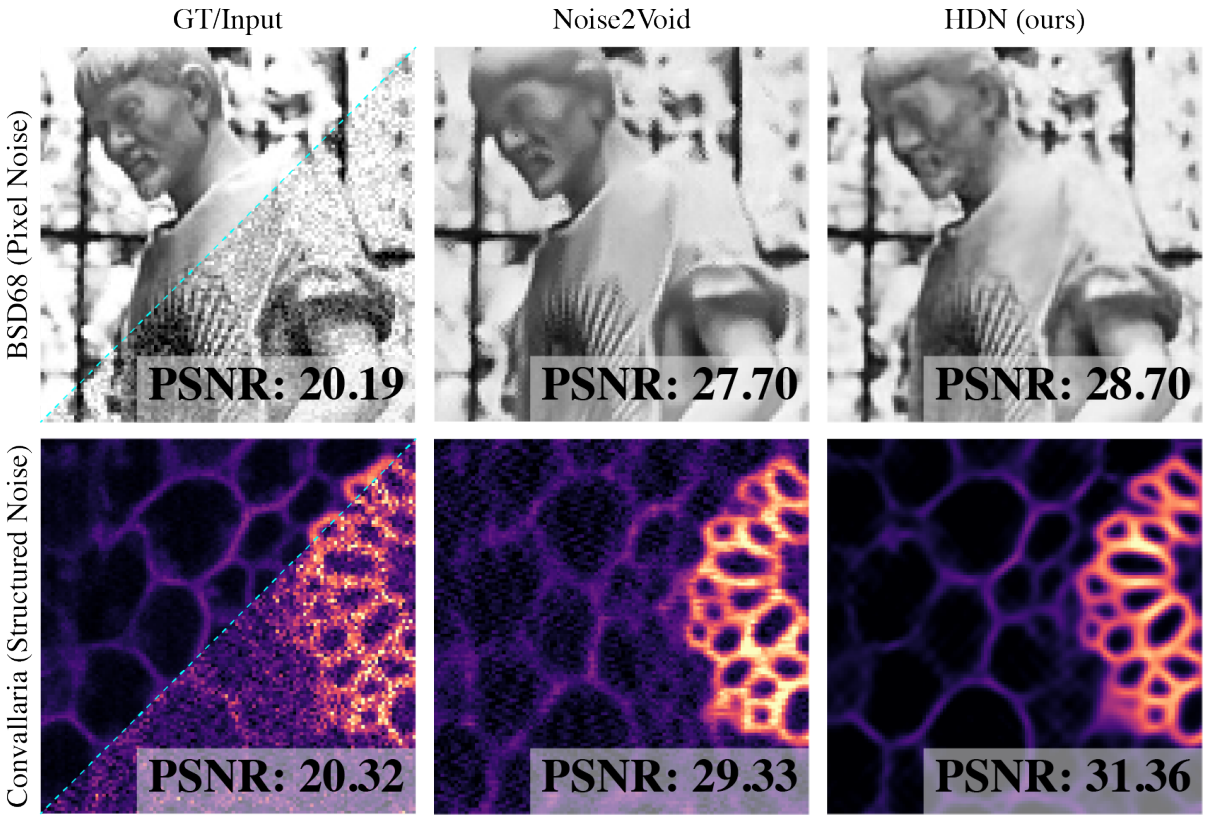

Figure 4.1: **Hierarchical DivNoising (HDN) for pixel-wise and structured noise removal.** We show ground truth and noisy input *(left)*, the result of a well known unsupervised baseline method *(middle)*, and the MMSE result obtained with Hierarchical DivNoising *(right)*. The *top row* is subject to pixel-noise, *i.e.*, Gaussian noise, while the *bottom row* is a real-world microscopy image subject to pixel-noise and spatially correlated structured noise (faint horizontal striping artefacts).

Generative Diversity Denoising (GDD) methods such as DivNoising introduced in Chapter 3 produce state-of-the-art (SOTA) results on many denoising benchmarks [97]. GDD methods can generate diverse denoised solutions, giving users access to samples from a distribution of sensible denoising results. The most impressive results are, so far, reported on limited image domains such as microscopy datasets and photographs of human faces. On richer domains, *e.g.*natural images, the expressivity of the employed generative models becomes a bottleneck [97]. Additionally, GDD methods are so far not used to remove structured noises.

In this chapter, we first introduce Hierarchical DivNoising (HDN), a new GDD method. HDN employs hierarchical VAEs and, as we show on eight publicly available benchmark datasets from diverse image domains, is the new SOTA method for pixel-noise removal.

We then study GDD methods based on Variational Autoencoders (VAEs) with respect to their ability to perform unsupervised structured noise removal. More specifically, we investigate if and how well VAE-GDD methods can learn to remove various sources of patterned artefacts from input images without supervision. Our experiments include simulated structured noises, real-world microscopy artefacts, and structured noise introduced during computed tomography (CT) reconstruction. We show, for the first time, that GDD methods can indeed be used to remove complex artefacts from noisy data. While this is known to not be possible with other unsupervised methods [40, 42, 55, 59], it immediately raises the question how GDD methods distinguish between structured noise, which should be removed, from true image signals, which need to be retained.

To provide some answers to this fundamental question, we perform a series of experiments and analyses, leading us to identifying several factors that contribute to successful artefact removal using GDD methods. We believe these analyses are important initial steps towards interpretable image restoration with GDDs.

4.2 The Image Restoration Task

Remember we introduced the image denoising task in Section 2.1 and Section 3.2. Here we will introduce the more general image restoration task since in this chapter we are dealing with not just pixel noise removal but artefacts and structured noise removal as well.

The task of image restoration involves the estimation of a clean and unobservable signal $\mathbf{s} = (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_n)$ from a corrupted reconstruction $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$, where $\mathbf{s}_i$ and $\mathbf{x}_i$, refer to the respective pixel intensities in the image domain. In general, the reconstructed image comes from solving an inverse imaging problem giving by the forward model,

$$\mathbf{y} = A(\mathbf{s}) + \mathbf{e}, \tag{4.1}$$

where A is the forward operator (tomography, blurring, sub-sampling, etc), $\mathbf{e}$ is noise in the measurements typically assumed iid, and $\mathbf{y}$ is the noisy measurements. An image reconstruction algorithm is needed to recover an estimation $\mathbf{x}$ of $\mathbf{s}$ from the noisy measurements $\mathbf{y}$. Typically, the image reconstruction is obtained through an optimization formulation, where we seek a solution $\mathbf{x}$ that fits the observed values and is compatible with some image prior R,

$$\mathbf{x} = \arg\min_{\mathbf{s}'} \|A(\mathbf{s}') - \mathbf{y}\|^2 + \lambda R(\mathbf{s}'). \tag{4.2}$$

The parameter $\lambda \geqslant 0$ is related to the level of confidence in the prior R. There exists an extensive amount of work defining image priors [57, 99–103].

Without loss of generality, we can decompose the reconstructed image as

$$\mathbf{x} = \mathbf{s} + \eta, \tag{4.3}$$

where η is the residual (noise) between the ideal image and the reconstructed one. Generally, the *noise* η on the reconstruction $\mathbf{x}$ is composed of pixel-noise (such as Poisson or Gaussian noise)

and multi-pixel artefacts or structured noise that affect groups of pixels in correlated ways. Such artefacts arise through dependencies that are introduced by the adopted reconstruction technique and the domain-specific inverse problem (*e.g.*tomography, microscopy, or ISP in an optical camera).

In Section 4.4 we make the assumption that the noise contribution η_i to each pixel $\mathbf{x}_i$ is conditionally independent given the signal $\mathbf{s}_i$, *i.e.*$p(\eta|\mathbf{s}) = \prod_i p(\eta_i|s_i)$ [40]. Note that this is the same assumption we made in chapters 2 and 3 and refers to the case of pixel-wise (Poisson and Gaussian) noise removal.

In many practical applications, including the ones presented in Section 4.5, this assumption does unfortunately not hold true, and the noise η is harder to formally grasp. This is equally true for supervised image restoration approaches, but by having access to paired image data with fixed $\mathbf{s}$ and changing η, a distinction between wanted and unwanted structures can be learned [38, 52, 54].

4.3 Generative Diversity Denoising (GDD)

GDD methods grant access to diverse restorations by means of providing samples s^k drawn from a posterior distribution of restored images. More formally, the restoration operation of a GDD model can be expressed by $f_\psi(\mathbf{x}) = s^k \sim p(\mathbf{s}|\mathbf{x})$, where f is a GDD model with parameters ψ. Here, we propose a new GDD method called Hierarchical DivNoising and compare it with two existing GDD methods, namely DivNoising [97] and vanilla VAEs [65, 66]. In the following we will first briefly recapitulate vanilla VAEs and DivNoising which we discussed in detail in Chapter 3 followed by an in-depth introduction to Hierarchical DivNoising.

Vanilla VAEs. VAEs are generative encoder-decoder models, capable of learning a latent representation of the data and capturing a distribution over inputs $\mathbf{x}$ [82,83]. The encoder maps input $\mathbf{x}$ to a conditional distribution $q_\phi(\mathbf{z}|\mathbf{x})$ in latent space. The decoder, $g_\theta(\mathbf{z})$, takes a sample from $q_\phi(\mathbf{z}|\mathbf{x})$ and maps it to a distribution $p_\theta(\mathbf{x}|\mathbf{z})$ in image space. Encoder and decoder are neural networks, jointly trained to minimize the loss

$$\mathscr{L}_{\phi,\theta}(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log p_\theta(\mathbf{x}|\mathbf{z})] + \mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right), \tag{4.4}$$

with the second term being the KL-divergence between the encoder distribution $q_\phi(\mathbf{z}|\mathbf{x})$ and prior distribution $p(\mathbf{z})$ (usually a unit normal distribution). The network parameters of the encoder and decoder are given by ϕ and θ, respectively. The decoder is usually modelled to factorize over pixels as

$$p_\theta(\mathbf{x}|\mathbf{z}) = \prod_{i=1}^{n} p_\theta(\mathbf{x}_i|\mathbf{z}), \tag{4.5}$$

with $p_\theta(\mathbf{x}_i|\mathbf{z})$ being a normal distribution predicted by the decoder. The encoder distribution is modeled in a similar way, factorizing over the dimensions of the latent space $\mathbf{z}$.

DivNoising (DN). DN [97], introduced in Chapter 3, is a VAE based method for unsupervised pixel noise removal that incorporates an explicit noise model $p(\mathbf{x}|\mathbf{s})$ in the decoder. More formally, the generic normal distribution over pixel intensities of Equation 4.5 is replaced with a known noise model $p(\mathbf{x}|\mathbf{s})$ which factorizes as a product of pixels, *i.e.*, $p(\mathbf{x}|\mathbf{s}) = \prod_i^n p(\mathbf{x}_i|\mathbf{s}_i)$, and the decoder learns a mapping from $\mathbf{z}$ directly to the space of restored images, *i.e.*, $g_\theta(\mathbf{z}) = \mathbf{s}$. Therefore,

$$p_\theta(\mathbf{x}|\mathbf{z}) = p(\mathbf{x}|g_\theta(\mathbf{z})). \tag{4.6}$$

The loss of DN hence becomes

$$\mathscr{L}_{\phi,\theta}(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\sum_{i=1}^{n} -\log p(\mathbf{x}_i|g_\theta(\mathbf{z}))\right] + \mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right). \tag{4.7}$$

DN is the current SOTA for many unsupervised denoising benchmarks, but can have difficulties to learn a good posterior distribution of clean data for complex domains such as natural images (see Appendix A.7).

Hierarchical DivNoising. One of our contributions in this chapter is a new GDD method based on hierarchical VAEs [104–107]. Hierarchical DivNoising (HDN) splits its latent representation over $h > 1$ hierarchically dependent stochastic latent variables $\mathbf{z} = \{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_h\}$. As in [107], $\mathbf{z}_1$ is the bottom-most latent variable, seeing the input essentially in unaltered form. The top-most latent variable is $\mathbf{z}_h$, receiving an $h-1$ times downsampled input and starting a cascade of latent variable conditioning back down the hierarchy. This architecture follows the description in [104,108], with the *bottom-up* and *top-down* networks having parameters ϕ and θ, respectively. The *top-down* network performs a downward pass to compute a hierarchical prior $p_\theta(\mathbf{z})$ which factorizes as

$$p_\theta(\mathbf{z}) = p_\theta(\mathbf{z}_h) \prod_{i=1}^{h-1} p_\theta(\mathbf{z}_i|\mathbf{z}_{j>i}), \tag{4.8}$$

with each factor $p_\theta(\mathbf{z}_i|\mathbf{z}_{j>i})$ being a learned multivariate Normal distribution with diagonal covariance, and $\mathbf{z}_{j>i}$ referring to all $\mathbf{z}_j$ with $j > i$. Note that even $p_\theta(\mathbf{z}_h)$, the prior of the top-most layer, is being learned during training.

Given a noisy input $\mathbf{x}$, the encoder distribution $q_\phi(\mathbf{z}|\mathbf{x})$ is computed in two steps. First the *bottom-up* network performs a deterministic upward pass and extracts representations from noisy input $\mathbf{x}$ at each layer. Next, the representations extracted by the *top-down* network during the downward pass are merged with results from the *bottom-up* network before inferring the conditional distribution

$$q_\phi(\mathbf{z}|\mathbf{x}) = q_\phi(\mathbf{z}_h|\mathbf{x}) \prod_{i}^{h-1} q_{\phi,\theta}(\mathbf{z}_i|\mathbf{z}_{j>i}, \mathbf{x}). \tag{4.9}$$

As in DN, the conditional distribution $p_\theta(\mathbf{x}|\mathbf{z})$ is described in Equation 4.6. The generative model along with the prior $p_\theta(\mathbf{z})$ and noise model $p(\mathbf{x}|\mathbf{s})$ describes the joint distribution

$p_\theta(\mathbf{z}, \mathbf{x}, \mathbf{s}) = p(\mathbf{x}|\mathbf{s})p_\theta(\mathbf{s}|\mathbf{z})p_\theta(\mathbf{z})$. Considering Equation 4.6 and Equation 4.8, the denoising loss function for HDN thus becomes

$$\mathscr{L}_{\phi,\theta}(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\sum_{i=1}^{n} -\log p(\mathbf{x}_i|g_\theta(\mathbf{z}))\right] + \mathbb{KL}\left(q_\phi(\mathbf{z}_h|\mathbf{x})||p_\theta(\mathbf{z}_h)\right) + \sum_{i=1}^{h-1} \mathbb{E}_{q_\phi(\mathbf{z}_{j>i}|\mathbf{x})}[\mathbb{KL}\left(q_{\phi,\theta}(\mathbf{z}_i|\mathbf{z}_{j>i}, \mathbf{x})||p_\theta(\mathbf{z}_i|\mathbf{z}_{j>i})\right)]. \quad (4.10)$$

Following the suggestions in [106, 107], we use residual blocks in the encoder and decoder. Additionally, as proposed in [105, 108], our generative path uses skip connections which enforce conditioning on all layers above. To avoid *KL-vanishing* [109], we use the so called *free-bits* approach [110, 111], which defines a threshold on the KL term in Equation 4.10 and then only uses this term if its value is above it. A schematic of our fully convolutional network architecture is shown in Appendix B.1.

4.4 Application: Pixel-noise Removal

	Non DL	Single Image DL		Multi Image DL			GDD Methods		Supervised	
Dataset	BM3D	DIP	S2S	N2V	N2Same	PN2V	DN	HDN	N2N	CARE
Convallaria	35.45	-	-	35.73	36.46	36.47	36.90	**37.39**	36.85	36.71
Flywing	23.45	24.67	-	24.79	22.81	24.85	25.02	**25.59**	25.67	25.79
BSD68	28.56	27.96	28.61	27.70	27.95	28.46	27.42	**28.82**	28.86	29.07
Set12	29.94	28.60	29.51	28.92	29.35	29.61	28.24	**29.95**	30.04	30.36
BioID Faces	33.91	-	-	32.34	34.05	33.76	33.12	**34.59**	35.04	35.06
CelebA HQ	33.28	-	-	30.80	31.82	33.01	31.41	**33.54**	33.39	33.57
MNIST	15.82	-	-	19.04	18.79	13.87	19.06	**20.87**	20.29	20.43
Kanji	20.45	-	-	19.95	20.28	19.40	19.47	**20.72**	20.56	20.64

TABLE 4.1: **Quantitative pixel-noise removal with HDN.** For all conducted experiments, we report results in terms of mean Peak signal-to-noise ratio (PSNR in dB) over 5 runs. The best performing method is indicated by being underlined, best performing unsupervised method is shown in bold. Our Hierarchical DivNoising is the new unsupervised SOTA method, even superseding the competing supervised baselines on 3 of the 8 used datasets.

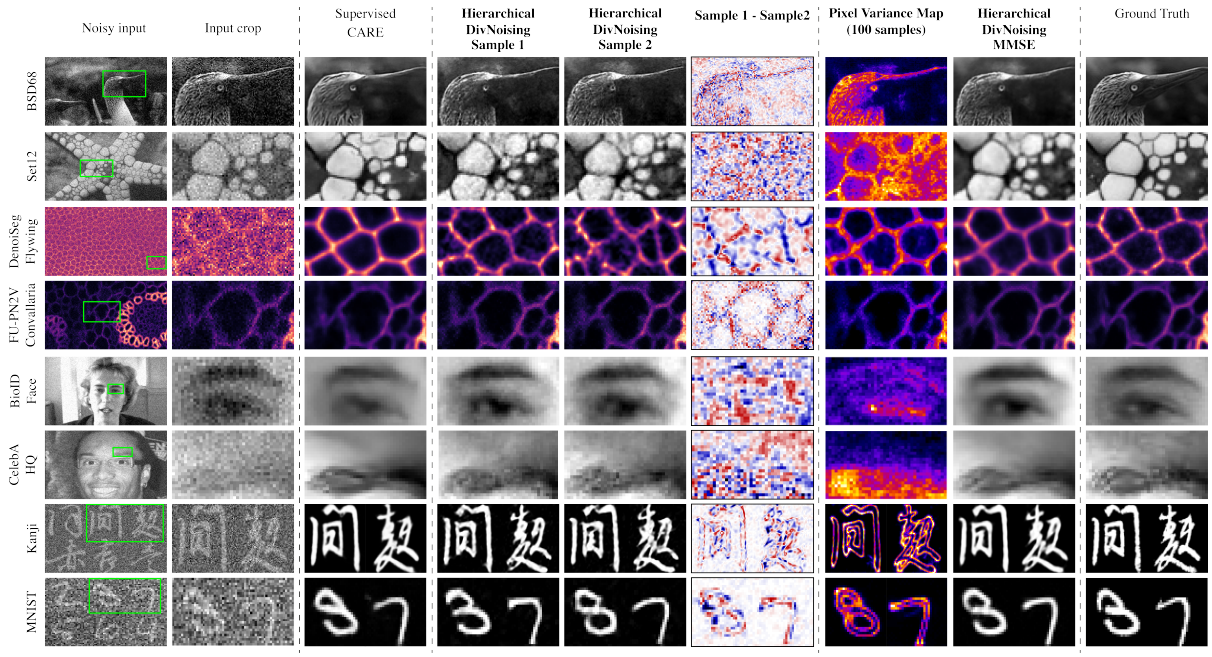

Figure 4.2: **Qualitative pixel-noise removal with Hierarchical DivNoising (HDN).** For each dataset we tested *(rows)*, we show denoising results for one representative input image. Columns show, from left to right, the input image and an interesting crop region, results of CARE, two randomly chosen HDN samples, a difference map between those samples, the per-pixel variance of 100 diverse samples, the MMSE estimate derived by averaging these 100 samples, and the ground truth image.

Datasets. We consider 8 publicly available denoising benchmark datasets from different image domains, including natural images, microscopy data, images of faces and digits. Some of these datasets are synthetically corrupted with pixel-noise while others are intrinsically noisy. A qualitative noisy sample from each dataset is shown in Figure 4.2. Appendix B.1 describes all datasets in more detail.

Baselines. We compare HDN on all datasets against (*i*) 7 unsupervised baseline methods, *i.e.*BM3D [45], Deep Image Prior (DIP) [57], Self2Self (S2S) [59], Noise2Void (N2V) [40], Probabilistic Noise2Void (PN2V) [53], Noise2Same [60](N2Same), and DivNoising (DN) [97], and (*ii*) against 2 supervised methods, namely Noise2Noise (N2N) [54] and CARE [36].

Note that we are limiting the evaluation of DIP and S2S to 3 and 2 datasets, respectively. This is motivated by the prohibitive training time of these methods (training is required for each input image).

All training parameters for N2V, PN2V, DN, N2N and CARE are as reported in [40, 53, 62, 97]. Noise2Same, Self2Self and DIP are trained using the default parameters mentioned in [60], [59] and [57], respectively.

HDN Training and Denoising. All but one HDN networks make use of 6 stochastic latent variables $\mathbf{z}_1, \ldots, \mathbf{z}_6$. The one exception holds for the *MNIST* dataset, where we only use 3 latent variables. Each $\mathbf{z}_i$ has 32 dimensions per pixel. We use an initial learning rate of 0.0003 and always train for $300,000$ steps using Adam [64]. During training, we extract random patches from the training data (128×128 patches for *BioID Faces* and natural image datasets, 256×256 patches for *CelebA HQ*, 28×28 patches for *MNIST*, and 64×64 patches for all other datasets). Additional training details can be found in Appendix B.1.

All denoising results are quantified in terms of Peak-Signal-to-Noise Ratio (PSNR) against available ground truth (GT) in Table 4.1. MMSE results of HDN are obtained by averaging 100 diverse denoised samples per noisy input. HDN outperforms all unsupervised baselines on all dataset, only for the *Set12* data being marginally outperformed by BM3D. Note also that on 4 datasets HDN even outperforms both supervised baselines.

4.5 Application: Structured Noise Removal

In this section, we ask if, and to what degree, GDD models can be used for image restoration, *i.e.*, to remove artefacts spanning multiple pixels (structured noise). To the best of our knowledge, we are the first to ask this question. Please note that none of the GDD methods described in Section 4.3 are devised with the intention of removing structured noise. We observe that all three methods are capable of removing structured noises, raising the question why this is the case.

Striping Artefacts in Microscopy. Additional to pixel-noise, microscopy images are often subject to line artefacts [17]. These undesired patterns are not removed using unsupervised non GDD methods (see Figure 4.1).

We tested vanilla VAEs, DN, and HDN on the microscopy dataset from [17], which is subject to intrinsic pixel-noise as well as said striping artefacts. In order to apply DN, we start by learning a pixel-wise GMM noise model as described in [58], using the available public implementation. Remember, this noise model only captures information about the pixel-noises contained in the data. Our HDN setup uses the same noise model but the more expressive hierarchical latent space, as previously described.

Results obtained by multiple baselines methods (N2V, PN2V and Struct N2V [17]) as well as the results obtained using a vanilla VAE, DN, and HDN are shown in Table 4.2. We observe that both the vanilla VAE and DN outperform unsupervised baselines in terms of achieved PSNR values to the available GT. While N2V and PN2V can by design not remove structured noise, Struct N2V, however, was specifically proposed for striping artefact removal in microscopy [17].

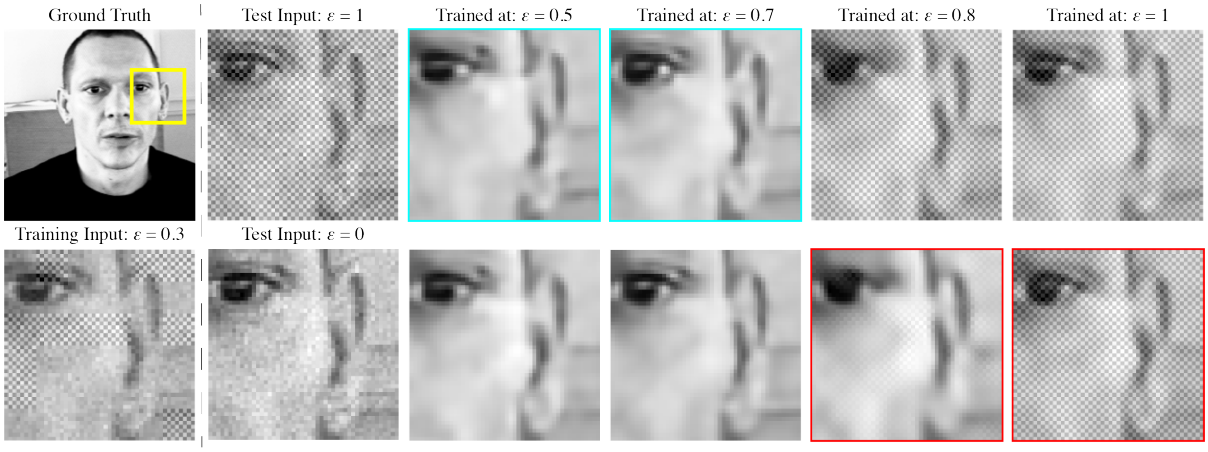

Figure 4.3: **Testing structured noise removal with DivNoising at changing artefact abundance.** The first column shows a ground truth image and below a crop of a training image subject to Gaussian noise and a superimposed chessboard patterns at an abundance of $\varepsilon = 0.3$. The second column shows two test inputs, the top one with the highest possible abundance of the chessboard patterns ($\varepsilon = 1$), while the one below does not contain this structured noise at all ($\varepsilon = 0$). The remaining 4 columns show results obtained with DivNoising networks trained on input data exposed to chessboard patterns at abundances ε of 0.5, 0.7, 0.8, and 1, respectively. Cyan frames in the top row indicate cases where DivNoising successfully removes the chessboard patterns. Red frames, in the bottom row, indicate cases where denoising results show restored chessboard patterns despite the corresponding input not containing them (*i.e.*, hallucinated artefacts).

It is interesting to see that HDN, which is clearly the best-performing method for pixel-denoising tasks (see Table 4.1), shows much worse restoration results on microscopy striping artefacts than the vanilla VAE or DN (see Table 4.2). To better understand why this is the case, we have to first understand why other GDD methods (vanilla VAE and DN) are capable of removing structured noises in the first place.

The very nature of VAE is to find a compact latent space encoding $\mathbf{z}$. By design, only what is encoded in $\mathbf{z}$ can be reconstructed by the VAEs generator. Limiting the dimensionality of $\mathbf{z}$ also means that the variational autoencoder needs to find a more compact representation of the image distribution it needs to capture. Since all methods minimize some pixel-wise loss, it pays to capture more dominant/frequent and larger structures over small and/or infrequent details. This is additionally facilitated by the fact that larger structures can be encoded in fewer bits (or latent space dimensions) as the very many bits it takes to encode finer structures, artefacts, or pixel-noises.

These considerations and the observations in [57] make us believe that GDD methods minimize the loss by first encoding the largest and most frequent/dominant structures into its latent code, and then successively add increasingly finer or less abundant (but at the same time more

	Unsupervised non-GDD Methods			Unsupervised GDD Methods				Supervised
	N2V	PN2V	Struct N2V	Vanilla VAE	DN	HDN	HDN $_{3\text{-}6}$	CARE
Struct. Conv. [17]	29.33	29.43	30.02	30.53	31.09	29.96	**31.36**	31.56

TABLE 4.2: **Quantitative results for removal of striping artefacts in microscopy data.** On a real-world microscopy benchmark (Struct-Convallaria [17]), GDD methods generally outperform all unsupervised baselines in terms of peak signal-to-noise ratio (PSNR, dB). Our HDN $_{3\text{-}6}$ method is the new SOTA on this benchmark.

latent space consuming) image features, only stopping once no further capacity in $\mathbf{z}$ remains (find some additional thoughts and experimental results giving more substance to these ideas in Appendix B.4).

With this in mind, we can now try to understand why HDN is not performing as well as vanilla VAEs or DN on microscopy data with striping artefacts. The hierarchical latent space of HDN is simply expressive enough to not only capture larger-scale image features, but can actually even pack the slightly smaller scale striping artefacts into its encoding. Interestingly, this also means that the reason why GDD methods are capable of removing structured noises from data is essentially accidental, facilitated by insufficient expressivity/dimensionality of used latent spaces.

However, owing to its hierarchical nature, we can inspect the contributions of individual latent variables $\mathbf{z}_1, \dots, \mathbf{z}_h$ to a reconstructed image. We use the trained HDN network to visualize the image aspects captured at hierarchy levels 1 to h (see Appendix B.3). Maybe little surprisingly, we find that striping artefacts are (mostly) captured in $\mathbf{z}_1$ and $\mathbf{z}_2$, the two bottom-most levels, corresponding to the highest resolution image features. With this in mind, we modified our HDN setup to compute the conditional distribution $q_{\phi,\theta}(\mathbf{z}_i|\mathbf{z}_{j>i}, \mathbf{x})$ of Equation 4.9 only using information from the *top-down* pass, neglecting the input that is usually received from *bottom-up* computations.

Since we want to exclude artefacts which are captured in $\mathbf{z}_1$ and $\mathbf{z}_2$ we apply these modifications only to these two layers and keep all operations for layers 3 to 6 unchanged. Mentioning all layers that remain unchanged, we denote this particular setup by HDN_{3-6}. As can be seen in Table 4.2, the trained HDN_{3-6} network outperforms all other unsupervised methods by a rather impressive margin.

Synthetic Structured Noise Experiments. With the previous experiments in mind, we want to better understand how GDD models remove structured noises and collect additional evidences for our intuitions. To this end, we take the *BioID Faces* dataset and synthetically apply structured corruptions and Gaussian pixel-noise (see Figure 4.3 and Figure 4.4). A detailed description of used synthetic artefacts is given in Appendix B.1. Next, we train DN using a

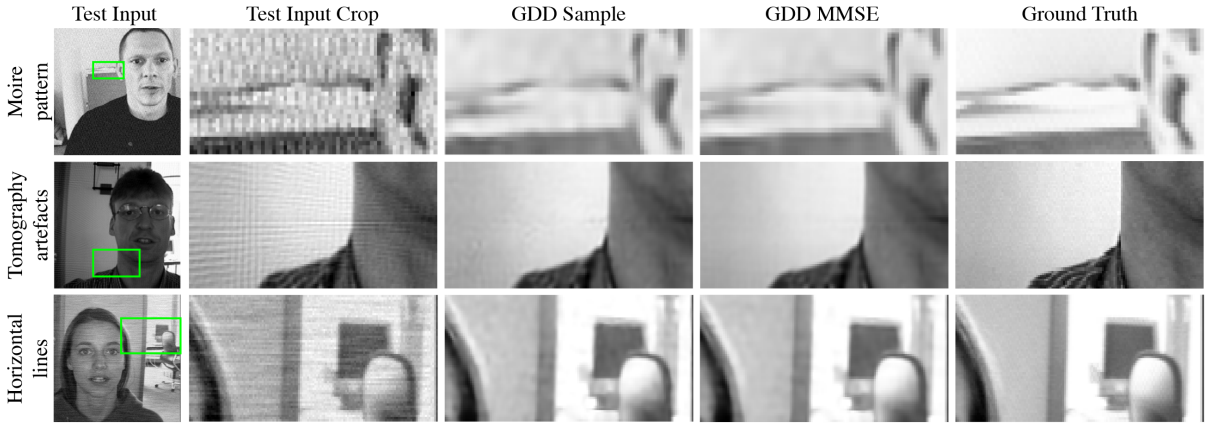

Figure 4.4: **Qualitative structured noise removal with GDDs.** We show image restoration results on data containing structured Moire patterns *(top row)*, artefacts introduced by tomographic reconstruction *(middle row)*, and horizontal line artefacts *(bottom row)*. Results shown in top and bottom rows were obtained using DivNoising, while for the middle row we used a vanilla VAE. In each row we show the input image and a selected crop, a random sample, the MMSE result (avg of 100 samples), and corresponding ground truth crop.

noise model matching the Gaussian pixel-noise we applied while corrupting the dataset. This noise model is not in any form expressing any prior knowledge about the structured noises we have used in any of the conducted experiments.

In order to better discuss our findings, we introduce the following terminologies.

Abundance/dominance and ε-fractions. During training, we would like to have control over the *abundance* or *dominance* of a given source of structured noise. Hence, we modulate the abundance by only corrupting an ε-fraction of all pixels of any given input image with structured noise. We then trained a series of DN networks, with training data covering a range of ε-fractions (see Figure 4.3).

After training networks on various ε-fractions, we test the quality of restorations on inputs with $\varepsilon = 0$ and $\varepsilon = 1$ and empirically distinguish operational regimes separated by two observational thresholds on ε.

The encoding threshold. The *encoding threshold* is the minimal abundance of a given type of structured noise in training data so that the latent space encoding of a given GDD setup starts to capture these unwanted structures. In other words, it is the largest ε-fraction one can use during training such that predictions made with the trained GDD network remain artefact free even for worst-case inputs, *i.e.*, inputs that show the structured noise everywhere ($\varepsilon = 1$).

The hallucination threshold. While the encoding threshold tells us about how abundant artefacts need to be in order to make it into the latent space encoding of a given GDD setup, a second question to ask is when a given GDD network would start to misinterpret inputs as containing the unwanted structures even though the inputs do not have any artefacts. Hence, the *hallucination-threshold* is defined as the largest ε-fraction above which a GDD method starts to *hallucinate* these structured noises, even when applied to images not corrupted with structured noise ($\varepsilon = 0$).

In Figure 4.3 we show results for a series of DN networks trained on images corrupted with Gaussian pixel-noise and structured chessboard artefacts at various ε-fractions. We observe that the *encoding threshold* and *hallucination threshold* are somewhere between 0.7 and 0.8. Results for horizontal line artefacts and Moire patterns are shown in Appendix B.7. We additionally look into DN networks trained with data below and above the *encoding threshold* and analyze/visualize latent space features of these networks. We find that networks either lack any representation of structured noise patterns or express those very broadly (see Appendix B.5).

In the top half of Table 4.3, we report image restoration results using DN after being trained at various ε-fractions for all three synthetic sources of structured noise. Additionally, we compare DN results against supervised methods Noise2Noise and CARE and report these results in Appendix B.6.

Artefact Removal in CT Reconstructions. Last but not least, we report some results on artefact removal in Computed Tomography (CT) reconstructions. To this end we have devised a synthetic CT imaging pipeline that allows us to perform in-silico CT on any given image. This has the advantage that ground truth images are known, which is typically not the case in medical CT images. Since we do not expose generated CT images to pixel-noises, we choose the vanilla VAE for our experiments. We show a qualitative result in Figure 4.4 and report quantitative results on 7 randomly selected test images using our VAE setup, DIP and Self2Self in Table 4.3.

Computed tomography uses a series of 1D projection of a 2D sample (in our case a ground truth image), typically acquired from some equally spaced viewing angles. Perfect CT reconstructions using a method called *filtered backprojection* (FBP) [112, 113] can theoretically be obtained if such projections are available at sufficient density. Reducing the number of available projections gives rise to an increasing amount of streaking artefacts [114], as can be seen in Figure 4.4.

Here we have no direct influence on the ε-fraction of corrupted pixels during training. Instead, we are modulating the abundance of CT artefacts by changing the number of projections used to create FBP reconstructions. Our results show that there exists a regime below a certain *encoding threshold* where the VAE does not encode CT artefacts. Results obtained by our VAE outperform both baselines. A comparison against supervised methods is given in Appendix B.6.

Method	Chessboard	Horizontal	Moire
DN $_{\varepsilon = 0.01}$	**33.22**	**35.11**	32.91
DN $_{\varepsilon = 0.03}$	33.04	34.98	**33.08**
DN $_{\varepsilon = 0.05}$	33.06	35.04	32.93
DN $_{\varepsilon = 0.10}$	33.05	34.80	32.85
DN $_{\varepsilon = 0.50}$	32.97	34.76	28.64
DN $_{\varepsilon = 1.0}$	19.59	34.10	26.63
test input:	200 pas	180 pas	150 pas
DIP	39.84	39.48	38.74
S2S	36.19	36.05	35.29
VAE $_{200}$	41.78	**41.18**	**39.34**
VAE $_{180}$	41.92	41.12	39.15
VAE $_{150}$	**42.07**	41.01	38.78

TABLE 4.3: **Structured noise removal with GDDs.** Top 6 rows show DivNoising (DN) results on the *BioID-Faces* dataset, subjected to 3 sources of structured noises (see text for details). Rows differ by the abundance (ε-fraction) of the respective structured noise during training. All results are evaluated on test-inputs generated at maximum artefact abundance ($\varepsilon = 1$). Bottom 5 rows compare results of DIP [57], S2S [59], and vanilla VAEs trained on tomographic reconstructions from either 150, 180, or 200 projection angles (pas), also indicated as subscript in column 1. While DIP and S2S always use the same pas during training and testing, for VAE setup we evaluated all nine combinations. All numbers indicate PSNR values *w.r.t.* available ground truth images.

4.6 Discussion

In this work, we have introduced Hierarchical DivNoising (HDN), a new GDD method utilizing expressive hierarchical latent spaces, leading to SOTA results on 7 out of 8 unsupervised denoising benchmarks. Additionally, we showed that different layers of the hierarchical latent space encode for image features at different spatial scales. We used this to selectively "deactivate" bottom-up input to latent layers when they encode undesired structured noises. We demonstrated this approach on a real-world microscopy dataset corrupted by line-scanning artefacts, where our HDN approach produces SOTA results.

We expect that selective "deactivation" of latent layers will find practical application for many microscopy datasets corrupted with structured noise. Since microscopy data is typically diffraction limited, the pixel-resolution of micrographs typically exceeds the optical resolution of visible structures. This means that true image structures are blurred by a blur kernel called the point spread function. Hence, the spatial scale of the true signal in microscopy data is not the same as the spatial scale of many structured microscopy noises and the method we proposed for microscopy data restoration in Section 4.5 will likely apply.

We have then explored the capability of GDD methods to remove structured noises beyond striping artefacts. In total we experimented with 3 synthetic sources of structured noise as well as artefacts that arise by a commonly used reconstruction pipeline in computed tomography (CT). We saw that all sources of structured noise can be removed with GDD methods as long as the trained latent space does not encode the unwanted structures. To better understand our observations, we introduced *encoding threshold* and *hallucination threshold* that we used to evaluate how robustly the GDD setups we used remove the respective source of structured noise (as a function of their, as we called it, abundances).

As we have reasoned in Section 4.5, unsupervised removal of structured noise with GDD methods is a consequence of finding a compressed latent space encoding that can or cannot describe all observable image features contained in a given body of training data. Hence, whether or not artefacts are removed is an indirect consequence of a combination of factors that are not easily controllable, *i.e.*, (*i*) the expressivity of the latent space, (*ii*) the abundance or dominance of artefacts in the training data, (*iii*) the predictability/complexity of the artefacts, and (*iv*) how much encoding the structured noise contributes to the loss used to train the GDD setup.

While, as we have shown, unsupervised GDD pixel-noise removal leads to SOTA results, for structured noise removal the situation is more complicated. Whenever the structured noise is not abundant enough to be learned by a given GDD setup, SOTA results can be achieved. Still, in order to increase the scope and practicality of GDD methods for unsupervised structured noise removal, finer grained control over what structures a GDD method picks up or ignores will be needed. In future work we will address this remaining shortcoming.

Part II

Image Segmentation

II-A

Deep Learning based Segmentation with Few Annotations

CHAPTER 5

LEVERAGING UNSUPERVISED DENOISING FOR IMAGE SEGMENTATION

This chapter marks the beginning of the second part of the thesis where we will focus primarily on the task of image segmentation as introduced in Section 1.2. However, we will not completely leave behind the image restoration ideas discussed in previous chapters. Instead, unsupervised image denoising will be used as a valuable stepping stone for enhancing the performing of Deep Learning (DL) segmentation networks. In particular, our focus is on learning the segmentation task with only limited quantity of annotated training data and we will explore different ways in which unsupervised denoising can benefit the segmentation task in these scenarios.

5.1 Introduction

The advent of modern microscopy techniques has enabled the routine investigation of biological processes at sub-cellular resolution. The growing amount of microscopy image data necessitates the development of automated analysis methods, with object segmentation often being one of the desired analyses. Over the years, a sheer endless array of methods have been proposed for segmentation [115], but deep learning (DL) based approaches are currently best performing [116–118]. Still, even the best existing methods offer plenty of scope for improvements, motivating further research in this field [119–122].

Arguably the two main causes of weak segmentation performance are (*i*) unavailability of large body of training data, and (*ii*) input images acquired at low signal-to-noise ratios (SNR). Let's elaborate on these causes and discuss some solutions that have been proposed in literature to counter them.

A trait common to virtually all DL based segmentation methods is their requirement for tremendous amounts of labeled ground truth (GT) training data, the creation of which is extraordinarily time consuming. In order to make the most out of a given amount of segmentation training data, data augmentation [123, 124] is used in most cases. Another way to increase the amount of available training data for segmentation is to synthetically generate it, *e.g.*by using Generative Adversarial Networks (GANs) [125–127]. However, the generated training data needs to capture all statistical properties of the real data and the respective generated labels, thereby making this approach cumbersome in its own right. Finally, transfer learning [128, 129] is yet another effective way to alleviate the data hunger for training DL based newtorks. Transfer learning employs networks pretrained on similar tasks and/or data for which ample training

---

Parts of this chapter is taken from published article *Prakash, Buchholz, Lalit, Tomancak, Jug, Krull*, ISBI 2020.

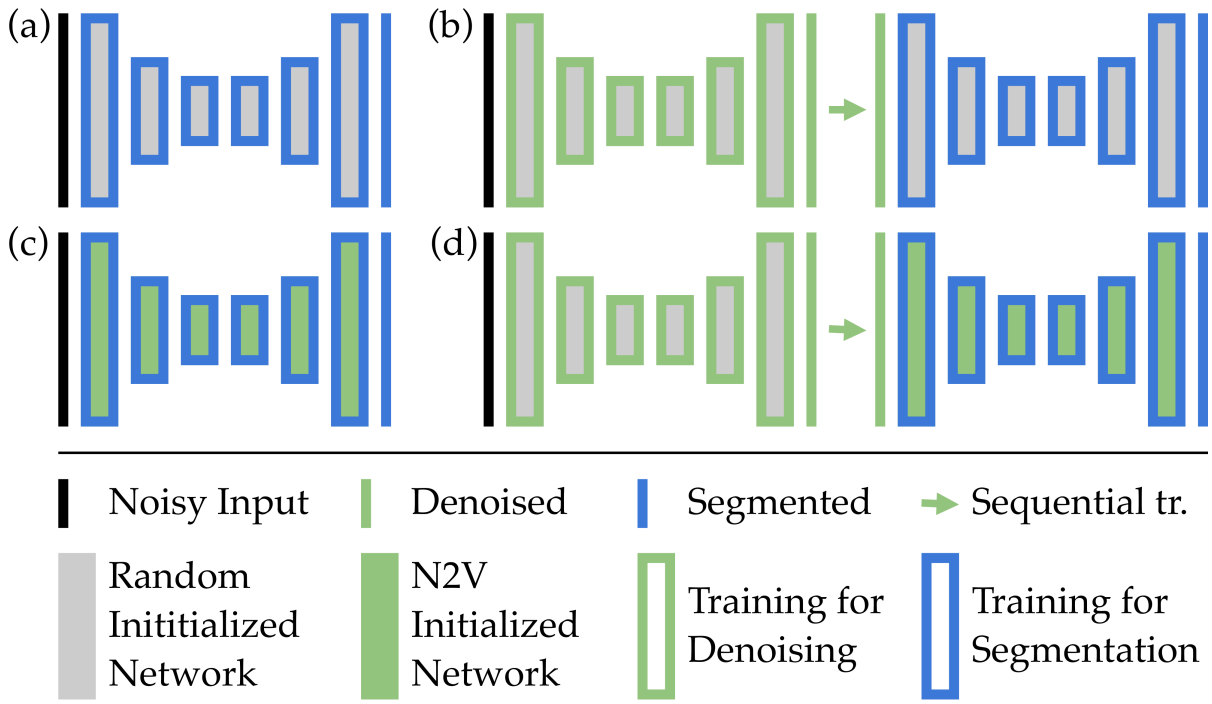

Figure 5.1: **Tested network architectures and training schedules.** **(a)** Baseline methods are directly trained to segment noisy data, **(b)** sequential setup, with denoising being the preprocessing step for subsequent segmentation, **(c)** finetuning of a pretrained denoising network for segmentation, and **(d)** finetune-sequential, combining the ideas of (b) and (c).

data is available and finetunes these networks for the task/data at hand for which training data may be scarcely available.

Especially in the context of microscopy, automated segmentation task becomes even more daunting owing to the low signal-to-noise ratio (SNR) of the acquired images. Noisy low SNR images are unavoidable as long time-lapse imaging dictates that the sample being imaged should not be exposed to high photon dose to prevent photobleaching or phototoxicity. The image corruption thus caused poses severe challenges for image segmentation. Additionally, manually creating labeled segmentation GT for noisy low SNR images is extremely painstaking making DL based segmentation approaches for such images even more inaccessible. To address the low SNR, a number of powerful content-aware restoration and denoising methods have recently been developed [36, 38, 50, 54]. Among them are self-supervised methods [40, 42, 53, 55, 58, 60, 61, 97] discussed in previous chapters which do not require paired training data, and can be directly trained on the raw data to be denoised.

In this chapter, we investigate various ways, in which self-supervised denoising can enable cell/nuclei segmentation, even in the presence of extreme levels of noise and limited training data. We explore the efficacy of denoising as a preprocessing step, as part of a transfer learning schema, as well as, in a combination of the two.

We conducted all experiments with two popular DL-based segmentation methods: a standard U-Net [63, 130] and the more sophisticated StarDist [119]. While we find that self-supervised denoising generally improves segmentation results, especially when noise is abundant and training data limited, we provide detailed results, comparing all approaches for various amounts of training data, noise levels, and types of data. All datasets, results, and code are publicly available [1].

5.2 Methods and Experiments

As sketched in Figure 5.1, we propose three ways involving self-supervised denoising to improve segmentations and compare our results to two baseline segmentation methods, namely (*i*) a standard U-Net [63] for 3-class pixel classification, and (*ii*) StarDist [119], designed to learn and utilize a star convex shape prior. These baselines are chosen based on popularity and because they follow rather different segmentation paradigms. For the self-supervised denoising component of our proposed methods, we choose Noise2Void (N2V) network discussed in Chapter 2. The following setups are the ones we propose.

- **Sequential (Figure 5.1(b))** Here, two networks are employed. The first network is a Noise2Void (N2V) network [40], trained to denoise the full body of available image data. The second network, which henceforth receives the denoised N2V output, is then either a U-Net or StarDist network, trained on all or parts of the available segmentation labels (GT). Note that all weights of the N2V network remain constant during training the segmentation network.
- **Finetune (Figure 5.1(c))** In contrast to the sequential setup, here we retrain the N2V network for segmentation. Since StarDist does not use the exact same network architecture as N2V, this approach only applies to the U-Net baseline.
- **Finetune Sequential (Figure 5.1(d))** Very similar to the sequential setup, also here we first train a N2V denoising network. In contrast to before, the segmentation network is initialized by a copy of the trained N2V network and then finetuned for segmentation. Also here, the weights of the first network stay unchanged during the training of the segmentation network.

Next we describe the detailed setup of the N2V, Segmentation U-Net, and StarDist networks.

N2V *Denoising Network:* We use the Noise2Void setup as described in [40]. Conveniently, N2V is just a default U-Net with a modified loss for denoising, allowing us to design a single

[1] github.com/juglab/VoidSeg

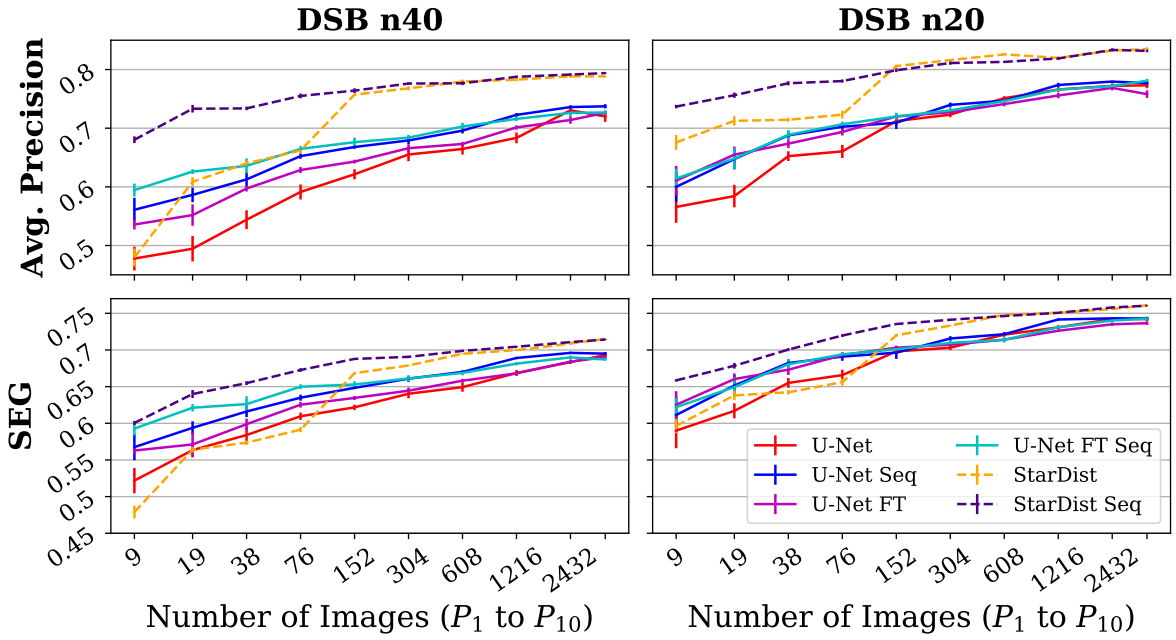

Figure 5.2: **Quantitative segmentation results for noise level n40 and n20 on DSB data.** Sequential is abbreviated as Seq and Finetune is abbreviated as FT. It can be seen that our proposed training schemes consistently outperform the respective baselines, mainly when only limited segmentation GT is available.

network that can later be used for N2V training as well as for the U-NET segmentation baseline. We use 32 initial feature maps with batch norm and a batch size of 128 and employ 3×3 convolution kernels. For all experiments we choose the depth of the U-NET as described below.

U-NET *Segmentation Network:* We created a U-NET capable of performing either 3-class pixel classification (foreground, border, background) [131, 132] or N2V denoising. Hence, the U-NET we use has four output channels, one for each pixel class, and one to regress denoised pixel intensities. Note that, during pixel classification, we give extra emphasis to the border class, by weighing it five times higher in the used loss as suggested in [119]. Again we use 32 feature maps, batch size of 128, and 3×3 kernels. For all experiments, the depth of the U-NET is chosen to saturate segmentation performance (making the network deeper would not lead to improved results). Hence, results below are not limited by the capacity of network. All networks are trained with a standard learning rate scheduler as used in [40]. We use an initial learning rate of 0.0004 and a batch size of 128 with batch normalization. Training is done for 200 epochs, each consisting of 400 steps. Training data is augmented 8 fold by flips and 90 degree rotations.

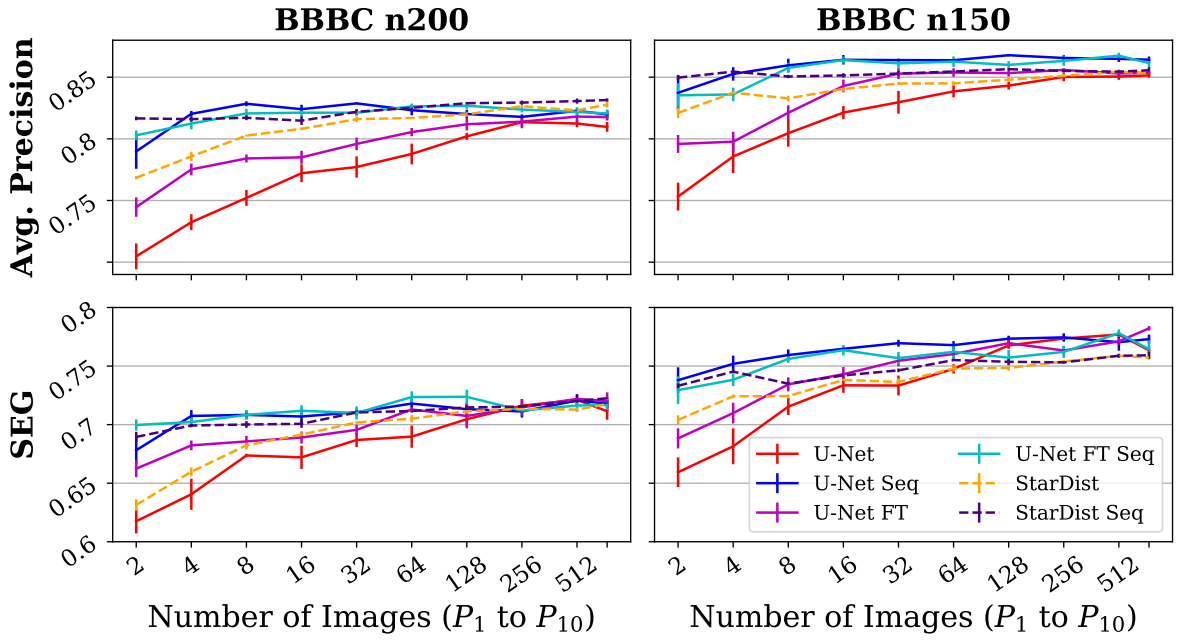

Figure 5.3: **Quantitative segmentation results for noise level n200 and n150 on BBBC data.** The abbreviations are the same as in Figure 5.2. Again, all proposed training schemes outperform their baselines. Here our proposed sequential U-Net schemes even outperform StarDist and StarDist Sequential.

StarDist Segmentation Network: Number of feature maps, batch size, convolution kernels, network depth, learning rate, number of training epochs, and step size per epoch used for StarDist are set as described above for U-Net. Again, the training data is augmented 8 fold by flips and 90 degree rotations. However, StarDist uses 33 output channels that are trained as described in [119].

5.3 Data and Evaluation Metrics

In this work we use publicly available data, which we randomly split into training and test sets (see following subsections for details). We further split the training data into $P_1 \subset P_2 \subset \ldots \subset P_{10}$, ten stacked subsets we will use to evaluate our methods in data-limited training regimes. Additionally, we corrupt the raw microscopy data with pixel independent, identically distributed Gaussian noise. Sample images for all datasets are shown in Figure 5.4.

DSB 2018 Data: From the Kaggle 2018 Data Science Bowl challenge, we take the same subset of data as has been used in [119], showing a diverse collection of cell nuclei imaged by various fluorescence microscopes. We extracted 4470 image patches of size 128×128 from the training set. For this data, manually generated segmentation GT is available. Training subsets P_1 through P_{10} consist of $10, 19, 38, 76, 152, 304, 608, 1216, 2432, 3800$ randomly chosen image patches, respectively. The remaining 670 patches constitute the validation set while the test set has 50 additional images of different sizes. Additional noise is added with mean 0 and standard deviations 10, 20, and 40 to training, validation and test data. We refer to the modified datasets as n10, n20, and n40, respectively.

BBBC 004 Data: This data is available from the Broad Bioimage Benchmark Collection and consists of synthetic nuclei images. Since the data is synthetic, perfect GT labels are available by construction. Here we use only the images having non-touching nuclei. We extracted 880 image patches (of size 128×128) from the training set. Training subsets P_1 through P_{10} consist of $2, 4, 7, 15, 30, 60, 120, 239, 479, 748$ image patches, respectively while the validation set consists of remaining 132 patches. The test set consists of additional 220 patches. Additional noise is added with mean 0 and standard deviations 150 and 200 to training and test data. Following the naming convention from above, we refer to this data as n150 and n200.

All experiments we conduct are evaluated in terms of Average Precision (AP) [133] and SEG [134] scores. The SEG measure is based on the Jaccard similarity index (J), computed for matching objects S and R, and is given by $J(S, R) = (|R \cap S|)/(|R \cup S|)$. A ground truth object R and a segmented object S are considered to be matching if and only if at least 50% of the pixels of R are overlapped by pixels in S. AP, in contrast, counts the ratio of true positives to the sum of true positives, false positives, and false negatives. All AP and SEG values we report here are obtained by finding the threshold on the validation set that maximizes AP. For the U-Net this threshold is used to cut the foreground probability maps into discrete image regions. For StarDist the threshold controls the non-maxima suppression step [119].

5.4 Results

We investigated all setups described above, on all noise levels, using all 10 subsets of training data P_i, making a total of 60 experimental setups. Each experiment on the DSB data was repeated 8 times while all experiments on the BBBC data were repeated 5 times, allowing us to report mean performance and standard error for selected noise levels in Figure 5.2 and Figure 5.3. Additionally we show results for DSB data at noise level n40 and results for BBBC data at n200 in Table 5.1. A complete set of figures and tables, for all conducted experiments, can be found online at `github.com/juglab/VoidSeg/wiki`.

Looking at all results it can be observed that all our proposed schemes outperform their respective baseline when the amount of available training data is limited. The Finetune Sequential scheme is typically performing best among all U-Net based pixel classification pipelines. StarDist, in itself a more powerful method, is indeed the better performing baseline. As before, the proposed StarDist Sequential scheme clearly outperforms its baseline method when fewer training images are available. Note that even if ample training data is provided, all our proposed training schemes perform at least on par with

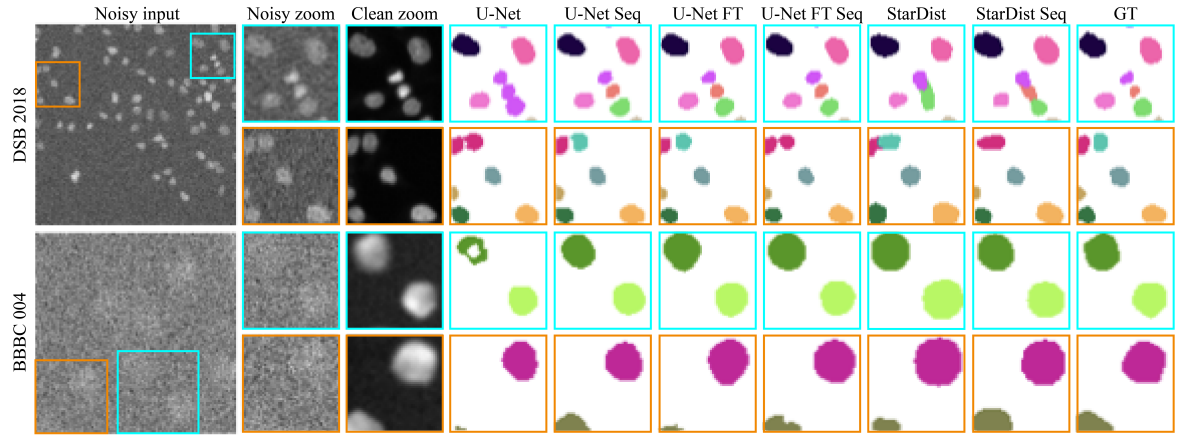

Figure 5.4: **Visual comparison of segmentation results with baseline methods and proposed training schemes for DSB n40 P_1 and BBBC n200 P_1.** From left to right we show first one noisy input image, then the two insets, respective noise-free data, and the various segmentation results with each object shown in a distinct color. Sequential is abbreviated as Seq and Finetune is abbreviated as FT. In line with the overall performance on the full body of data, also in the examples we show our proposed methods outperform the quality achieved by the baselines.

their baselines. It is important to be reminded that improved results using sequential training schemes are not due to limiting network sizes – we have tested various network sizes for both baseline methods and have settled for the best performing configuration we could find.

To our surprise, on the BBBC data, both sequential U-Net schemes outperform StarDist and StarDist Sequential for the n150 and n200 noise levels, despite the StarDist baseline consistently and significantly outperforming the U-Net baseline.

A visual comparison of segmentation results with all the methods trained on the training subset P_1 is given in Figure 5.4. For the DSB data we show insets that exemplify the often occurring problem of merging segments (bad for AP), while the shown BBBC insets show variations in segmented areas (bad for SEG). These segmentation mistakes are particularly exemplified for baseline schemes whereas sequential schemes for both U-Net and StarDist seem to yield better quality segmentation, in general.

5.5 Discussion

It is known that there is an overlap between denoising and segmentation tasks [129]. In this work we investigated how disentangling the two can be exploited in practice, when noisy data is abundant, but annotations are rare – a situation that is virtually ubiquitously true in biomedical applications.

In these situations, all our proposed schemes show above baseline performance. Among all conducted experiments, sequential training schemes generally lead to the best results. Since this is not only true for the simple U-Net baseline, but also for StarDist, it stands to reason that similar observations would

also hold for other DL based approaches and tasks. Here we do not test other segmentation approaches, but we believe that N2V, or other self-supervised denoising methods [41, 55], can serve as universal preprocessing blocks for networks solving any given super-task in which denoising is a helpful sub-task.

These denoising blocks can benefit from the whole body of available noisy data, without relying on annotated GT labels required for the super-task. Hence, finding sensible training schedules to train such larger, modular networks is a promising direction of research.

In summary, we show that commonly used networks for image segmentation can likely be boosted in performance by combining them in various ways with unsupervised denoising modules. Our work offers simple recipes for improving DL based segmentation results. Since this is increasingly true at lower signal-to-noise regimes and when segmentation GT is limited, direct benefits for the biomedical imaging community will be inevitable.

DSB 2018 n40										
Scheme	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
U-Net	0.477,	0.494,	0.543,	0.591,	0.621,	0.655,	0.664,	0.683,	0.730,	0.719,
	0.521	*0.563*	*0.584*	*0.609*	*0.621*	*0.640*	*0.649*	*0.668*	*0.683*	*0.692*
U-Net Seq.	0.560,	0.586,	0.612,	0.652,	0.667,	0.679,	0.695,	**0.722**,	**0.736**,	**0.737**,
	0.567	*0.593*	*0.616*	*0.634*	*0.648*	*0.660*	***0.670***	***0.689***	***0.696***	***0.695***
U-Net FT	0.535,	0.551,	0.597,	0.628,	0.643,	0.6658,	0.673,	0.701,	0.714,	0.726,
	0.562	*0.571*	*0.598*	*0.625*	*0.634*	*0.644*	*0.658*	*0.668*	*0.684*	*0.690*
U-Net FT Seq.	**0.594**,	**0.625**,	**0.635**,	**0.664**,	**0.676**,	**0.683**,	**0.702**,	0.715,	0.726,	0.726,
	0.592	***0.621***	***0.626***	***0.649***	***0.652***	***0.661***	*0.668*	*0.681*	*0.689*	*0.687*
StarDist	0.479,	0.608,	0.640,	0.662,	0.757,	0.7679,	**0.779**,	0.782,	0.788,	0.788,
	0.478	*0.563*	*0.573*	*0.591*	*0.668*	*0.678*	*0.694*	*0.699*	*0.708*	***0.715***
StarDist Seq.	**0.680,**	**0.733,**	**0.733,**	**0.754,**	**0.764,**	**0.776,**	0.776,	**0.787,**	**0.791,**	**0.793,**
	0.600	***0.639***	***0.654***	***0.672***	***0.687***	***0.690***	***0.698***	***0.704***	***0.710***	*0.714*
BBBC 004 n200										
Scheme	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}
U-Net	0.704,	0.732,	0.752,	0.772,	0.777,	0.787,	0.802,	0.813,	0.812,	0.809,
	0.617	*0.640*	*0.673*	*0.672*	*0.686*	*0.689*	*0.704*	***0.716***	*0.721*	*0.711*
U-Net Seq.	0.789,	**0.820**,	**0.828**,	**0.824**,	**0.828**,	0.823,	0.820,	0.817,	**0.822**,	0.819,
	0.678	***0.707***	***0.708***	*0.706*	***0.710***	*0.717*	*0.713*	*0.711*	*0.720*	*0.717*
U-Net FT	0.744,	0.775,	0.784,	0.785,	0.795,	0.8055,	0.811,	0.813,	0.818,	0.817,
	0.662	*0.682*	*0.685*	*0.688*	*0.695*	*0.712*	*0.707*	*0.714*	***0.721***	***0.720***
U-Net FT Seq.	**0.802**,	0.812,	0.820,	0.821,	0.821,	**0.826**,	**0.826**,	**0.823**,	0.821,	**0.820**,
	0.699	*0.702*	*0.708*	***0.712***	*0.710*	***0.723***	***0.723***	*0.712*	*0.716*	*0.717*
StarDist	0.768,	0.785,	0.802,	0.808,	0.815,	0.816,	0.819,	0.826,	0.823,	0.827,
	0.631	*0.660*	*0.682*	*0.691*	*0.702*	*0.705*	*0.711*	*0.713*	*0.713*	*0.717*
StarDist Seq.	**0.817**,	**0.816**,	**0.817**,	**0.815**,	**0.822**,	**0.825**,	**0.829**,	**0.829**,	**0.831**,	**0.831**,
	0.689	***0.699***	***0.700***	***0.701***	***0.710***	***0.712***	***0.715***	***0.715***	***0.720***	***0.722***

TABLE 5.1: **Mean performance in terms of average precision (AP) and SEG (in italic) for DSB n40 (8 repetitions) and for BBBC n200 (5 repetitions).** Bold number indicate the best performing scheme for a given fraction of segmentation GT (P_i). See the main text for further details.

CHAPTER 6

DENOISEG: JOINT DENOISING AND SEGMENTATION

In the previous chapter, we demonstrated on various microscopy datasets that self-supervised denoising with Noise2Void [40] prior to object segmentation leads to greatly improved segmentation results, especially when only small numbers of segmentation GT images are available for training. The advantage of this approach stems from the fact that the self-supervised denoising module can be trained on the full body of available microscopy data. In this way, the subsequent segmentation module receives images that are easier to interpret, leading to an overall gain in segmentation quality even without having a lot of GT data to train on. In the context of natural images, a similar combination of denoising and segmentation was proposed by Liu *et al.* [135] and Wang *et al.* [136]. However, both methods lean heavily on the availability of paired low- and high-quality image pairs for training their respective denoising module. Additionally, their cascaded denoising and segmentation networks make the training comparatively computationally expensive.

In this chapter, we build on the learnings from the last chapter and further explore the connection between image denoising and segmentation. Here we present DenoiSeg, another novel training scheme that leverages denoising for object segmentation (see Figure 6.1). Like the last chapter, we employ self-supervised Noise2Void (N2V) for denoising. However, while the most successful methods from the last chapter relied on two sequential steps for denoising and segmentation, here we propose to use a single network to jointly predict the denoised image and the desired object segmentation. We use a simple U-Net [63] architecture, making training fast and accessible on moderately priced consumer hardware. Similar to the methods presented in the last chapter, DenoiSeg also focuses especially on the segmentation of noisy microscopy data while requiring only a small fraction of images annotated with GT segmentations.

6.1 Methods

We propose to jointly train a single U-Net for segmentation and denoising tasks. While for segmentation only a small amount of annotated GT labels are available, the self-supervised denoising module benefits from all available raw images. In the following we will first briefly recapitulate how these tasks can be addressed separately, introduce necessary notations for this chapter and then introduce a joint loss function combining the two tasks.

Segmentation. Following what we introduced in the previous chapter, we see segmentation as a 3-class pixel classification problem [131, 132, 137] and train a U-Net to classify each pixel as foreground, background or border (this yields superior results compared to a simple classification into foreground and background [119]). Our network uses three output channels to predict each pixel's probability of belonging to the respective class. We train it using the standard cross-entropy loss, which will be denoted

---

Parts of this chapter is taken from published article *Buchholz*, Prakash*, Schmidt, Krull, Jug*, BIC@ECCV 2020.

*Equal contribution (alphabetical order)

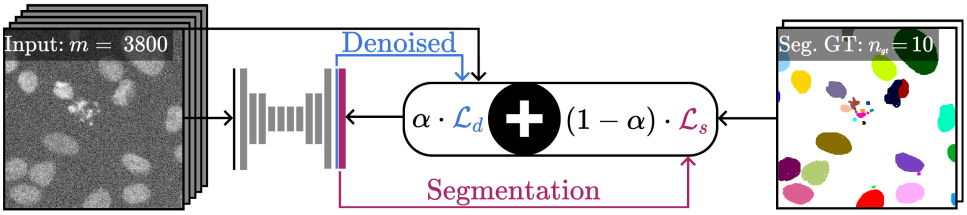

Figure 6.1: **The proposed DenoiSeg training scheme.** A U-Net is trained with a joint self-supervised denoising loss ($\mathscr{L}_d$) and a classical segmentation loss ($\mathscr{L}_s$). Both losses are weighted with respect to each other by a hyperparameter α. In this example, $\mathscr{L}_d$ can be computed on all $m = 3800$ training patches, while $\mathscr{L}_s$ can only be computed on the $n_{gt} = 10$ annotated ground truth patches that are available for segmentation.

as $\mathscr{L}_s\big(\mathbf{c}^i, f(\mathbf{x}^i)\big)$, where $\mathbf{x}^i$ is the i-th training image, $\mathbf{c}^i$ is the ground truth 3-class segmentation, and $f(\mathbf{x}^i)$ is the network output.

Self-Supervised Denoising. Following our approach in the previous chapter, we use the N2V setup described in [40] as our self-supervised denoiser of choice. We extend the above mentioned 3-class segmentation U-Net by adding a fourth output channel, which is used for denoising and trained using the N2V scheme. N2V uses a Mean Squared Error (MSE) loss, which is calculated over a randomly selected subset of blind spot pixels that are masked in the input image. Since the method is self-supervised and does not require ground truth, this loss $\mathscr{L}_d\big(\mathbf{x}^i, f(\mathbf{x}^i)\big)$ can be calculated as a function of the input image $\mathbf{x}^i$ and the network output $f(\mathbf{x}^i)$.

Joint-Loss. To jointly train our network for denoising and segmentation we use a combined loss. For a given training batch $(\mathbf{x}^1, \mathbf{c}^1, \ldots, \mathbf{x}^m, \mathbf{c}^m)$ of m images, we assume that GT segmentation is available only for a subset of $n_{gt} \ll m$ raw images. We define $\mathbf{c}^i = \mathbf{0}$ for images where no segmentation GT is present. The loss over a batch is calculated as

$$\mathscr{L} = \frac{1}{m} \sum_{i=1}^{m} (\alpha \cdot \mathscr{L}_d\big(\mathbf{x}^i, f(\mathbf{x}^i)\big) + (1-\alpha) \cdot \mathscr{L}_s\big(\mathbf{c}^i, f(\mathbf{x}^i)\big)), \tag{6.1}$$

where $0 \leqslant \alpha \leqslant 1$ is a tunable hyperparameter that determines the relative weight of denoising and segmentation during training. Note that the N2V loss is self-supervised, therefore it can be calculated for all raw images in the batch. The cross-entropy loss however requires GT segmentation and can only be evaluated on a subset of images, where this information is available. For images where no GT segmentation is available we define $\mathscr{L}_s\big(\mathbf{c}^i = \mathbf{0}, f(\mathbf{x}^i)\big) = 0$.

In the setup described above, setting $\alpha = 1$ corresponds to pure N2V denoising. However, setting $\alpha = 0$ does not exactly correspond to the vanilla 3-class segmentation, due to two reasons. Firstly, only some of the images are annotated but in Equation 6.1 the loss is divided by the constant batch size m. This effectively corresponds to a reduced batch size and learning rate, compared to the vanilla method. Secondly, our method applies N2V masking of blind spot pixels in the input image.

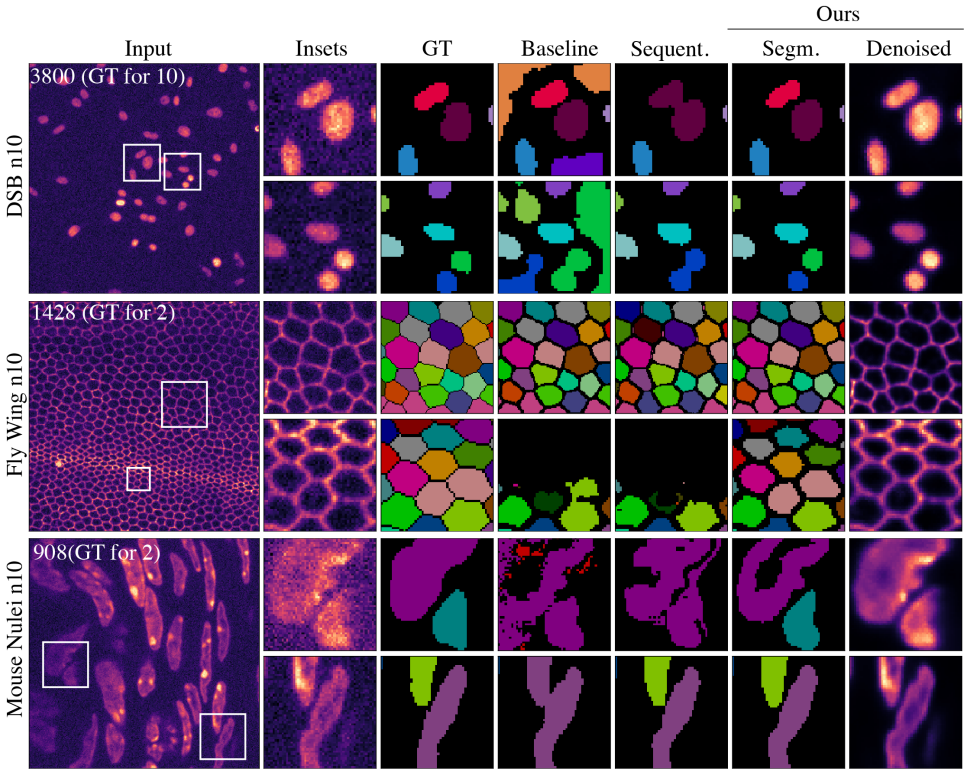

Figure 6.2: **Qualitative segmentation results on Flywing n10 (first row), DSB n10 (second row) and Mouse Nuclei n10 (third row).** The first column shows an example test image. Numbers indicate how many noisy input and annotated ground truth (GT) patches were used for training. Note that segmentation GT was only available for at most 10 images, accounting for less than 0.27% of the available raw data. Other columns show depicted inset regions, from left to right showing: raw input, segmentation GT, results of two baseline methods, and our DenoiSeg segmentation and denoising results.

Implementation Details. Our DenoiSeg implementation is publicly available[1]. The proposed network produces four output channels corresponding to denoised images, foreground, background and border segmentation. For all our experiments we use a U-Net architecture of depth 4, convolution kernel size of 3, a linear activation function in the last layer, 32 initial feature maps, and batch normalization during training. All networks are trained for 200 epochs with an initial learning rate of 0.0004. The learning rate is reduced if the validation loss is not decreasing over ten epochs. For training we use 8-fold data augmentation by adding 90° rotated and flipped versions of all images.

[1]https://github.com/juglab/DenoiSeg, https://imagej.net/DenoiSeg

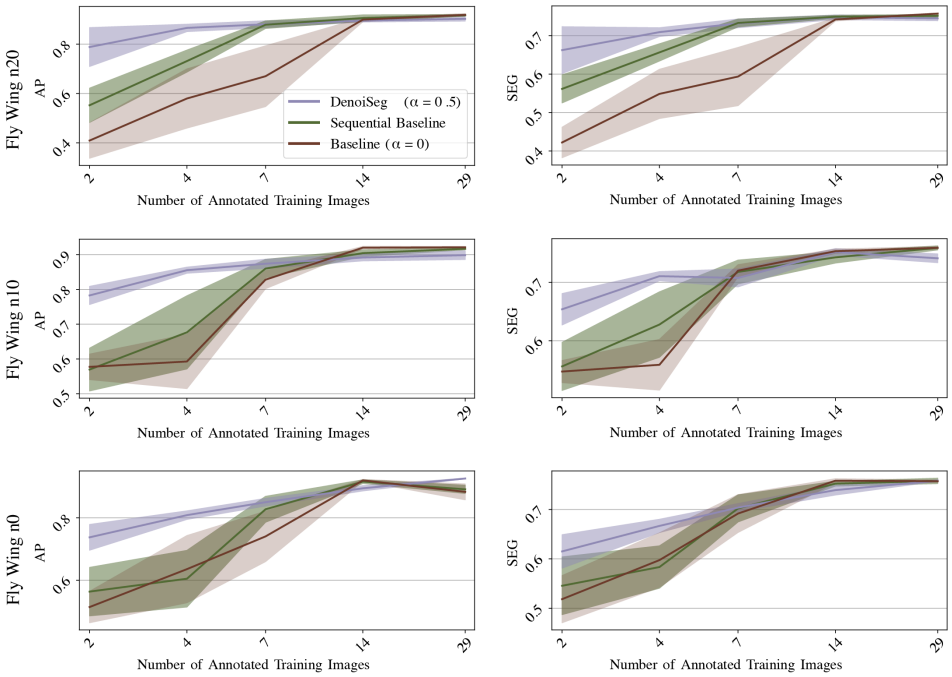

Figure 6.3: **Quantitative segmentation results for Flywing n0, n10 and n20, evaluated with Average Precision (AP) [119] and SEG-Score [134]**. DenoiSeg outperforms both baseline methods, mainly when only limited segmentation ground truth is available.

Additionally, we also provide a DenoiSeg plugin for Fiji [31], a popular Java-based image processing software. The plugin does not require computational expertise to be used and provides comprehensive visual feedback during training. Trained models can then be applied to new data and shared with other users for use in either Python or Fiji.

6.2 Experiments and Results

We use three publicly available datasets for which GT annotations are available (data available at DenoiSeg-Wiki[1]). For each dataset we generate noisy versions by adding pixel-wise independent Gaussian noise with zero-mean and standard deviations of 10 and 20. The dataset names are extended by n0, n10, and n20 to indicate the respective additional noise. For network training, patches of size 128×128 are extracted and randomly split into training (85%) and validation (15%) sets.

[1]https://github.com/juglab/DenoiSeg/wiki

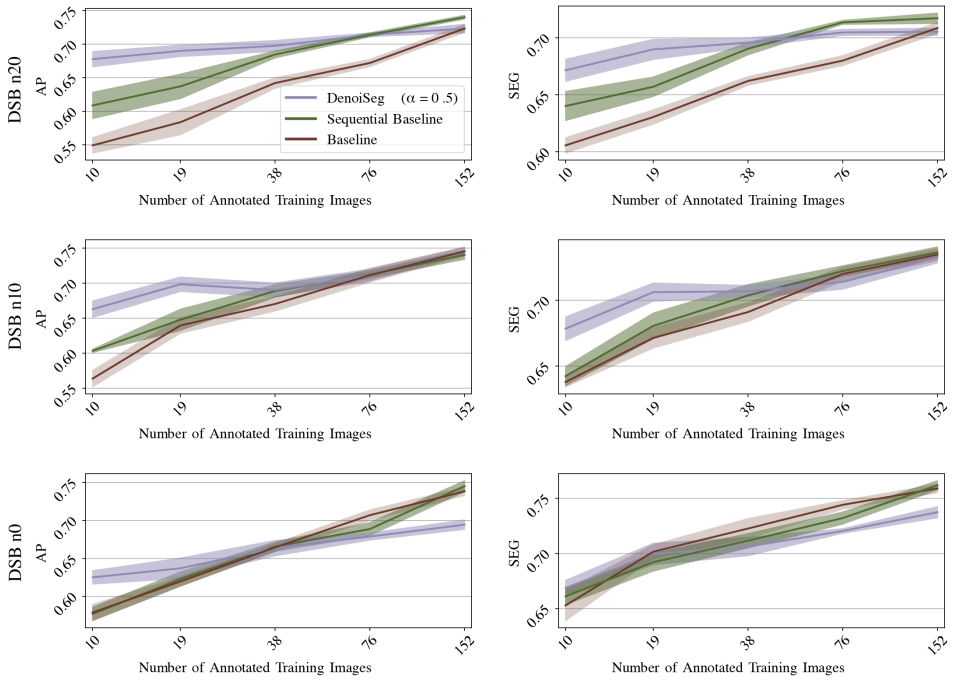

Figure 6.4: **Quantitative segmentation results for DSB n0, n10 and n20, evaluated with Average Precision (AP) [119] and SEG-Score [134].** DenoiSeg outperforms both baseline methods, mainly when only limited segmentation ground truth is available. Note that the advantage of our proposed method is at least partially compromised when the image data is not noisy (row 3).

- **Flywing.** This dataset consists of 1428 training and 252 validation patches of size 128×128 showing a membrane labeled flywing. The test set is comprised of 50 additional images of size 512×512.
- **DSB.** From the Kaggle 2018 Data Science Bowl challenge, we take the same images as used by [137]. The training and validation sets consist of 3800 and 670 patches respectively of size 128×128, while the test set counts 50 images of different sizes.
- **Mouse Nuclei.** Finally, we choose a challenging dataset depicting diverse and non-uniformly clustered nuclei in the mouse skull, consisting of 908 training and 160 validation patches of size 128×128. The test set counts 67 additional images of size 256×256.

For each dataset, we train DenoiSeg and compare it to two different competing methods: DenoiSeg trained purely for segmentation with $\alpha = 0$ (referred to as *Baseline*), and the sequential scheme introduced in the last chapter that first trains a denoiser and then the aforementioned baseline (referred to as *Sequential*). We chose our network with $\alpha = 0$ as baseline to mitigate the effect of batch normalization on the learning rate as described in Section 6.1. A comparison of our baseline to a vanilla 3-class

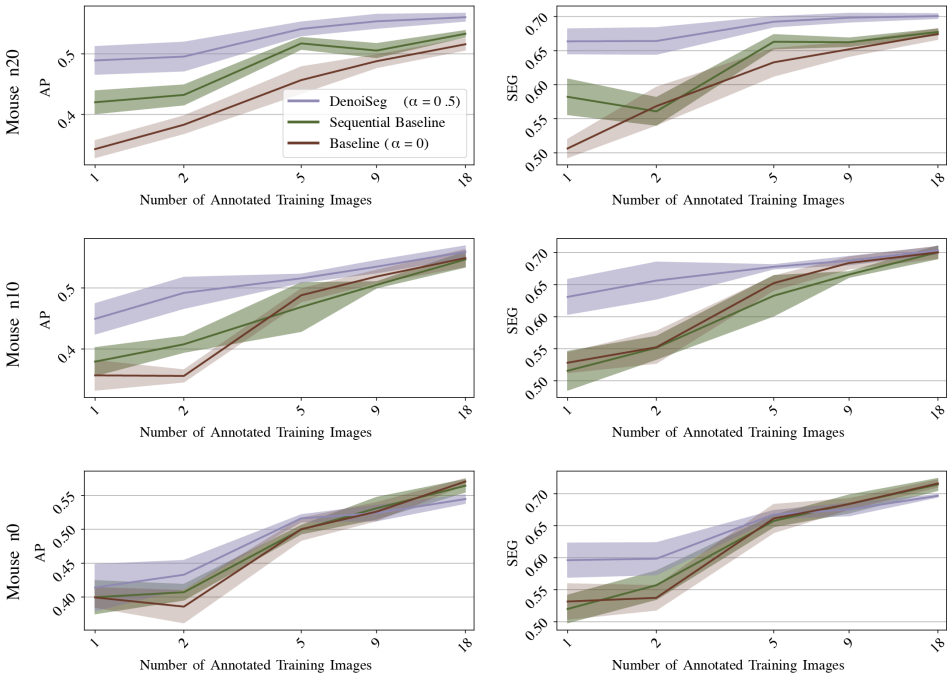

Figure 6.5: **Quantitative segmentation results for Mouse Nuclei n0, n10 and n20, evaluated with Average Precision (AP) [119] and SEG-Score [134]**. DenoiSeg outperforms both baseline methods, mainly when only limited segmentation ground truth is available.

U-Net with the same hyperparameters leads to very similar results and can be found in Appendix C.2. Furthermore, we investigate DenoiSeg performance when trained with different amounts of available GT segmentation images. This is done by picking random subsets of various sizes from the available GT annotations as also done in the previous chapter. Note that the self-supervised denoising task still has access to all raw input images. A qualitative comparison of DenoiSeg results with other baselines (see Figure 6.2) indicates the effectiveness of our method.

As evaluation metrics, we use Average Precision (AP) [133] and SEG [134] scores introduced in the previous chapter. The AP metric measures both instance detection and segmentation accuracy while SEG captures the degree of overlap between instance segmentations and GT. To compute the scores, the predicted foreground channel is thresholded and connected components are interpreted as instance segmentations. The threshold values are optimized for each measure on the validation data. All conducted experiments were repeated 5 times and the mean scores along with ± 1 standard error of the mean are reported in Appendix C.1.

Performance with Varying Quantities of GT Data and Noise. Figure 6.3 shows the results of DenoiSeg with $\alpha = 0.5$ (equally weighting denoising and segmentation losses) for Flywing n0, n10

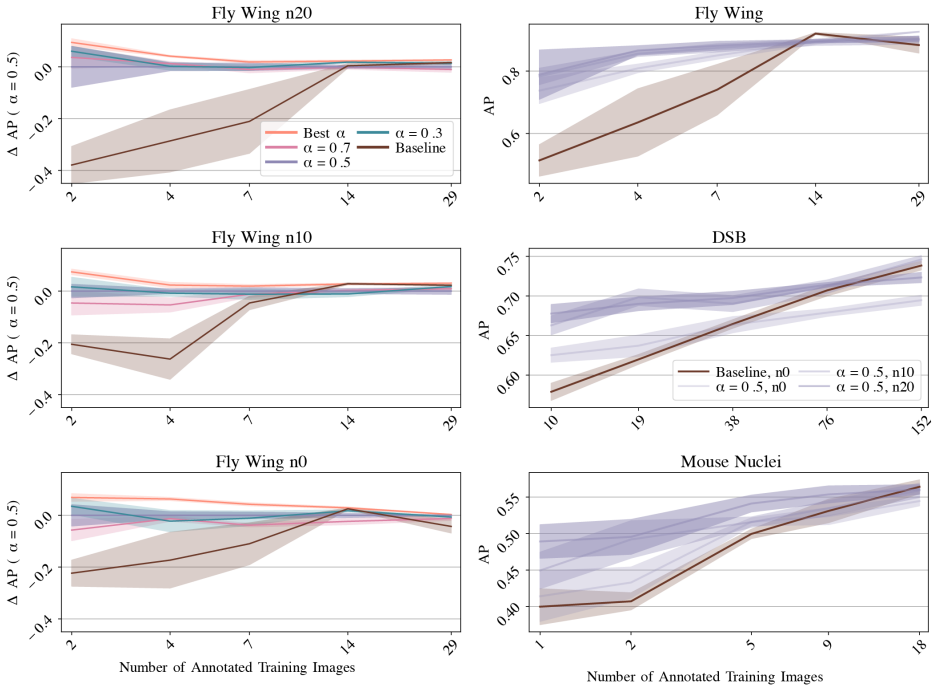

Figure 6.6: **Importance of relative weighting to denoising and segmentation losses and importance of noise on inputs for Fly Wing data.** In **(a)**, we show that DenoiSeg consistently improves results over the baseline for a broad range of hyperparameter α values by looking at the difference Δ of AP to $\alpha = 0.5$. The results come close to what would be achievable by choosing the best possible α (see main text). In **(b)**, we show that adding synthetic noise can lead to improved DenoiSeg performance. For the Flywing, DSB, and Mouse Nuclei data, we compare baseline results with DenoiSeg results on the same data (n0) and with added synthetic noise (n10 and n20, see main text).

and n20 datasets. For low numbers of GT training images, DenoiSeg outperforms all other methods. Similar results are seen for the other two datasets (see Figure 6.4 and Figure 6.5). Note that, as the number of GT training images reaches full coverage, the performance of the baselines are similar to DenoiSeg. Appendix C.1 shows this for all datasets. Results for all performed experiments showing overall similar trends can be found on the DenoiSeg-Wiki.

Importance of α. We further investigated the sensitivity of our results to the hyperparameter α. In Figure 6.6(a) we look at the segmentation performance for different values of hyperparameter α. We compare the results of $\alpha = 0.3$ and $\alpha = 0.7$ by computing the difference (Δ **AP**). Δ **AP** is the difference of the obtained AP score and the AP score obtained with the default $\alpha = 0.5$. Additionally we also compare to the Baseline and results that use (the a priori unknown) best α. The best α for each trained

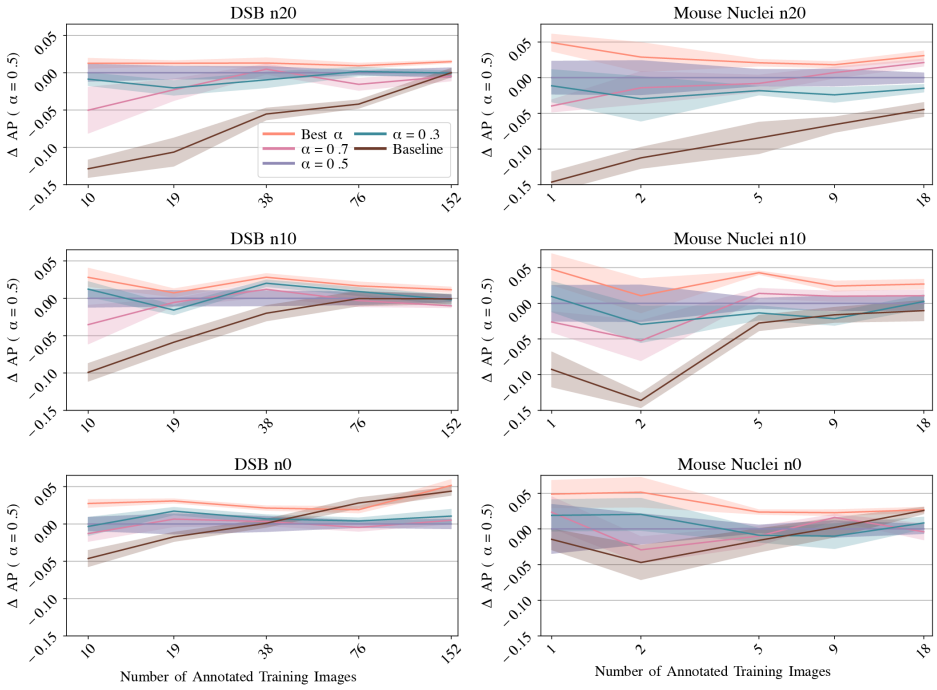

Figure 6.7: **Importance of relative weighting to denoising and segmentation losses for DSB and Mouse Nuclei data.** We show the sensitivity of hyperparameter α for **(a)** DSB and **(b)** Mouse Nuclei datasets by looking at the difference Δ of AP to $\alpha = 0.5$. Note that DenoiSeg consistently improves results over the baseline for a broad range of hyperparameter α values.

network is found by a grid search for $\alpha \in \{0.1, 0.2, \ldots, 0.9\}$. Figure 6.6(a) shows that our proposed method is robust with respect to the choice of α. Results for the other datasets showing similar trends are illustrated in Figure 6.7.

Noisy Inputs Lead to Elevated Segmentation Performance. Here we want to elaborate on the interesting observation we made in Figure 6.3: when additional noise is synthetically added to the raw data, the segmentation performance reaches higher AP and SEG scores, even though segmentation should be more difficult in the presence of noise. We investigate this phenomenon in Figure 6.6(b). We believe that in the absence of noise the denoising task can be solved trivially, preventing the regularizing effect that allows DenoiSeg to cope with small amounts of training data.

Evaluation of Denoising Performance. Although we are not training DenoiSeg networks for their denoising capabilities, it is interesting to know how their denoising predictions compare to dedicated denoising networks. Table 6.1 compares our denoising results with results obtained by N2V. It can

be seen that co-learning segmentation is only marginally impeding the network's ability to denoise its inputs.

	DSB (GT for 10)		Flywing (GT for 2)		Mouse Nuclei (GT for 1)	
Noise	DenoiSeg	N2V	DenoiSeg	N2V	DenoiSeg	N2V
n10	37.57±0.07	38.01±0.05	33.12±0.01	33.16±0.01	37.42±0.10	37.86±0.01
n20	35.38±0.08	35.53±0.02	30.45±0.20	30.72±0.01	34.21±0.19	34.59±0.01

TABLE 6.1: **Comparing the denoising performance of DenoiSeg and Noise2Void (N2V).** Mean Peak Signal-to-Noise Ratio values (with ±1 SEM over 5 runs) are shown. Similar tables for DenoiSeg results when more segmentation GT was available can be found online in the DenoiSeg-Wiki.

6.3 Discussion

Here we have shown that (*i*) joint segmentation and self-supervised denoising leads to improved segmentation quality when only limited amounts of segmentation ground truth is available (Figure 6.2, Figure 6.3, Figure 6.4 and Figure 6.5), (*ii*) the hyperparameter α is modulating the quality of segmentation results but leads to similarly good solutions for a broad range of values (Figure 6.6(a), Figure 6.7), and (*iii*) results on input data that are subject to a certain amount of intrinsic or synthetically added noise lead to better segmentations than DenoiSeg trained on essentially noise-free raw data (Figure 6.6(b)).

We reason that the success of our proposed method originates from the fact that similar "skills" are required for denoising and segmentation. The segmentation task can profit from denoising and performs even better when jointly trained within the same network. When a low number of annotated images are available, denoising is guiding the training and the features learned from this task, in turn, facilitate segmentation.

Since DenoiSeg is crucially depending on Noise2Void, we also inherit the limited applicability from it. More concretely, DenoiSeg will perform best when the noise in the input data is conditionally pixel-independent [40].

Users of DenoiSeg should further be aware that the used segmentation labels need to sample all structures of interest. While we show that only a few labeled images can suffice, for more diverse structures and datasets it might be needed to choose the labeled images wisely (manually or even with automated approaches [138]).

We believe that DenoiSeg offers a viable way to enable the learning of dense segmentations when only a very limited amount of segmentation labels are available, effectively making DenoiSeg applicable in cases where other methods are not. We also show that the amount of required training data can be so little, even ad-hoc label generation by human users is a valid possibility, expanding the practical applicability of our proposed method manyfold.

II-B

Segmentation as a Label Fusion Approach

CHAPTER 7

METASEG: A FRAMEWORK FOR SEMI-AUTOMATED LABEL FUSION

The previous chapter demonstrated how our novel Deep Learning (DL) framework DenoiSeg can learn useful shared representations when trained jointly for similar tasks such as denoising and segmentation. As a consequence, it learns to perform good quality segmentation even when trained with challenging noisy microscopy datasets using very limited quantity of segmentation annotations for training. Although co-learning denoising and segmentation tasks provides a significant boost in the performance of segmentation networks in low training data regime, there is plenty of room for improvement. For instance, the segmentation performance of networks trained with large amount of training data is still far superior compared to the performance of DenoiSeg trained with limited annotations. But since obtaining large body of annotations is cumbersome and time consuming, we are constrained to use limited quantity of annotated training data. As a compromise, we often get faulty segmentations in different spatial and temporal regions.

In this chapter, we introduce an alternative segmentation framework called MetaSeg which tackles the segmentation task as a label fusion problem. The advantage of viewing segmentation task from the lens of label fusion is that faulty segmentations are not necessarily treated as adversaries anymore. Instead, if different segmentation methods make mistakes in different spatial/temporal regions, we can devise a sensible mechanism which picks out the most appropriate segmentation candidate for each region. Following this idea, using faulty poor quality segmentations only as input, MetaSeg based label fusion strives to come up with a high quality segmentation using minimal human effort.

7.1 Introduction

Real time imaging of live cells provides unprecedented insights into the behavior of cells and their morphological dynamics. Usually, in long time-lapse imaging, cells and nuclei undergo diverse changes in terms of shape, size, appearance, intensity and other morphological traits. Accurate quantification of these features requires segmentation of cells and nuclei in the entire time-lapse movies.

The spatial and temporal diversity present in long time-lapse microscopy movies poses several unique challenges for segmentation. For instance, the density of nuclei in a developing *Drosophila* embryo increases by a factor of around 4 to 5 over time during the nuclear cycle $11-14$ [139]. As another example, during division, the shape and brightness of the nuclei undergoing mitosis are distinctly different from all the other nuclei even in the same frame. Furthermore, significant variations in cell/nuclei appearance across time may also arise on account of changing experimental conditions and unforeseen technical issues.

In general, cell/nuclei segmentation task is challenging even for a set of non time-lapse images if they exhibit ample variations in imaging conditions, cell shape and size, intensity etc. from one frame to another. While it is often relatively easy to optimize an automated segmentation algorithm for a particular part of the dataset, it can be hard or even impossible to optimize it to do a good job across the whole dataset owing to the ample diversities present in the dataset.

In recent past, Deep Learning (DL) based segmentation methods have shown remarkable performance for cell/nuclei segmentation [27, 63, 121, 122, 132, 140–150]. However, training DL networks requires copious amounts of annotated ground truth (GT) data which is most often done manually and hence, is

time consuming and cumbersome. Additionally, DL methods perform well on unseen test images only if the distribution of training data is similar to that of the test data.

A natural implication is that for a DL based segmentation method to work well, the training data must be representative of the diversity present in long time-lapse microscopy movies. Hence, diverse (in space and time) image patches adequately representing the ample diversity present in microscopy datasets first need to be selected either manually or automatically [138, 151, 152] and they then need to be annotated in order to obtain good results on diverse test data. This places severe limitations for highly diverse microscopy image sequences and requires immense time, resources and human effort to produce good quality training data in the first place. Owing to these drawbacks, DL based methods alone are not suitable for segmentation of all kinds of cells/nuclei in microscopy images.

In contrast to DL methods, classical methods such as active contour based methods [153–156], watershed based methods [157–160], thresholding based methods [161–163], edge detection based methods [164,165], graph cuts [166–169] and random forest based methods [170–172] do not need high volumes of annotated GT training data and can be easily applied to diverse datasets or diverse long time-lapse microscopy image sequences. However, these methods either rely on explicit description of some prior knowledge about the data at hand or need extensive feature engineering. Besides, the performance of classical methods is below par compared to DL based methods. Hence, reliably obtaining high quality segmentation results for diverse microscopy image sequences is difficult, in general, with either DL based methods or classical methods.

However, multiple faulty segmentation for the same dataset corresponding to different segmentation methods/sources can be obtained relatively easily. Such a strategy yields a pool of segmentation hypotheses corresponding to each cell/nucleus in the dataset. As long as at least one correct segmentation hypothesis exists for each cell/nucleus we can devise an algorithm to automatically select the best possible segmentation hypothesis for each object. We refer to any such algorithm which takes multiple segmentation corresponding to the same object as input and strives to estimate a superior segmentation result compared to any of the existing faulty segmentation as a *label fusion* algorithm.

In this chapter, we introduce MetaSeg, a novel label fusion method. MetaSeg is a semi-supervised learning framework where the algorithm learns from few object level classifications by humans (binary decisions indicating if a particular hypothesis is a good or bad segmentation corresponding to a cell/nucleus) to recognize what a good/bad segmentation looks like. Finally, MetaSeg computes dense segmentation for each object in the dataset by solving a discrete optimization problem.

In the rest of the chapter, we first describe other label fusion methods in Section 7.2. The proposed MetaSeg approach and its differences to existing label fusion approaches is described in Section 7.3. Then, we experimentally show the benefits of MetaSeg based label fusion for segmentation of diverse datasets in Sections 7.4 and 7.5. Finally, Section 7.6 gives a short summary of the chapter and presents possible future directions.

7.2 Related Work

Label fusion has been explored extensively in the context of medical image segmentation [173]. In medical imaging such as Magnetic Resonance Imagining (MRI), often raw images and corresponding atlases (expert-labeled images) are available. Given a target image to segment, a popular approach is to first register the target image with an available raw image for which atlas is available and then use the deformation fields thus obtained to propagate the labels from the corresponding atlas on to the target image. However, this approach is far from perfect. Morphological differences between raw images, noise on raw images as well as errors caused during registration adversely affect label propagation. To counter

these issues, often multiple reference atlases are used for label propagation which gives rise to multiple segmentation for the same target image. The resulting segmentation pool is then used for label fusion.

Several label fusion techniques for medical image segmentation have been proposed over the years. STAPLE [174] is one of the most well known and widely used label fusion approach. STAPLE is a probabilistic fusion method which models the atlases as noisy observations of some unknown (hidden) ground truth segmentation. This method estimates a rating for the quality of each individual atlas/segmentation and using an expectation-maximization approach, it computes a probabilistic estimate of the true segmentation. Although initially proposed only for binary segmentation, STAPLE was extended to accommodate multi-class segmentation [175]. STAPLE and its variants [176–179] have achieved remarkable success for the task of label fusion in medical image segmentation.

Langerak et al. [180] proposed another popular label fusion method called SIMPLE. Contrary to STAPLE, this method iteratively compiles intermediate consensus fusion and reassigns weights of the segmentation sources based on their similarity to the current consensus. During this iterative process, a subset of atlases may also be discarded from fusion process depending on their weights. The fusion process stops when no weight change occurs. The label fusion performance with SIMPLE has been reported to be superior than STAPLE [180] while requiring significantly lesser time.

In recent years, the problem of medical image segmentation via label fusion has also been addressed using Deep Learning (DL). In [181], the authors propose VoteNet, a DL method which locally selects set of reliable atlases which are then fused via plurality voting. More recently, VoteNet++ [182] was introduced which improves over VoteNet by incorporating an additional step to correct the registration errors prior to employing label fusion.

Despite the remarkable segmentation performance of label fusion methods in medical imaging, none of these methods have been applied to microscopy image segmentation. The reason for this may be attributed to the fact that microscopy image segmentation present different challenges compared to medical image segmentation. For instance, usually medical imaging such as MRI does not involve long time-lapse imaging whereas live imaging spanning many hours and days is a very common practice in microscopy. Hence, the amount of data that needs to be segmented in microscopy can be far more voluminous compared to medical imaging datasets. STAPLE, VoteNet and their variants are computationally very time consuming which is not suitable for large time-lapse microscopy datasets. Additionally, all of the methods mentioned above perform label fusion per pixel and thus, are oblivious to the object level features present in different atlases/segmentation. A behavioral comparison to manually selecting correct labels from the pool of segmentations would reveal that object level features such as area, convexity etc. of a given label play a key role in deciding if the label should be chosen or not as part of the final fused solution. Hence, performing label fusion on per-pixel basis leads to significantly worse results (see Section 7.5).

Very recently, [94] proposed to use label fusion for microscopy image segmentation. This method (referred as BIC from here on) performs a simple majority voting or weighted voting per pixel to estimate the fused solution from available pool of segmentation. However, unlike other fusion algorithms discussed so far, BIC requires expert annotated detection markers for each cell/nucleus in the image. Based on these markers, the algorithm performs a voting procedure for each pixel in every object. Thus, BIC has a notion of object in a rough sense provided by the detection markers but it still performs fusion on a per-pixel basis for each object. Additionally, obtaining detection markers for each object in long time-lapse microscopy sequences containing hundreds and thousands of objects per frame can be tedious and expensive in itself.

We present MetaSeg to address the discussed deficiencies of existing label fusion methods. To the best of our knowledge, MetaSeg is the first algorithm which performs label fusion directly on cell/nucleus level (rather than on the pixel level) which arguably is a more sensible approach for instance

segmentation of microscopy images. In contrast to other methods, MetaSeg does not require extensive parameter tuning and scales well to large datasets and time-lapse movies. Finally, unlike BIC, MetaSeg does not require GT detection markers and operates only with a pool of segmentation hypotheses and saves significant human effort.

7.3 Approach

Problem Description. Here, we will formalize our label fusion approach for a single image frame that needs to be segmented. The presented idea can be easily extended for multiple image frames by solving the label fusion problem for single frames at a time.

For each image frame, we assume to have access to segmentation maps from multiple segmentation sources. For any object of interest in the image, we therefore have multiple segmentation candidates. Each segmentation candidate is called a *segmentation hypothesis* $\hbar_i, i \in \mathbb{N}$. Each hypothesis has a binary variable x_i associated with it indicating whether this particular hypothesis is chosen ($x_i = 1$) as part of solution or not ($x_i = 0$).

To model the label fusion/segmentation task, we use the representation of a problem graph $\mathbf{G}=\{\mathbf{V},\mathbf{E}\}$, where nodes $\mathbf{V} = \bigcup_{i=1}^{m} \hbar_i$ is the set of all segmentation hypotheses and edges $\mathbf{E} \subseteq 2^{\mathbf{V}}$ represent coupling constraints between the nodes. The edge set is defined as $\mathbf{E} = \bigcup \mathscr{E}_c$ where $\mathscr{E}_c$ is the set of conflicting hypotheses. Any set of $n \geqslant 2$ hypotheses are said to be conflicting if every hypothesis in the set has a non-zero intersection with all other hypotheses in the set. The edges prohibit the selection of conflicting hypotheses. More formally,

$$\forall c \in \mathscr{E}_c : \Sigma_{i \in c} x_i \leqslant 1 \tag{7.1}$$

Let $\mathscr{X} \subseteq \{0,1\}^{|\mathbf{V}|}$ be the set of binary vectors x satisfying the constraint given by Equation 7.1. Additionally, we associate costs $\theta_i \in \mathbb{R}$ to the variable x_i corresponding to $\hbar_i \in \mathbf{V}$ as

$$\theta_i = \begin{cases} 0 & \text{if } x_i = 0 \\ \theta_i & \text{if } x_i = 1 \end{cases} \tag{7.2}$$

We can then define the optimization problem as

$$\min_{x \in \mathscr{X}} \langle \theta, x \rangle, \tag{7.3}$$

which is an Integer Linear Program (ILP) and can be solved using either off-the-shelf commercial solvers such as Gurobi [183] and CPLEX [184] or using open-source alternative solvers [185, 186]. Solving the ILP label fusion problem guarantees a globally optimal solution but the worst-case runtime complexity maybe exponential for some problem instances. For all our datasets discussed in this chapter, we find that the runtime is only few seconds per image frame.

Active Learning for cost assignment. As evident from Equation 7.3, the costs associated with segmentation hypotheses play a crucial role in the optimization setup and directly influence the quality of solutions. Hence, it is important to come up with a sensible cost assignment mechanism. We choose an active learning framework for semi-automated cost assignment. Our framework assigns costs based on image and object features of the underlying segmentation hypotheses.

Given a set of raw images and a pool of segmentation hypotheses coming from different segmentation sources, the system first randomly selects a segmentation hypothesis and presents to the user for a simple two-class classification. The user then classifies the hypothesis as either a good hypothesis or

bad hypothesis based on the prior domain knowledge on what a good hypothesis looks like and vice-versa. For instance, the user may have the notion that for a particular dataset roundish nuclei are more probable than elongated ones and hence classify an elongated nucleus as a bad candidate. This sequence is repeated until the user has not classified at least one hypothesis as good and one as bad.

As soon as at least one element has been assigned to each class, a random forest is trained with the manually classified hypotheses forming the training set. The test set consists of all other hypotheses which have not been classified yet. The features used for training random forest are mostly geometric attributes of the training hypotheses such as area, perimeter, convexity, circularity, solidity, and perimeter of the convex hull. In addition, image attributes such as sum of the raw pixel intensities on the boundary of the hypothesis normalized by perimeter of the hypothesis and sum of the raw pixel intensities inside the boundary of the hypothesis normalized by area are also used as features for random forest training. Training is performed in an iterative fashion where every additional manual classification grows the training set and shrinks the test set. The classifier is retrained using the new training set. After each training iteration, the random forest predicts a probability score $p_i \in [0, 1]$ for any hypothesis in the test set, indicating the likelihood of it being a good segment.

We also employ additional tricks while training the random forest. In order to train the random forest with the most informative hypotheses, we employ an uncertainty sampling approach. This means that while training the random forest iteratively, the system chooses those hypotheses to present to the user for classification whose class it is most uncertain about. These correspond to hypotheses with $p_i \approx 0.5$. Besides, in order to prevent class imbalance in the training set during iterative training, we oversample hypotheses likely belonging to the minority class and present them to the user for classification.

Finally, the cost θ_i for turning hypothesis $\hbar_i$ active is given as $-(log_e(p_i+\epsilon)+0.693)$ where $\epsilon = 1e-9$. Adding the offset of 0.693 sets a 0 cost for a highly uncertain hypothesis ($p_i = 0.5$), a negative cost for a hypothesis likely belonging to good class ($p_i > 0.5$) and a positive cost for a hypothesis likely belonging to bad class ($p_i < 0.5$). It should be noted that this cost assignment is particularly suitable for the objective function given by Equation 7.3. It ensures that those hypotheses which are more likely belonging to good class are also encouraged to be picked as part of final fused solution while the hypotheses quite likely belonging to bad class are actively discouraged from being chosen as part of the final solution.

The user can decide to stop the iterative training of random forest after any number of manual classification steps. After terminating iterative training, the random forest is used to predict costs for all the hypotheses in the test set as described above. The hypotheses in the training set are also included in the optimization problem with the cost for good hypotheses set to -0.693 and that for bad hypotheses are set to 20. It should be noted that setting a high positive cost for the bad hypotheses in training set is done to ensure that these hypotheses are never selected during the optimization since they have already been flagged as bad ones by the user with certainty. Equipped with costs for all hypotheses, the ILP given by Equation 7.3 is formulated and solved with Gurobi.

7.4 Experimental Details

Datasets. We consider 3 2D datasets showing cells and nuclei with different morphological characteristics and posing different challenges for segmentation. The *DenoiSeg Flywing* dataset consists of 21 images of membrane labeled developing Flywing of size 512×512 used in [97, 145]. The *Drosophila embryo* dataset from [186] is a time-lapse depicting developmental drosophila embryo consisting of 252 frames of size 256×256. Finally, we also choose the publicly available cell segmentation dataset *Fluo-N2DH-GOWT1* [187] from Cell Tracking Challenge [134] consisting of two time-lapse movies. We use one of the movies for training and the other for testing. All test images are 1024×1024 and depict

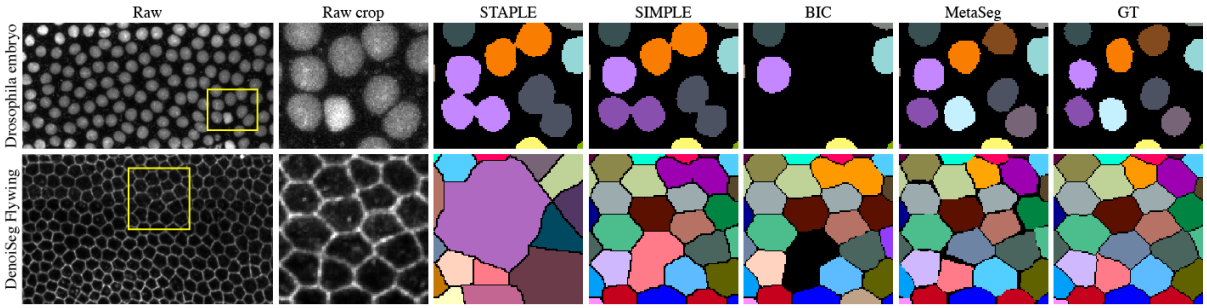

Figure 7.1: **Qualitative comparison of MetaSeg with other label fusion methods for Drosophila embryo and DenoiSeg Flywing datasets.** From left to right we show first a raw image which needs to be segmented, an interesting inset, and the various label fusion results obtained with STAPLE, SIMPLE, BIC and MetaSeg. The ground truth segmentation is shown in the last column. In line with the overall performance on these datasets, also in these qualitative examples we find that MetaSeg yields better label fusion results compared to our methods.

developing mouse stem cells. For all these datasets except *Fluo-N2DH-GOWT1* dataset, full segmentation GT are available which will be used to evaluate the label fusion performance of MetaSeg. For *Fluo-N2DH-GOWT1* dataset, only partial GT is available. For the purpose of evaluation, we manually create the full GT for test set where annotations are missing.

Multiple Segmentation Generation. An important assumption of MetaSeg is that we have access to multiple diverse segmentation hypotheses for each object that needs to be segmented in any given image.

For *DenoiSeg Flywing* and *Drosophila embryo* datasets, we generate multiple segmentations in fully unsupervised manner. To do so, we resort to DivNoising as introduced in Chapter 3. DivNoising is an unsupervised denoising method which produces diverse denoising solutions corresponding to a noisy input. If meaningful diversity is present in denoised images at the scale of cells and nuclei, segmenting these diverse images will naturally result in diverse segmentation hypotheses. As shown in [97] for *DenoiSeg Flywing* dataset, we synthetically corrupt this dataset with Gaussian noise of mean zero and standard deviation 70 and train a DivNoising denoising network. We used the trained network to predict multiple denoised solutions for each frame of the test dataset. Similarly, for each frame of *Drosophila embryo* dataset, we corrupt the images with zero mean Gaussian noise with standard deviations 50 and train a DivNoising network. During DivNoising prediction, we sample 17 and 15 diverse denoised solutions per frame of *DenoiSeg Flywing* and *Drosophila embryo* datasets respectively. For obtaining segmentation, we use automatic local thresholding on the diverse denoised solutions for each frame in the open source image analysis software Fiji [31]. We chose Niblack filter and Otsu filter with local radius of 50 for *DenoiSeg Flywing* and *Drosophila embryo* datasets respectively for thresholding followed by connected component analysis to obtain instance segmentation.

For *Fluo-N2DH-GOWT1* dataset, we train 3 different DL based segmentation algorithms, namely, 3-class U-Net [63, 132], StarDist [119], CellPose [143] and a random forest open source segmentation algorithm called Labkit available as a Fiji [31] plugin. Since each of these methods has a different working

principle, the segmentation results obtained by all these methods are sufficiently diverse. CellPose is a pre-trained network which works well for a variety of microscopy datasets and we use this directly without additional training. For training 3-class U-Net and StarDist, a separate time-lapse movie is available, but it is only partially annotated containing GT for few objects and few frames only. In each frame, we crop 160×160 patches around the cells which have GT annotations available and we thus obtain 112 patches in total. We use 95 randomly selected patches for training the DL based networks and the rest for validation. We use the default training parameters for training these networks as detailed in [144].

7.5 Results

MetaSeg outperforms state-of-the-art label fusion methods. Label fusion performance of MetaSeg is compared against the individual diverse segmentation maps used as inputs to MetaSeg. We also use well-known medical label fusion methods such as STAPLE [174] and SIMPLE [180] as our baselines. Lastly, we also compare our results against the recently proposed microscopy label fusion method BIC [94] which additionally needs GT detection markers for objects of interest as input. Note that MetaSeg, STAPLE and SIMPLE do not require any GT markers and thus the performance comparison with BIC is not completely fair. Still we want to investigate if MetaSeg can achieve on par performance or even outperform BIC. We compare all results in terms of F1 [188], SEG [134] and Average precision (AP) [133] metrics.

For *Drosophila embryo* dataset, Table 7.1 illustrates that MetaSeg is on par with BIC in terms of F1 score, is second best to BIC in terms of AP score and comes third with respect to SEG score after SIMPLE and BIC. For the other two datasets, MetaSeg significantly outperforms all baselines including BIC for most metrics as shown in Table 7.1 and Table 7.3. A qualitative comparison of different label fusion methods for *Drosophila embryo* and *DenoiSeg Flywing* datasets is illustrated in Figure 7.1.

A quantitative comparison with STAPLE could not be made on the entire datasets because of the long run time of STAPLE. We set a hard termination threshold of 4 hours for STAPLE (all other methods including MetaSeg have an execution time of few minutes at most). STAPLE could perform label fusion for only $1, 6$ and 76 frames only for *Fluo-N2DH-GOWT1*, *DenoiSeg Flywing* and *Drosophila embryo* datasets respectively within the stipulated time of 4 hours. For fair comparison with STAPLE, we also computed the segmentation metrics of MetaSeg for the same frames which STAPLE managed to segment. The reported scores for different datasets in Table 7.4 indicate that MetaSeg outperforms STAPLE for all datasets on all metrics while needing significantly lesser time (exact runtimes reported in the following subsection).

MetaSeg needs very few human annotations. As demonstrated in Figure 7.2(a) for *Fluo-N2DH-GOWT1* dataset, the performance of MetaSeg improves as the number of human annotations (yes/no classification) increases. It is seen that for this dataset around 100 annotations are sufficient to achieve good quality results in terms of F1 and AP scores. This corresponds to only about 0.005% of the total number of labels present in the pool of segmentations and it takes only about 5 minutes to annotate them. This shows that with relatively low number of annotations and minimal human effort, it is possible to obtain high quality segmentation solutions with MetaSeg. For the *DenoiSeg Flywing* and *Drosophila embryo* datasets, best performance in terms of segmentation metrics is achieved for around $150 - 200$ annotations. This corresponds to around $15 - 20$ minutes of annotation time for either of the datasets and constitutes only about 0.09% and 0.04% of the total number of labels present in the pool of segmentations for *DenoiSeg Flywing* dataset and *Drosophila embryo* dataset respectively. In contrast, STAPLE did not terminate on either of the datasets within 4 hours. While SIMPLE is much faster

Drosophila embryo			
Segmentation	SEG	AP	F1
Seg. 1	0.698	0.845	0.769
Seg. 2	0.698	0.845	0.768
Seg. 3	0.698	0.846	0.770
Seg. 4	0.699	0.849	0.770
Seg. 5	0.698	0.846	0.769
Seg. 6	0.699	0.846	0.770
Seg. 7	0.699	0.849	0.771
Seg. 8	0.700	0.847	0.770
Seg. 9	0.698	0.843	0.768
Seg. 10	0.698	0.846	0.770
Seg. 11	0.699	0.845	0.769
Seg. 12	0.699	0.845	0.769
Seg. 13	0.699	0.847	0.769
Seg. 14	0.698	0.847	0.769
Seg. 15	0.699	0.845	0.769
STAPLE	-	-	-
SIMPLE	**0.732**	0.904	0.809
BIC	0.724	**0.932**	**0.818**
MetaSeg	0.718	0.925	**0.818**

TABLE 7.1: **Label fusion performance of MetaSeg for *Drosophila embryo* dataset.** The individual segmentation maps (Seg.) are used as inputs to the label fusion algorithms STAPLE, SIMPLE, BIC and MetaSeg. The segmentation performance in terms of SEG, AP and F1 are shown. MetaSeg is on par with BIC in terms of F1 score, is marginally behind BIC in terms of AP score and comes close third with respect to SEG score after SIMPLE and BIC. Note that BIC uses additional GT detection markers which MetaSeg does not need and is better than SIMPLE for most metrics and other datasets (also see Table 7.2 and Table 7.3). STAPLE could not be run on the entire dataset due to long runtimes. A comparison with STAPLE on a subset of images is presented in Table 7.4.

and only takes at most about 30 seconds for any of these datasets, its performance is generally worse compared to MetaSeg (see Table 7.1, Table 7.2 and Table 7.3). Finally, BIC takes about 4 minutes, 9 minutes and 3 minutes for the *DenoiSeg Flywing*, *Drosophila embryo* and *Fluo-N2DH-GOWT1* datasets respectively and its performance usually lags behind MetaSeg.

MetaSeg indicates which segmentation methods are more important. MetaSeg can also be used for assessing which segmentation routines are the most important for the purpose of fusion. In the absence of GT segmentation, this information can enable a user to determine which segmentation routines need to be trained for any dataset and which ones can potentially be discarded without much loss of final fusion performance, thereby, potentially saving valuable time for users in the future. This

DenoiSeg Flywing			
Segmentation	SEG	AP	F1
Seg. 1	0.686	0.731	0.712
Seg. 2	0.685	0.728	0.711
Seg. 3	0.684	0.729	0.711
Seg. 4	0.685	0.730	0.712
Seg. 5	0.685	0.732	0.711
Seg. 6	0.688	0.737	0.714
Seg. 7	0.686	0.731	0.713
Seg. 8	0.685	0.728	0.712
Seg. 9	0.685	0.729	0.712
Seg. 10	0.684	0.729	0.710
Seg. 11	0.684	0.725	0.709
Seg. 12	0.684	0.729	0.711
Seg. 13	0.686	0.733	0.712
Seg. 14	0.686	0.732	0.713
Seg. 15	0.681	0.724	0.706
Seg. 16	0.687	0.732	0.714
Seg. 17	0.686	0.730	0.712
STAPLE	-	-	-
SIMPLE	**0.742**	0.850	0.798
BIC	0.674	0.801	0.733
MetaSeg	0.729	**0.880**	**0.802**

TABLE 7.2: **Label fusion performance of MetaSeg for *DenoiSeg Flywing* dataset.** The individual segmentation maps (Seg.) are used as inputs to the label fusion algorithms STAPLE, SIMPLE, BIC and MetaSeg. The segmentation performance in terms of SEG, AP and F1 are shown. MetaSeg outperforms not only individual segmentation maps but also all label fusion methods with respect to AP and F1 metrics and only marginally second best on SEG score. STAPLE could not be run on the entire dataset due to long runtimes. A comparison with STAPLE on a subset of images is presented in Table 7.4.

information can be mined by looking into the final fused solution and querying which segmentation sources contribute more to the fused solution than the others. For instance, we show the respective contributions of different segmentation methods to the final fused solution of MetaSeg for *Fluo-N2DH-GOWT1* dataset in Figure 7.2(b). We find that DL methods contribute about 80% hypotheses to the fused MetaSeg solution with StarDist being the most prominent source contributing with 29.60% hypotheses, followed by 3-class U-Net's contribution of 27.64% hypotheses and the segmentations from CellPose making up 22.63% of the total hypotheses.

Fluo-N2DH-GOWT1			
Segmentation	SEG	AP	F1
3-class U-Net	0.767	0.709	0.694
StarDist	0.686	0.761	0.852
CellPose	0.780	0.923	0.851
Labkit	**0.810**	0.756	0.712
STAPLE	-	-	-
SIMPLE	0.792	0.940	0.862
BIC	0.789	0.936	0.860
MetaSeg	**0.810**	**0.965**	**0.883**

TABLE 7.3: **Label fusion performance of MetaSeg for *Fluo-N2DH-GOWT1* dataset.** The individual segmentation maps from Labkit, 3-class U-Net, StarDist and CellPose are used as inputs to the label fusion algorithms STAPLE, SIMPLE, BIC and MetaSeg. The segmentation performance in terms of SEG, AP and F1 are shown. MetaSeg outperforms not only individual segmentation maps but also all label fusion methods with respect to all considered metrics. STAPLE could not be run on the entire dataset due to long runtimes. A comparison with STAPLE on a subset of images is presented in Table 7.4.

Drosophila embryo			
Segmentation	SEG	AP	F1
STAPLE	0.900	0.772	0.836
MetaSeg	**0.913**	**0.782**	**0.843**
DenoiSeg Flywing			
STAPLE	0.388	0.280	0.379
MetaSeg	**0.877**	**0.723**	**0.797**
Fluo-N2DH-GOWT1			
STAPLE	0.795	0.777	0.795
MetaSeg	**0.869**	**1.0**	**0.929**

TABLE 7.4: **Comparison of label fusion performance of MetaSeg with STAPLE.** Owing to the long run time of STAPLE, it could only be tested on a subset of images in all datasets (see main text for exact number of images in the subsets corresponding to each dataset). The segmentation metrics for the same subsets obtained with MetaSeg is reported. MetaSeg outperforms STAPLE for all considered metrics for all datasets while taking only few minutes compared to many hours for STAPLE (see main text for exact runtimes).

7.6 Conclusion

In this chapter, we presented MetaSeg, a proof-of-concept label fusion algorithm to merge potentially faulty but diverse segmentation labels from different segmentation sources to estimate a higher

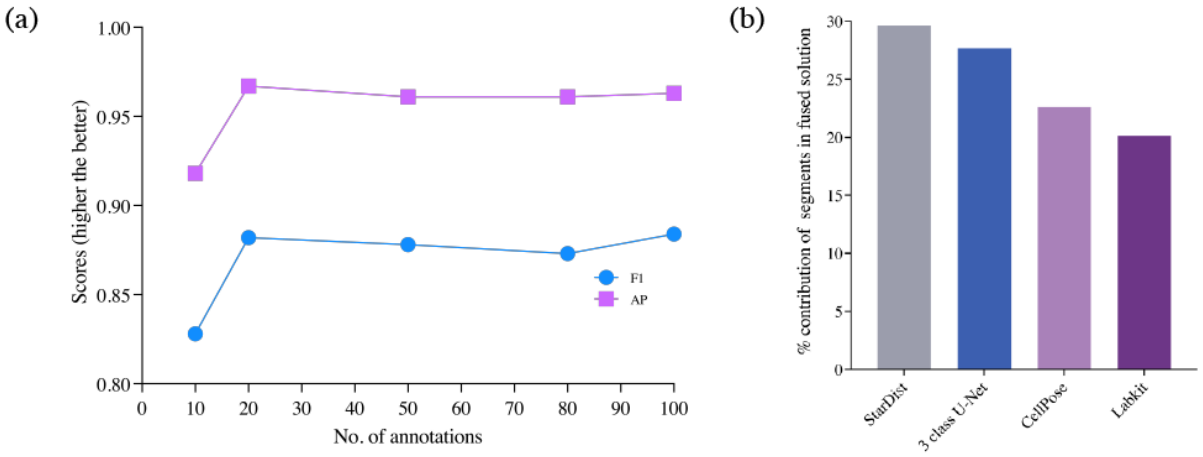

Figure 7.2: **MetaSeg performance on *Fluo-N2DH-GOWT1* dataset with number of annotations and contribution analysis of different segmentation sources to MetaSeg final solution. (a)** MetaSeg performance improves with number of manual annotations in terms of both F1 and AP scores. With around 100 annotations ($\approx 0.005\%$ of total labels), peak performance is achieved. This corresponds to only about 5 minutes of annotation time. **(b)** MetaSeg can also be used to quantify the relative importance of segmentation algorithms for the fusion task. Deep Learning based methods (StarDist, 3 class U-Net and CellPose) are the biggest contributors to the fused solution.

quality segmentation result compared to any of the individual segmentations. Our method relies on the assumption that a good segmentation hypothesis corresponding to any cell/nucleus is present in the given pool of segmentation hypotheses. We then present an active learning framework which learns a score assignment for all segmentation hypotheses from limited human interactions. This method relies on object level geometric priors as well as raw image level priors to learn a sensible cost assignment. This is a notable difference compared to other existing label fusion methods which only rely on given segmentation maps to compute the fused solution and do not have a notion of object as a whole while performing label fusion. Armed with sensible costs for all segmentation hypotheses obtained by our active learning module, we cast the label fusion problem as an Integer Linear Program (ILP) and the solution to this problem is the merged segmentation.

We showed that our method yields high quality segmentation results and outperforms other baseline fusion methods as well as different segmentation routines by a large margin with respect to different segmentation metrics on multiple microscopy datasets. MetaSeg offers a huge potential for microscopy image segmentation where annotated data for training state-of-the-art DL based segmentation methods is scarcely available. DL methods trained with limited training data perform poorly but as long as different segmentation methods make generalization mistakes in different temporal/spatial regions, MetaSeg can be utilized to fuse their solutions, thereby saving precious annotation time. Additionally, for all datasets tested so far, we observed that the run-time of MetaSeg (including human interactions and ILP solving time) is in the order of at most $10-15$ minutes. Thus, with relatively less effort, MetaSeg can be used to generate high quality segmentation results.

It is important to emphasize that MetaSeg cannot come up with a segmentation hypothesis unless it is already present in the pool of segmentation hypotheses. This means that if a desired candidate hypothesis for any cell/nucleus is not present in the pool of segmentation hypotheses, MetaSeg will not be able to come up with a good segmentation estimate for such objects. Furthermore, we only showcased the efficacy of MetaSeg on 2D datasets. But the idea is equally applicable for arbitrary dimensional images. In the future, evaluating MetaSeg performance on other 2D and 3D datasets will be useful to explore its true capabilities. Most importantly, an extension of our framework to enable interactive proofreading and curation of the final fused solution will be a valuable contribution. The idea is that once a user indicated an error in the fused solution, the random forest learns from the curation and the optimization routine restarts and finds the best overall solution that fixes the error pointed out by the user and other similar errors elsewhere in the dataset. We are hopeful that this will be a valuable addition and each curation iteration will fix a number of similar errors, hence saving additional curation cycles.

CHAPTER 8

CONCLUSIONS AND OUTLOOK

In this chapter, we conclude the thesis by providing a brief summary of the key results and discussing some open questions.

8.1 Conclusions

In the face of ever increasing image data acquired through modern microscopy methods, there is an undeniable necessity for reliable and robust (semi-) automated image analysis techniques to save manual effort. This thesis primarily focuses on two key image processing tasks - image restoration (Chapters 2, 3 and 4), and image segmentation (Chapters 5, 6 and 7).

We started by looking at the problem of unsupervised image denoising. Pixel-wise noises such as Gaussian and Poisson noise, are the dominant and most common sources of noise in biomedical imaging and photography alike. Of the existing Deep Learning (DL) based methods for unsupervised image denoising introduced in Chapter 2, the probabilistic denoising method Probabilistic Noise2Void (PN2V) [53] is the current state-of-the-art. However, this method requires a noise model probabilistically describing the noise which is modeled as a histogram obtained from a set of calibration images. We showed in Chapter 2 that histogram based noise models are not ideal and leave room for improvements. To counter the deficiencies of histogram based noise model, we introduced Gaussian Mixture Model (GMM) based noise models and experimentally showed that PN2V setups trained with GMM based noise models are consistently better than those trained with their histogram based counterparts. Additionally, we introduced a bootstrapping mechanism to obtain noise models without needing any calibration data at all. This rendered PN2V fully unsupervised requiring only noisy images to be denoised.

Although our proposed fully unsupervised PN2V establishes a new state-of-the-art for pixel-wise noise removal, it does not account for the fact that image denoising is an inverse problem with a non-unique solution. In fact, all existing denoising methods address this problem by only predicting a single denoised solution which is a compromise between all possible solutions depending on the objective function being optmized. For instance, denoisers optimizing the squared error loss between the reconstructions and the inputs find the Minimum Mean Squared Error (MMSE) estimate. To address this fundamental problem, we introduced a diversity denoising framework called DivNoising in Chapter 3 which trains a fully convolutional Variational Autoencoder (VAE) to learn an approximate posterior distribution of clean images given only a body of noisy data. During inference, DivNoising can be used to sample multiple plausible solutions corresponding to a noisy image and different point estimates such as MMSE and Maximum A Posteriori (MAP) estimate can be obtained at no additional training cost. Furthermore, we also showed that DivNoising not only establishes a new state-of-the-art for pixel-wise noise removal but the diverse denoising samples from DivNoising posterior greatly benefit downstream applications such as cell segmentation and Optical Character Recognition.

In Chapter 4, we build further on the idea of diversity denoising and present an improved architecture for the same called Hierarchical DivNoising. This architecture based on hierarchical VAEs [104–108] massively outperforms the denoising performance of DivNoising not only on biomedical benchmark datasets but also on natural image denoising benchmarks which DivNoising is not suitable for. Besides, for the first time, we investigate the performance of VAE based denoising models for removal of spatially dependent structured noise and artefacts common in microscopy and medical imag-

ing. We showed that generative diversity denoising (GDD) models such as vanilla VAEs, DivNoising and Hierarchical DivNoising are capable of removing structured noises and artefacts. We rigorously investigated factors which modulate the structured noise removal capabilities of GDD models. Our experiments established GDD methods as the new state-of-the-art for structured noise removal in microscopy and medical imaging.

Chapter 5 marks the beginning of the second part of the thesis which deals with microscopy image segmentation task. DL has arguably emerged as the method of choice for the detection and segmentation of biological structures in microscopy images. However, DL typically needs copious amounts of annotated training data that is for biomedical projects typically not available and excessively expensive to generate. Additionally, tasks become harder in the presence of noise, requiring even more high-quality training data. Hence, in Chapter 5 we proposed to use self-supervised Noise2Void denoising networks to improve the performance of other DL-based image segmentation methods. More specifically, we studied the suitability of learned representations by self-supervised Noise2Void denoising networks and presented ideas to use these representations for improving cell/nucleus segmentation in microscopy data. We proposed two stage pipelines where we first trained Noise2Void denoising networks in fully unsupervised fashion and then used them with suitable modifications in frameworks akin to transfer learning to improve segmentation despite having access to limited ground truth annotations for the segmentation task. Using two state-of-the-art segmentation baseline methods, U-Net [63, 132] and StarDist [27], we showed that our ideas consistently improve the quality of resulting segmentations, especially when only limited training data for noisy micrographs are available.

Motivated by the success of the ideas presented in Chapter 5, we harnessed the benefits of self-supervised denoising for segmentation further in Chapter 6 by proposing a new method called DenoiSeg for joint denoising and segmentation. Unlike the ideas presented in Chapter 5 which used two stage training pipelines and trained denoising and segmentation networks separately, DenoiSeg can be trained end-to-end on only a few annotated ground truth segmentations. We achieved this by extending Noise2Void denoising scheme that can be trained on noisy images alone, to also predict dense 3-class segmentations. The network becomes a denoising expert by seeing all available raw data, while co-learning to segment, even if only a few segmentation labels are available. DenoiSeg offers a viable way to circumvent the tremendous hunger for high quality training data and effectively enables learning of dense segmentations when only very limited amounts of segmentation labels are available.

Finally, Chapter 7 tackles the problem of image segmentation from a fundamentally different perspective of label fusion. Here, we make the assumption that many poor quality segmentation can be achieved from multiple segmentation sources relatively easily. For instance, using a random forest based segmentation model like Ilastik [173] and a DL based model such as U-Net [63] on the same dataset will very likely give segmentation results of different quality and will most likely also make errors in different spatial/temporal regions of the dataset. We introduce a label fusion method called MetaSeg which takes advantage of the diversity present in a pool of segmentation from different sources of the same dataset and comes up with a higher quality segmentation compared to any of the individual segmentation. In contrast with existing label fusion methods in literature [94, 174, 179–182] which operate on one pixel at a time for fusion, MetaSeg is the first such technique which uses the notion of object instances and object features for fusion. This unique ability allows MetaSeg to obtain superior fusion results while requiring considerably lesser time compared to most other label fusion methods and thereby saving significant manual effort for obtaining high quality segmentation results.

In general, we have presented many different methods and ideas for self-supervised image restoration, and image segmentation using very limited quantities of ground truth annotations or no ground truth annotation at all. We have experimentally demonstrated the applicability and efficacy of our methods from a practical point of view in biomedical image analysis. It is the hope that some of the ideas

presented in this thesis will directly benefit the biomedical practitioners and will be used as ingredients for building tools and methodologies in other computer vision systems.

8.2 Future Work

The ideas presented in this thesis bring to the fore some interesting questions for future research.

Learning disentangled representations for systematic structured noise removal. In the context of image restoration, we have empirically explored the structured noise removal capability of generative diversity denoising (GDD) models. We observed that under certain conditions, GDD models are inherently capable of removing structured noises. However, we did not come up with a well-defined recipe which allows the GDD models to explicitly learn a distinction between signal/structure of interest which we care about and wish to preserve and structured noise/artefacts which are undesirable and we wish to remove. One potential approach for this problem maybe to explicitly learn disentangled representations for structures of interest and structured noises.

Disentangled representation learning refers to learning of representations of generative factors in the data [189] which are independent of each other. If structures of interest and structured noises/artefacts can be encoded in mutually exclusive latent sub-spaces of GDD models, then we may systematically remove structured noises by suppressing the sub-space of the latent space corresponding to structured noise encoding. Among other approaches for disentangled representation learning for generative models [189–191], contrastive learning [192, 193] based approaches have found recent success [194–197]. Although these approaches are designed with the purpose of image generation, adopting similar approaches for learning disentangled representations for structures of interest and structured noises would be an interesting and valuable contribution towards systematic removal of structured noises with GDD models.

Generating samples from different modes of learned posterior distribution. We showed in Chapter 3 that given noisy input data, diverse denoised samples generated by a well trained DivNoising model can be used for multiple downstream processing tasks such as for ambiguity removal in applications such as Optical Character Recognition/text understanding and object segmentation, etc. For such applications, it is critical to be able to generate diverse interpretations for the same input.

However, randomly drawn samples from the learned posterior are not constrained to maximize diversity. Additionally, the learned posterior can be multi-modal and there might be a dominant mode that prevents efficient sampling from other modes. At the same time, we might still want to avoid sampling highly improbable interpretations.

Finding efficient sampling schemes will also enable efficient inference of point estimates such as *maximum a posteriori* (MAP) estimates. In Chapter 3, we have demonstrated a clustering based approach for MAP estimation given a pool of samples from the learned posterior. But unfortunately, this approach also requires the sampling of many interpretations in the first place. Hence, more research in this direction is of very high practical value.

Further development of ideas presented in MetaSeg. We presented the idea of MetaSeg as a label fusion approach to generate high quality segmentation from a pool of poor quality but diverse segmentations. This idea, in its current form, exists mostly as a proof of concept and further evaluation is needed on different 2D and 3D datasets. It would also be worth investigating to explore better active learning schemes for iterative random forest training in MetaSeg. Maybe most importantly, it would

be highly relevant practically to enable proofreading and interactive curation of the fused solutions in MetaSeg. The basic idea is that users should have the possibility to indicate any error in the current solution. Based on the interactive user curation, the random forest can learn from the curation and the label fusion optimization routine should restart and find the best overall solution that fixes not only the error pointed out by the user but other similar errors that may exist in other parts of the solution. This will be a valuable addition which will potentially save additional curation cycles and lead to even better fusion results.

APPENDICES

Appendix A

Supplementary material for Chapter 3

A.1 Intrinsically Noisy Microscopy Data

We use public microscopy datasets which show realistic levels of noise, introduced by the respective optical imaging setups. The *FU-PN2V Convallaria* [53, 58] data, consists of 100 noisy calibration images (intended to generate a noise model), and 100 images of size 1024×1024 showing a noisy Convallaria section. The *FU-PN2V Mouse nuclei* [58] data is composed of 500 noisy calibration images and 200 noisy images of size 512×512 showing labeled cell nuclei. The *FU-PN2V Mouse actin* [58] data from the same source consists of 100 noisy calibration images and 100 noisy images of size 1024×1024 of the same sample, but labeled for the protein actin. Finally, we use all 3 channels of 2 noise levels (avg1 and avg16) of the *W2S* [198] data. For each channel, corresponding high quality (ground truth) images are available. Each channel's training and test sets consist of 80 and 40 images, respectively. All images are 512×512 pixels in size.

A.2 Data Exposed to Synthetic Noise

We use the well known *MNIST* [199] as well as the *KMNIST* [200] dataset showing 28×28 images of handwritten digits and phonetic letters of hiragana, respectively. Both datasets contain 60000 training examples and 10000 test examples. Onto both datasets we added pixel-wise independent Gaussian noise with $\mu = 0$ and $\sigma = 140$. As a third text-based dataset we rendered the freely available *eBook* "The Beetle" [201] and extracted 40800 image patches of size 128×128. We separated 34680 patches for training and 6120 patches for validation, and added pixel-wise independent Gaussian noise with $\mu = 0$ and $\sigma = 255$. Additionally, we use three datasets from microscopy. The *DenoiSeg Mouse* [145] data, showing cell nuclei in the developing mouse skull, consists of 908 training and 160 validation images of size 128×128, with additional 67 images of size 256×256 for testing. Two noisy datasets were created with this data, one by exposing all images to pixel-wise independent Gaussian noise with $\mu = 0$ and $\sigma = 20$ and another one by first applying poisson noise with $\lambda = 1$ followed by adding gaussian noise with $\mu = 0$ and $\sigma = 10$ followed by randomly changing 3% of pixels to either 0 or 255. This dataset is called Mouse s&p in Table 3.1. The *DenoiSeg Flywing* [145] data is showing membrane labeled cells in a fly wing, consisting of 1428 training and 252 validation patches of size 128×128, with additional 42 images of size 512×512 for testing. We exposed this data to pixel-wise independent Gaussian noise with $\mu = 0$ and $\sigma = 70$ to create a synthetic low SNR version. All original datasets are 8-bit. Lastly, we randomly select 500 images of size 384×286 from BioID Face recognition database [202] and corrupt them with pixel-wise

independent Gaussian noise with $\mu = 0$ and $\sigma = 15$. We use $340, 60$ and 100 images for training, validation and test respectively.

A.3 Training and Network Details

Here, we provide additional details about the network architecture and training parameters used throughout Chapter 3. For all DivNoising experiments, we use rather lightweight depth 2 and depth 3 VAE architectures (see Appendix Figs. Figure I for a schematic of our depth 2 network). All networks use a single input channel and 32 feature channels in the first network layer except for the network trained on mouse s&p dataset which uses 96 feature channels in the first network layer. We use two 3×3 convolutions (with padding 1), each followed by ReLU activation, followed by a 2×2 max pooling layer. After each such downsampling step, we double the number of feature channels. For all experiments we use a network architecture of depth 2 (with 2 down/upsampling steps). The only exceptions are our experiments on *DenoiSeg Flywing* and *eBook* data, for which we use a depth 3 architecture (with 3 down/upsampling steps). In total, our depth 2 networks have only around $200k$ parameters and depth 3 networks have around $700k$ parameters.

While we generally use a VAE bottleneck of 64 latent space feature dimensions for each pixel of the image (after encoding), for the small 28×28 MNIST and KMNIST images we use only 8 such latent space dimensions.

We consistently use 8-fold data augmentation (rotation and flipping) in all experiments. All networks are trained with a batch size of 32 and an initial learning rate of 0.001. The learning rate is multiplied by 0.5 if the validation loss does not decrease for 30 epochs.

For all datasets other than MNIST and KMNIST, we extract training patches of size 128×128, and separate 15% of all patches for validation. We set the maximum number of epochs such that approximately 22 million steps are performed, and in each epoch the entire training data is being fed. Training is terminated if the validation loss does not decrease by at least 10^{-6} over 100 epochs.

For *DenoiSeg Flywing* we observed KL *vanishing* and solved it via *Annealing* within the first 15 epochs [109].

The fully unsupervised DivNoising decoder directly predicts the signal and the noise variance per pixel where the variance is constrained to linearly depend on the signal (See Section 3.4). To avoid numerical problems and ensure that the predicted variance always remains positive, we allow the user to set a minimum allowed variance/standard deviation ${\sigma^2}_{\text{min}}/{\sigma}_{\text{min}}$, and enforce this by clamping the predicted values. Note that a viable choice for this parameter depends on the intensity range of the dataset. We use the following values: For all *FU-PN2V* datasets ${\sigma}_{\text{min}} = 50$, for *DenoiSeg Flywing* and *DenoiSeg Mouse*

Appendix A (Continued)

datasets $\sigma_{\text{min}} = 3$, for *DenoiSeg Mouse s&p* dataset $\sigma_{\text{min}} = 1$, for *BioID Face* dataset $\sigma_{\text{min}} = 15$, for *W2S avg 1* datasets $\sigma_{\text{min}} = 25$ and for *W2S avg 16* datasets $\sigma_{\text{min}} = 3$.

Run Time and Hardware Requirements. DivNoising using light weight fully convolutional networks (see Appendix Fig. Figure I) runs on relatively cheap computational budget. Our depth 2 networks trained for all experiments requires about 1.8 GB GPU memory and our depth 3 networks roughly 5 GB GPU memory on a *NVIDIA TITAN Xp* GPU. The training time varied from $5 - 12$ hours on average depending on the dataset.

A.4 Clustering of Solutions and Deriving the MAP Estimate

Here we provide additional details on how the cluster centers and the approximate MAP estimate of Figure 3.5 (see main text) were found. We first drew 10000 sampled images from the approximate posterior as described in Chapter 3. We then performed mean shift [85] clustering (using the existing *scipy* implementation) on the cropped image region shown in the figure. We set a *bandwidth* of 800 and the the *maximum number of iterations* to 20, and used the 100 first samples of DivNoising as seeds. We finally show 9 of the resulting cluster centers in the figure.

To produce the MAP estimate, we employ a similar strategy. In order to find the mode of the sampled distribution efficiently, we assume that dependencies in the predicted samples should be local. This assumption is valid, since our network only has only a finite receptive field for each predicted pixel. Hence, we apply mean shift algorithm on locally overlapping regions. We use a window size of 10×10 pixels with an overlap of 3 pixels in x and y. On each such region, the mean shift algorithm is executed repeatedly with decreasing bandwidth, always using the latest result as new seed. We start by using the sample mean as seed and with an initial *bandwidth* of 200. After each iteration the bandwidth is decreased by a factor of 0.9, until it drops below 100.

Similar results should also be achievable by applying mean shift algorithm on the entire image. But since samples will differ at any location in the image, this global approach would require an excessively large number of DivNoising samples.

A.5 Instance Cell Segmentation

Here, we provide additional details regarding the downstream segmentation task described in Section 3.5 of the main text. We used the first 21 images in the test set of *DenoiSeg Flywing* for our analysis.

Given an input image, our segmentation pipeline consists of (*i*) generating segmentation masks using local thresholding with a mean filter of radius 15, followed by (*ii*) skeletonizing the space between these masks, followed by (*iii*) connected component analysis to obtain instance segmentation.

Appendix A (Continued)

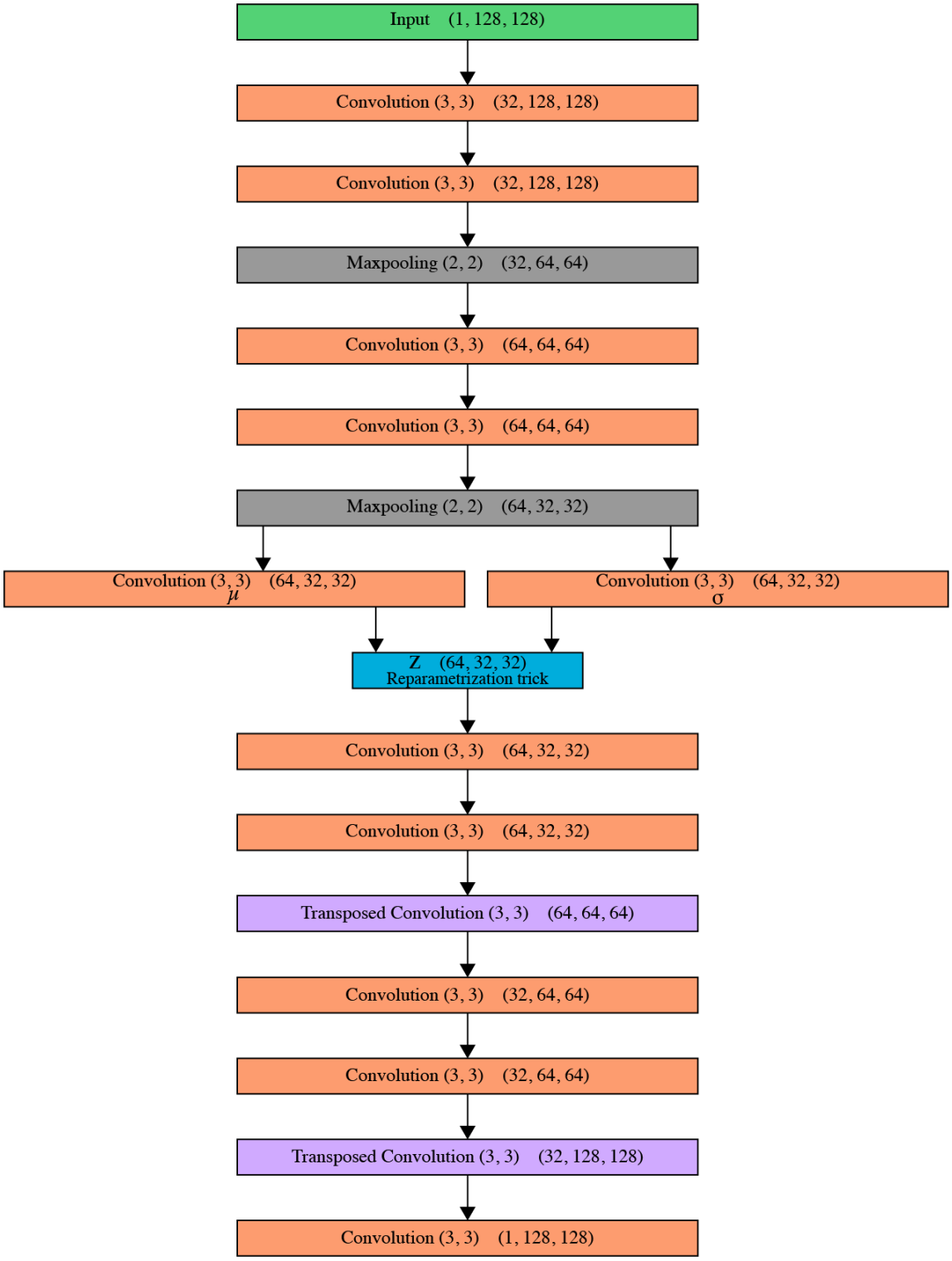

Figure I: **The fully convolutional architecture used for depth 2 networks.** We show the depth 2 DivNoising network architecture used for *FU-PN2V Convallaria, FU-PN2V Mouse nuclei, FU-PN2V Mouse actin*, all *W2S* channels and *DenoiSeg Mouse* datasets. These networks count about $200k$ parameters and have a GPU memory footprint of approximately 1.8GB on a *NVIDIA TITAN Xp*.

Appendix A (Continued)

Using this pipeline, we generated segmentation for the noisy (low SNR) images, ground truth (high SNR) images, as well as for the DivNoising MMSE estimate (obtained by averaging 1000 sampled denoised images).

We also apply the above described pipeline for each of the 1000 DivNoising samples separately to serve as input for the two label fusion methods, namely (*i*) *Consensus (BIC)*, and (*ii*) *Consensus (Avg)*. For the latter label fusion method we skip the connected component analysis and directly average the thresholded and skeletonized images. To obtain the final result, we again apply the full segmentation pipeline described above to this average image.

All segmentations were obtained with the open source image analysis software Fiji [203].

A.6 The Relative Importance of the KL Loss Component

We can generalize our DivNoising training loss as a weighted combination of a modified reconstruction loss (see Section 3.4 in the main text) and KL divergence loss, where the two loss components are weighted equally. Following the exposition in [189], we explore the effect of weighting the KL loss component during training with a factor β. Our modified training loss thus becomes

$$\mathscr{L}_{\phi,\theta}(\mathbf{x}) = \mathscr{L}^{\mathrm{R}}_{\phi,\theta}(\mathbf{x}) + \beta\mathscr{L}^{\mathrm{KL}}_{\phi}(\mathbf{x}), \tag{A.1}$$

where setting $\beta = 1$ gives our DivNoising setup described in Section 3.4 in the main text. Note that increasing or reducing β, *i.e.*, changing the relative importance of the reconstruction loss, is equivalent to using a wider or narrower noise model, such as a Gaussian noise model with larger or smaller standard deviation σ. We can thus interpret above results as the effect of using a mismatched noise model that is either too wide or too narrow.

Effect of β on Denoising Quality. We investigated the effect of β on the denoising ability of DivNoising network with the *DenoiSeg Flywing* dataset. As illustrated in Appendix Figure II(a), $\beta = 1$ gives the optimal results for the MMSE estimate (obtained by averaging 1000 samples). Both regimes, $\beta > 1$ and $\beta < 1$, yield sub-par denoising performance.

Appendix A (Continued)

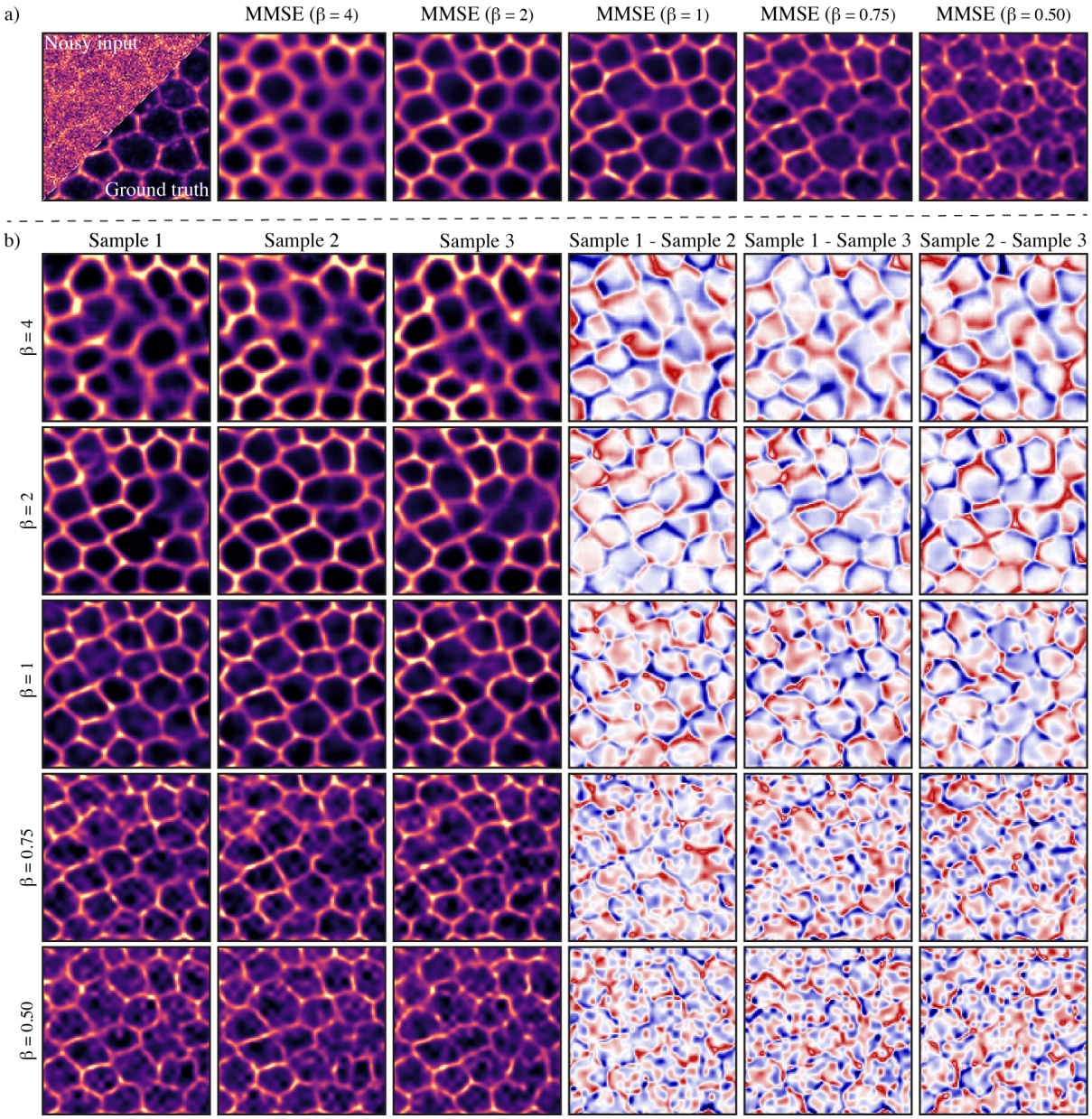

Figure II: **Qualitative analysis of the effect of weighting KL loss term with factor β for *DenoiSeg Flywing* dataset. (a)** We show the DivNoising MMSE estimate obtained by averaging 1000 samples for all considered β values (Supplementary Equation A.1). We observe that the reconstruction quality suffers on either increasing $\beta > 1$ or decreasing $\beta < 1$. Best results (with respect to PSNR) are obtained with $\beta = 1$. **(b)** For each β value, we show three randomly chosen DivNoising samples as well as difference images. Increasing $\beta > 1$, allows the DivNoising network to generate structurally very diverse denoised solutions, while typically leading to textural smoothing. Decreasing $\beta < 1$ generates DivNoising samples with overall much reduced structural diversity, introducing reconstruction artefacts/structures at smaller scales.

Appendix A (Continued)

A.7 Results on Natural Images

We investigated the denoising performance of DivNoising network on the natural images benchmark dataset $BSD68$ [204], where the input has been corrupted with Gaussian noise of $\sigma = 25$. With our depth 2 network having 96 feature channels in the first network layer, we achieve a PSNR of 27.45 dB while our unsupervised Noise2Void baseline gives 27.71 dB. As discussed in the main text, this does not come as a surprise since our DivNoising network is comparatively small and asked to learn a complete generative model of the entire data domain (see main text). Learning such a model for the tremendous diversity present in natural images is challenging, and likely the reason why other architectures solving problems posed on the domain of natural images are much larger than our networks are.

A.8 Derivation of DivNoising Loss Function from Probability Model Perspective

Here, we want to provide a more formal derivation of why our loss function can be used to train the VAE as desired. We follow a similar line of argument as has been laid out for the standard VAE by Doersch in [205].

In our framework, we assume that the observed data $\mathbf{x}$ is generated from some underlying latent variable $\mathbf{z}$ through some clean signal $\mathbf{s}$ via a known noise model $p_{\mathrm{NM}}(\mathbf{x}|\mathbf{s})$. This process of data generation is depicted as a graphical model shown in Appendix Figure III.

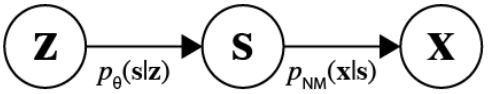

Figure III: **Graphical model of the data generation process.**

The decoder describes a full joint model for all three variables:

$$p_\theta(\mathbf{z}, \mathbf{x}, \mathbf{s}) = p(\mathbf{x}, \mathbf{s}|\mathbf{z})p(\mathbf{z}) = p(\mathbf{x}|\mathbf{s}, \mathbf{z})p_\theta(\mathbf{s}|\mathbf{z})p(\mathbf{z}) \tag{A.2}$$

Appendix A (Continued)

In the assumed graphical model in (Appendix Figure III) $\mathbf{x}$ is conditionally independent of $\mathbf{z}$ given $\mathbf{s}$. Formally, this implies that

$$p(\mathbf{x}|\mathbf{s},\mathbf{z}) = p_{\mathrm{NM}}(\mathbf{x}|\mathbf{s}). \tag{A.3}$$

Using Supp. Equation A.3, we can reformulate Supp. Equation A.2 as

$$p_\theta(\mathbf{z},\mathbf{x},\mathbf{s}) = p_{\mathrm{NM}}(\mathbf{x}|\mathbf{s})p_\theta(\mathbf{s}|\mathbf{z})p(\mathbf{z}). \tag{A.4}$$

To train the generative model from Appendix Figure III we try to adjust the parameters θ to maximize the likelihood of observing our training data $\mathbf{x}$. This means that we need to maximize

$$p_\theta(\mathbf{x}) = \int p_{\mathrm{NM}}(\mathbf{x}|\mathbf{s} = g_\theta(\mathbf{z}))p(\mathbf{z})d\mathbf{z}. \tag{A.5}$$

However, computing the integral in Supp. Equation A.5 is intractable due to the high dimensionality of $\mathbf{z}$. In our particular model, we would need to integrate over 64 dimensions for each pixel for all our datasets except MNIST and KMNIST datasets where we would need to integrate over 8 dimensions for each pixel. An alternative to computing the integral would be to approximate it by sampling a large number of values $\mathbf{z}^1, \mathbf{z}^2, ..., \mathbf{z}^K$ from $p(\mathbf{z})$ and computing $p_\theta(\mathbf{x}) \approx \frac{1}{K}\sum_{k=1}^{K} p_{\mathrm{NM}}(\mathbf{x}|\mathbf{s} = g_\theta(\mathbf{z}^k))$. However, since $p_{\mathrm{NM}}(\mathbf{x}|\mathbf{s} = g_\theta(\mathbf{z}^k))$ will be very close to 0 for almost all $\mathbf{z}^k$, this would require K to be a very large number for each image in our training set.

Following the idea introduced in [65], we overcome this problem by instead using an encoder to describe an auxiliary distribution $q_\phi(\mathbf{z}|\mathbf{x})$. The encoder can take a noisy image $\mathbf{x}$ and yield a distribution over $\mathbf{z}$ values, which in turn are likely to produce $\mathbf{x}$ under the generative model. We want the encoder distribution $q_\phi(\mathbf{z}|\mathbf{x})$ to approximate the true underlying distribution $q_\phi(\mathbf{z}|\mathbf{x}) \approx p_\theta(\mathbf{z}|\mathbf{x})$, as it is implicitly described by our graphical model. From Bayes theorem, $p_\theta(\mathbf{z}|\mathbf{x})$ factorizes as

$$p_\theta(\mathbf{z}|\mathbf{x}) = \frac{p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})}{p_\theta(\mathbf{x})}. \tag{A.6}$$

The decoder in DivNoising setup is a deterministic function of $\mathbf{z}$, i.e., $g_\theta(\mathbf{z}) = \mathbf{s}$. Hence, we can reformulate Supp. Equation A.6 as

$$p_\theta(\mathbf{z}|\mathbf{x}) = \frac{p_{\mathrm{NM}}(\mathbf{x}|\mathbf{s} = g_\theta(\mathbf{z}))p(\mathbf{z})}{p_\theta(\mathbf{x})}. \tag{A.7}$$

We can describe the quality of the encoder distribution, *i.e.*how well it approximates the true $p_\theta(\mathbf{z}|\mathbf{x})$ via the KL divergence

$$\mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})\right) = -\int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{p_\theta(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})} d\mathbf{z}. \tag{A.8}$$

Appendix A (Continued)

Substituting Supp. Equation A.7 in Supp. Equation A.8, we get

$$\begin{aligned}
\mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})\right) &= -\int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{p_{\text{NM}}(\mathbf{x}|\mathbf{s}=g_\theta(\mathbf{z}))p(\mathbf{z})}{p_\theta(\mathbf{x})q_\phi(\mathbf{z}|\mathbf{x})} d\mathbf{z} \\
&= -\int q_\phi(\mathbf{z}|\mathbf{x}) [\log \frac{p_{\text{NM}}(\mathbf{x}|\mathbf{s}=g_\theta(\mathbf{z}))p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} - \log p_\theta(\mathbf{x})] d\mathbf{z} \\
&= -\int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{p_{\text{NM}}(\mathbf{x}|\mathbf{s}=g_\theta(\mathbf{z}))p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} d\mathbf{z} + \int q_\phi(\mathbf{z}|\mathbf{x}) \log p_\theta(\mathbf{x}) d\mathbf{z} \\
&= -\int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{p_{\text{NM}}(\mathbf{x}|\mathbf{s}=g_\theta(\mathbf{z}))p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} d\mathbf{z} + \log p_\theta(\mathbf{x}) \int q_\phi(\mathbf{z}|\mathbf{x}) d\mathbf{z}.
\end{aligned}$$

Since $\int q_\phi(\mathbf{z}|\mathbf{x}) d\mathbf{z} = 1$, we get

$$\mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})\right) = -\int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{p_{\text{NM}}(\mathbf{x}|\mathbf{s}=g_\theta(\mathbf{z}))p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} d\mathbf{z} + \log p_\theta(\mathbf{x}).$$

This implies

$$\begin{aligned}
\log p_\theta(\mathbf{x}) &= \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{p_{\text{NM}}(\mathbf{x}|\mathbf{s}=g_\theta(\mathbf{z}))p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} d\mathbf{z} + \mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})\right) \\
&= ELBO + \mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})\right),
\end{aligned} \tag{A.9}$$

where $ELBO$ is the *Evidence Lower Bound* as also introduced in [82] in the context of standard VAEs and here $ELBO = \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{p_{\text{NM}}(\mathbf{x}|\mathbf{s}=g_\theta(\mathbf{z}))p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} d\mathbf{z}$. Note that the KL divergence term in Supp. Equation A.9 is always greater than or equal to 0 and hence, $ELBO$ is a lower bound for $\log p_\theta(\mathbf{x})$, i.e., $\log p_\theta(\mathbf{x}) \geqslant ELBO$. It follows from Supp. Equation A.9 that

$$ELBO = \log p_\theta(\mathbf{x}) - \mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})\right) \tag{A.10}$$

Supp. Equation A.10 implies that maximizing ELBO with respect to ϕ and θ maximizes $\log p_\theta(\mathbf{x})$ and minimizes $\mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})\right)$, the goals we seek to achieve. Hence,

$$\begin{aligned}
\max ELBO &= \max \left(\int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{p_{\text{NM}}(\mathbf{x}|\mathbf{s}=g_\theta(\mathbf{z}))p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} d\mathbf{z} \right) \\
&= \max \left(\int q_\phi(\mathbf{z}|\mathbf{x}) \log p_{\text{NM}}(\mathbf{x}|\mathbf{s}=g_\theta(\mathbf{z})) d\mathbf{z} + \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{p(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})} d\mathbf{z} \right) \\
&= \max \left(\int q_\phi(\mathbf{z}|\mathbf{x}) \log p_{\text{NM}}(\mathbf{x}|\mathbf{s}=g_\theta(\mathbf{z})) d\mathbf{z} - \mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right) \right) \\
&= \max \left(\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_{\text{NM}}(\mathbf{x}|\mathbf{s}=g_\theta(\mathbf{z}))] - \mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right) \right).
\end{aligned}$$

Appendix A (Continued)

Maximizing the $ELBO$ is equivalent to minimizing the negative $ELBO$, thus giving us the DivNoising loss function

$$\mathscr{L}_{\phi,\theta}(\mathbf{x}) = \min(\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[-\log p_{\mathrm{NM}}(\mathbf{x}|\mathbf{s} = g_\theta(\mathbf{z}))] + \mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right)), \tag{A.11}$$

where the expected value is approximated in each iteration by drawing a single sample from $q_\phi(\mathbf{z}|\mathbf{x})$. Note that the first term in the summation in Supp. Equation A.11 is the same as described in Section 3.4 in the main text whereas the second term in the summation is the same as used in the standard VAE loss.

Appendix B

Supplementary Material for Chapter 4

B.1 Datasets and Training Details

Datasets Corrupted with Pixel-wise Noise. We consider two microscopy datasets namely, the (*i*) *FU-PN2V Convallaria* dataset from [53, 58] which shows 1024×1024 images of a intrinsically real-world noisy *Convallaria* sections, and the (*ii*) *DenoiSeg Flywing* dataset from [145], showing images of a developing Flywing. Following [97], we synthetically corrupted *DenoiSeg Flywing* images with zero mean Gaussian noise of standard deviation 70. The train and validation splits for these datasets are as described in the respective publication. For the *FU-PN2V Convallaria* dataset, the test set consists of 100 images of size 512×512 while the test set for *DenoiSeg Flywing* dataset consists of 42 images of size 512×512.

From the domain of natural images, we consider two grayscale test datasets namely, (*iii*) the popular *BSD68* dataset from [204] containing 68 natural images of different sizes, and (*iv*) the *Set12* dataset from [50] containing 12 images of either size 512×512 or 256×256. Both datasets have been synthetically corrupted with zero mean Gaussian noise of standard deviation 25. For training and validation, we use a set of 400 natural images of size 180×180 as released by [50] and has later also been used in [40].

Next, we chose to use (*v*) the popular *MNIST* dataset [199] showing 28×28 images of digits, and (*vi*) the *Kuzushiji-Kanji* dataset [200] showing 64×64 images of Kanji characters. Both datasets have been synthetically corrupted with zero mean Gaussian noise of standard deviation 140. For the Kanji dataset, we randomly select 119360 images for training and 14042 images for validation. The test set contains 100 randomly selected images not present in either training or validation sets. For the *MNIST* dataset, the publicly available data has two splits containing 60000 training images and 10000 test images. We further divide the training images into training and validation sets containing 54000 and 6000 images, respectively. We take a random subset of 100 images from the test data and use this as our test set to evaluate all methods. The noisy train, validation and test images for *Kanji* and *MNIST* data will be released publicly.

Finally, we selected two datasets of faces, namely, (*vii*) the *BioID Faces* dataset [202] showing 286×384 images of different persons under different lighting conditions, and (*viii*) the *CelebA HQ* dataset at 256×256 image resolution containing faces of celebrities from [206][1]. The CelebA HQ dataset has RGB images and we take only the red channel

[1]The full dataset can be downloaded from the Google Drive link at `https://github.com/suvojit-0x55aa/celebA-HQ-dataset-download`

Appendix B (Continued)

images for our experiments. Both facial datasets have been synthetically corrupted with zero mean Gaussian noise of standard deviation 15. For the *BioID Faces* dataset, we choose 340 images in the training set and 60 images in the validation set while the test set contains 100 additional images. For the *CelebA HQ* dataset, the train and validation sets contain 27000 and 1500 images, respectively, while the test set contains 100 additional images. We will release the noisy versions of these datasets publicly.

Datasets Corrupted with Structured Noises. We used the *BioID Faces* dataset for all our experiments with synthetic structured noises. In total, we generated four structured noises (sources of artefacts):

Chessboard patterns: We first corrupted the images with zero mean Gaussian noise with standard deviation 15. Following this, we went over each 8×8 pixel groups on each image and added a pixel-wise chessboard pattern on it uniform at random with probability ϵ. The added chessboard pattern is adding either -25 and $+25$ to the underlying image pixels.

Horizontal striping patterns: We first corrupted the images with zero mean Gaussian noise with standard deviation 7. Following this, we went over adjacent groups of 20×1 pixels and added a horizontal striping pattern with probability ϵ. The added horizontal pattern is obtained by convolving a 20×1 kernel (each element of the kernel having value $1/20$) with a noise vector of the same shape and the noise elements lying in the range [0,60). Note, the added patterns are zero mean.

Moire patterns: We create an image of Moire patterns by the superposition of two sinusoidal waves with amplitudes 10. The first sinusoidal component has frequencies $45/\pi$ rad and $50/\pi$ rad in x and y directions, respectively. The second sinusoidal component has frequencies $-45/\pi$ rad and $50/\pi$ rad in x and y directions, respectively. Also, both components are phase shifted by π. We first corrupted our training images with zero mean Gaussian noise with standard deviation 15 and then added the Moire patterns uniform at random per-pixel with probability ϵ.

Computed Tomography (CT) artefacts: These artefacts are a consequence of how CT images are reconstructed from a given set of projections. We have used a CT simulator capable of projecting a given 2D image and then reconstructing a 2D image from these projections. Artefacts arise because a reconstruction based on too few projections suffers from missing data being observed. To create the data we used, we have cropped 283×283 tiles from each ground truth image in the respective training, validation and test sets. We used the CT simulator multiple times, each time specified a different number of projection angles, hence leading to artefacts of varying magnitude. More specifically, we choose $150, 180$ and 200 projection angles for CT reconstruction.

Appendix B (Continued)

B.2 Training details of Hierarchical DivNoising

Our HDN network is fully convolutional and consists of multiple hierarchical stochastic latent variables. For our experiments we use a network with 6 hierarchical latent variables, only for the *MNIST* dataset we have only used a hierarchy of height 3. Each latent group has $H_i \times W_i \times 32$ dimensions where H_i and W_i represent the dimensionality of the feature maps at layer i. The initial number of filters is set to 64 and we utilize batch-normalization [207] and dropout [67] with a probability of 0.2. Both *top-down* and *bottom-up* networks have 5 residual blocks between each latent group. A schematic of the residual blocks and the full network architecture is shown in Figure IV. A batch size of 16 was used for *BioID Faces* and *Natural* image datasets, while a batch size of 64 was used for *FU-PN2V Convallaria, DenoiSeg Flywing, Kanji* and *MNIST*. For the *CelebA HQ* dataset we used a batch size of 4. To prevent *KL vanishing* [109], we use the *free bits* approach as described in [110, 111]. For experiments on all datasets other than the *MNIST* dataset, we set the value of *free bits* to 1.0, while for the *MNIST* dataset a value of 0.5 was used.

Architecture of Hierarchical DivNoising. Figure IV shows our HIERARCHICAL DIVNOISING architecture with 3 stochastic latent variables, as used for experiments on the *MNIST* data. For all other experiments our architecture is the same, but has 6 latent variable groups instead of 3. Please note that we have used the exact same architecture for pixel-wise and structured noise removal experiments. Our networks with 6 latent groups need around 8 GB of GPU memory and were trained on a Titan Xp GPU. The training time until convergence varies from 1 day to 5 days depending on the task and dataset.

B.3 Visualizing Patterns Encoded by HDN Latent Layers and Targeted Deactivation

In this section we present a way to visualize what the individual levels of a HDN learned to represent. The idea for this way of inspecting a Ladder VAE is not new and can also be found in a GitHub repo[1] authored by a member of the team around [108].

More specifically, we want to visualize what our 6-layered HDN model has learned in Section 4.5 of the main text. We inspect layer i by performing the following steps:

- We draw a sample from all latent variables for layers $j > i$.
- We then draw 6 samples from layer i (conditioned on the samples we drew just before).

[1] `https://github.com/addtt/ladder-vae-pytorch`

Appendix B (Continued)

- For each of these 6 samples, we take the mean of the conditional distributions at layers $k < i$, *i.e.*, we do not sample for these layers.
- We generate a total of 6 images given the latent variable samples and means from above.

The results of this procedure on the HDN network used in Section 4.5 can be seen in Figure V.

Method	PSNR (dB)
HDN	29.96
HDN_{2-6}	31.24
HDN_{3-6}	31.36
HDN_{4-6}	28.13

TABLE I: **Performance of HDN networks with "deactivated" latent layers.** We show results for HDN setups (in terms of PSNR) with different layers of the original HDN architecture "deactivated" (see Appendix B.3 for how latent layers are deactivated). In line with our observations in Figure V, the HDN_{3-6} setup leads to the best performance on the data of [17], which we used here.

By visually inspecting Figure V one can observe that the bottom two layers (layers 1 and 2) learned fine details, including the line artefacts we would like to remove from input images. The other layers (layers 3 to 6), on the other hand, either learn smoother global structures or learn such abstracted concepts that the used visualization method is not very informative.

Still, the created visuals help us to understand why the full HDN leads to worse results for removing microscopy line artefacts than a DN setup did (see Table 4.2 in the main text). Owing to the high expressivity of the HDN model, the structured noise can actually be encoded in the hierarchical latent spaces and the artefacts will therefore be faithfully be represented in the otherwise denoised resulting image.

Motivated by these observations, we proposed a simple modification to HDN that has then led to SOTA results on the given benchmark dataset (see again Table 4.2 in the main text). We showed that by "deactivating" layers 1 and 2, *i.e.*, the ones that show artefact-like patterns in Figure V, the modified HDN model restored the data well and removed pixel noises as well as the undesired line artefacts (structured noise).

Appendix B (Continued)

Here, instead, we also computed results by "deactivating" only the bottom-most layer (HDN_{2-6}), the lowest two layers as described in the main text (HDN_{3-6}), and the bottom-most three layers (HDN_{4-6}). All results are shown in terms of peak signal-to-noise ratio (PSNR) in Table I. The best results are obtained with HDN_{3-6} setup.

B.4 Early Stopping of HDN Training

As reported in Section 4.5 and Appendix B.3, we show that unaltered GDD networks, if expressive enough to encode given structured noises, might not remove them. However, we observed that early stopping can also help in these situations to remove structured noises since GDD models encode for the structures of interest (we wish to preserve) first before starting to encode for structured noise (we wish to remove). In Figure VI we show results obtained with a vanilla VAE, trained on computed tomography data, at various intermediate states of training.

Note that similar observation has been made in the Deep Image Prior (DIP) work [57].

B.5 Visualizing GDD Operations

Here we will first introduce a method to visualize what operation a trained GDD network carries out on a given input image. We will then use this method to get a deeper look "behind the scenes" of two DN networks. Both networks are trained on images that are subjected to synthetic Moire patterns, one below the *encoding threshold* for this dataset ($\varepsilon = 0.01$), the other above ($\varepsilon = 1$).

The visualization we propose to use is based on an idea we found in [208]. More specifically, we take a linear approximation of our non-linear denoising network f_ψ by computing its Jacobian (J) evaluated for input x. We then perform a Singular Value Decomposition (SVD) on the computed Jacobian. In Figure VII we show examples of singular vectors of the Jacobians of the two trained networks.

When browsing through the singular vectors obtained with this method one notices that patterns that resemble the unwanted Moire patterns are either completely absent (for the network trained with $\varepsilon = 0.01$), or visible for all dominant (*i.e.*, all of at least the first 1000) singular vectors (for the network trained with $\varepsilon = 1.0$). These observations suggest that there is indeed a sharp transition between latent encodings learned below or above the *encoding threshold*.

B.6 Structured Noise Removal with GDD Methods vs. Supervised Baselines

We present a comparison of unsupervised GDD methods for structured noise/artefacts removal with the supervised methods Noise2Noise [54] and CARE [36] (see Table II).

Appendix B (Continued)

Due to having access to additional data, it is not surprising that both supervised methods outperform unsupervised GDD methods. However, depending on the structured noise, the Noise2Noise approach is not far ahead and has two decisive disadvantages. First, it requires more and specific paired training data, but maybe more importantly, it can and will only come up with one prediction while GDD approaches have the desired capability of sampling diverse data interpretations. The traditional supervised approach of CARE requires clean data during training, making a direct comparison even less fair. While, as we argue in the main text in great detail, GDD methods cannot apriori decide what spatial structures are signal and what others are artefacts, the paired training data for CARE makes this distinction very explicit.

B.7 Additional Qualitative Results for Structured Noise Removal

Here we show additional qualitative results obtained with DivNoising on images subjected to horizontal line artefacts or Moire patterns (see Figure VIII). More specifically we show, as we did also in Section 4.5, how the abundance of structured noises, modulated by the ε fractions used during training, changes the restoration properties of trained networks. Like in the main text, used test images are either exposed to the highest level of structured noise ($\varepsilon = 1$), or to inputs only being exposed to pixel-wise noise ($\varepsilon = 0$). Additionally, we show very similar results in the context of computed tomography (CT), where reconstruction artefacts are removed by a vanilla VAE (see Figure IX). Note that CT artefacts are modulated by the number of projection angles used during CT reconstruction, thus, instead of ε, we study CT streaking artefact removal as a function of projection angles used during CT reconstruction.

Appendix B (Continued)

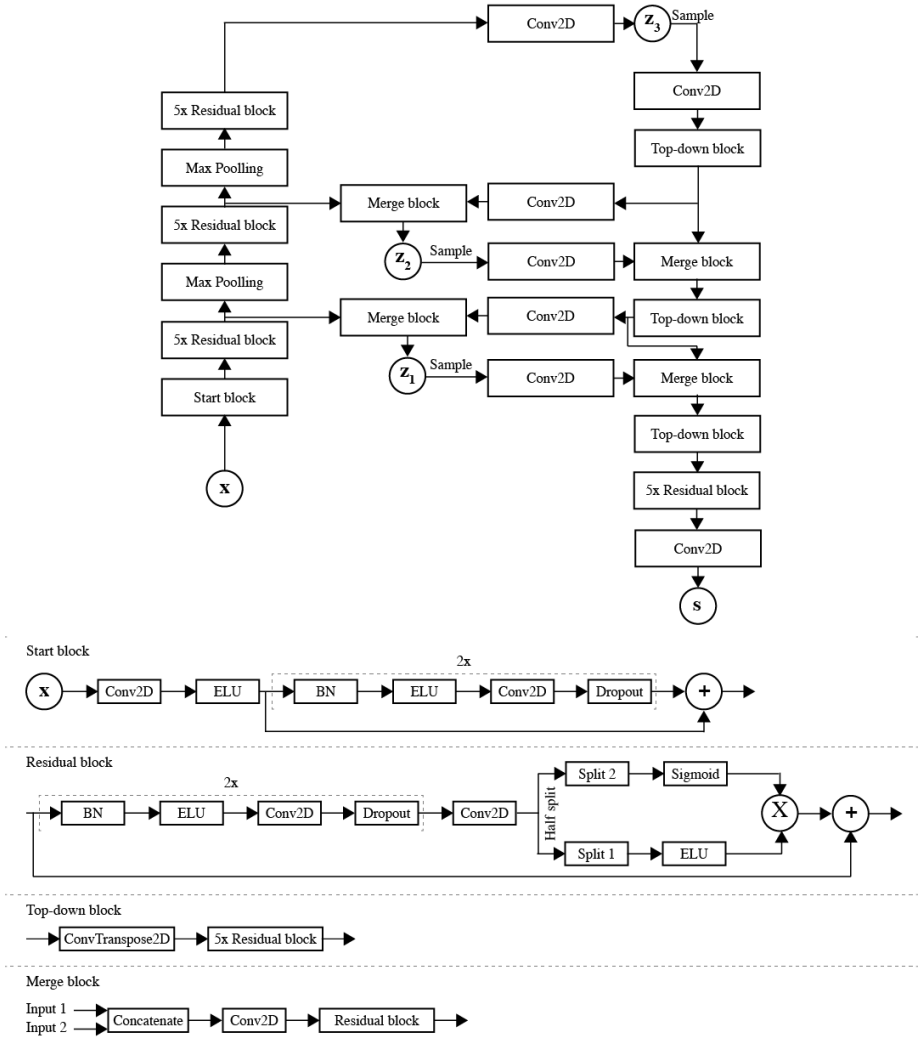

Figure IV: **Hierarchical DivNoising (HDN) network architecture.** The network architecture used for *MNIST* dataset is shown. For all other datasets, 6 stochastic latent groups are used instead of the 3 groups shown here.

Appendix B (Continued)

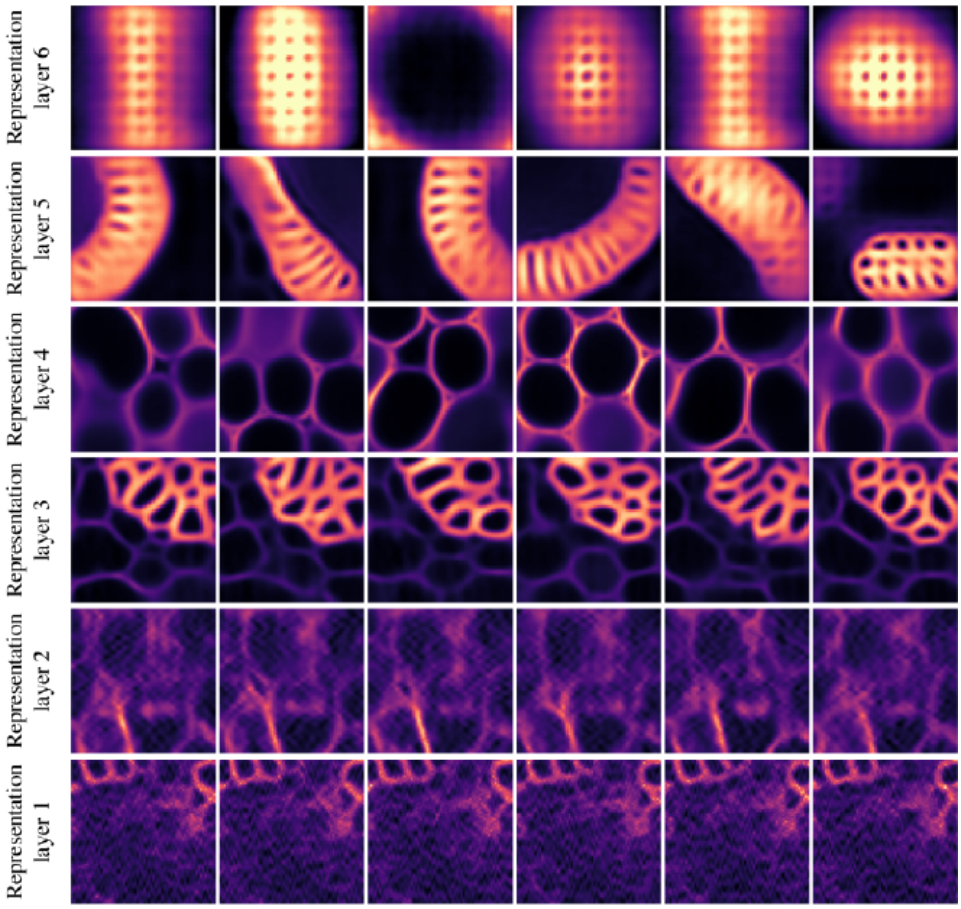

Figure V: **Visualizing what Hierarchical DivNoising encodes at each latent layer.** Please see Appendix B.3 for a detailed description of how the shown images have been created. The interesting observation is that structures that resemble the structured line artefacts are only visible in layers 1 and 2, which gave rise to the proposed HDN_{3-6} network used in Table 4.2 and Table I.

Appendix B (Continued)

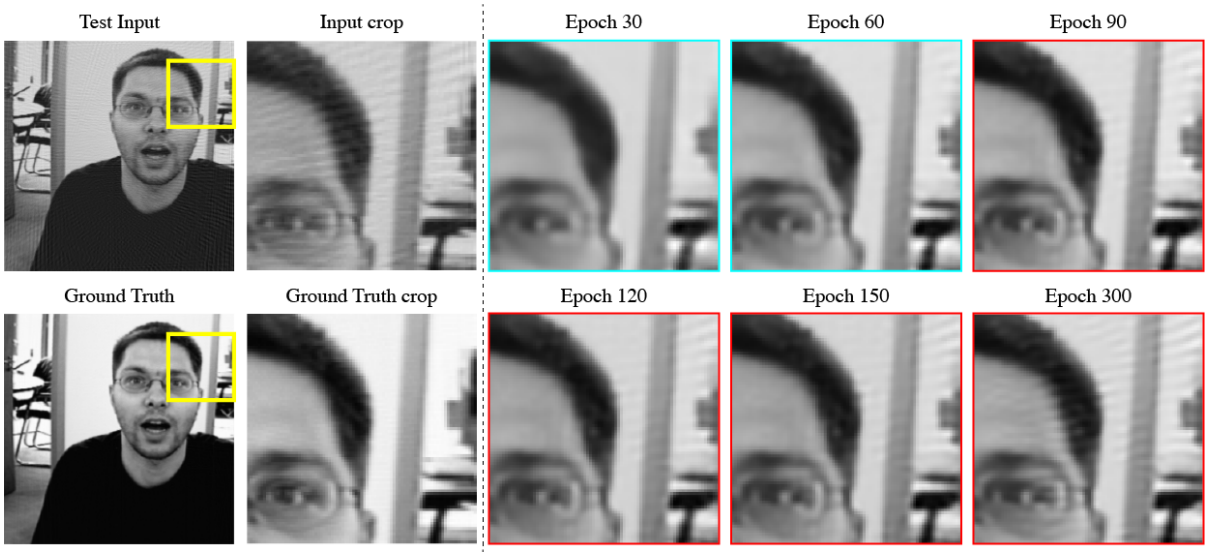

Figure VI: **Inductive Bias of GDD Network.** We show that vanilla VAE has a strong inductive bias towards learning the structures of interest that we wish to retain compared to the structured noise/artefacts that we wish to discard. The first column shows a test image corrupted with computed tomography (CT) artefacts image and below the corresponding Ground Truth image. The second column shows a cropped region while the results of a VAE network at different time intervals during training with inputs containing CT artefacts is shown in the next 3 columns. Cyan frames indicate training intervals where vanilla VAE does not encode for CT artefacts. Red frames indicate training intervals when vanilla VAE starts encoding for artefacts. The artefacts are only encoded after encoding for the signals of interest.

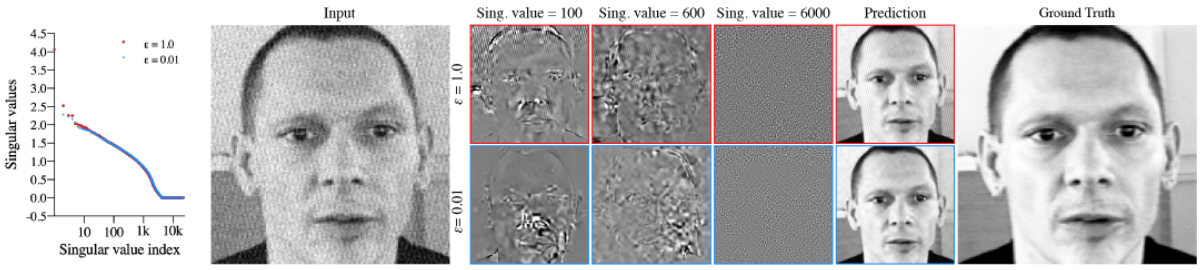

Figure VII: **Visualizing the restoration behavior of GDD for structured noise removal.** We consider two DivNoising networks trained with inputs corrupted with Gaussian noise superimposed with Moire patterns at abundance fractions of $\varepsilon = 0.01$ and $\varepsilon = 1.0$. We look at the singular vectors of Jacobian matrices of both networks at the shown test image. Even the dominant singular vectors (corresponding to index 100 and 600 for network trained with $\varepsilon = 1.0$ setup encode for Moire patterns while this is not the case for network trained with $\varepsilon = 0.01$ setup. The weaker singular vectors for both setups encode for pixel-wise noise.

Appendix B (Continued)

Method	Chessboard	Horizontal	Moire
DN $_{\varepsilon = 0.01}$	33.22	35.11	32.91
DN $_{\varepsilon = 0.03}$	33.04	34.98	33.08
DN $_{\varepsilon = 0.05}$	33.06	35.04	32.93
DN $_{\varepsilon = 0.10}$	33.05	34.80	32.85
DN $_{\varepsilon = 0.50}$	32.97	34.76	28.64
DN $_{\varepsilon = 1.0}$	19.59	34.10	26.63
N2N $_{\varepsilon = 0.01}$	34.80	35.35	32.15
N2N $_{\varepsilon = 0.03}$	34.59	36.06	32.15
N2N $_{\varepsilon = 0.05}$	33.84	35.95	32.82
N2N $_{\varepsilon = 0.10}$	31.29	36.27	32.89
N2N $_{\varepsilon = 0.50}$	24.92	36.97	30.63
N2N $_{\varepsilon = 1.0}$	19.76	36.84	26.94
CARE $_{\varepsilon = 0.01}$	33.69	35.94	32.22
CARE $_{\varepsilon = 0.03}$	35.12	35.49	32.21
CARE $_{\varepsilon = 0.05}$	35.09	35.97	31.63
CARE $_{\varepsilon = 0.10}$	35.02	36.13	32.92
CARE $_{\varepsilon = 0.50}$	34.94	34.78	34.89
CARE $_{\varepsilon = 1.0}$	**35.17**	**36.01**	**35.20**
test input:	200 pas	180 pas	150 pas
VAE $_{200}$	41.21	40.65	38.79
VAE $_{180}$	41.32	40.59	38.60
VAE $_{150}$	41.44	40.48	38.16
CARE $_{200}$	44.32	42.55	38.40
CARE $_{180}$	**45.69**	**44.75**	40.52
CARE $_{150}$	45.27	44.73	**43.12**

TABLE II: **Structured noise removal with GDDs.** Top 6 rows show DivNoising (DN) results on the *BioID-Faces* dataset, subjected to 3 sources of structured noises (see Section 4.5 for details). Rows differ by the abundance (ε-fraction) of structured noise used during training. All results are evaluated on test-inputs generated at maximum artefact abundance ($\varepsilon = 1$). The next two groups of 6 rows indicate the results of the supervised methods Noise2Noise [54] and CARE [36], respectively. The 6 bottom-most rows compare results of CARE, and vanilla VAEs trained on tomographic reconstructions from either 150, 180, or 200 projection angles (pas), also indicated as subscript in column 1. All numbers indicate PSNR values *w.r.t.* available ground truth images. As expected, supervised CARE method gives the best results in all categories.

Appendix B (Continued)

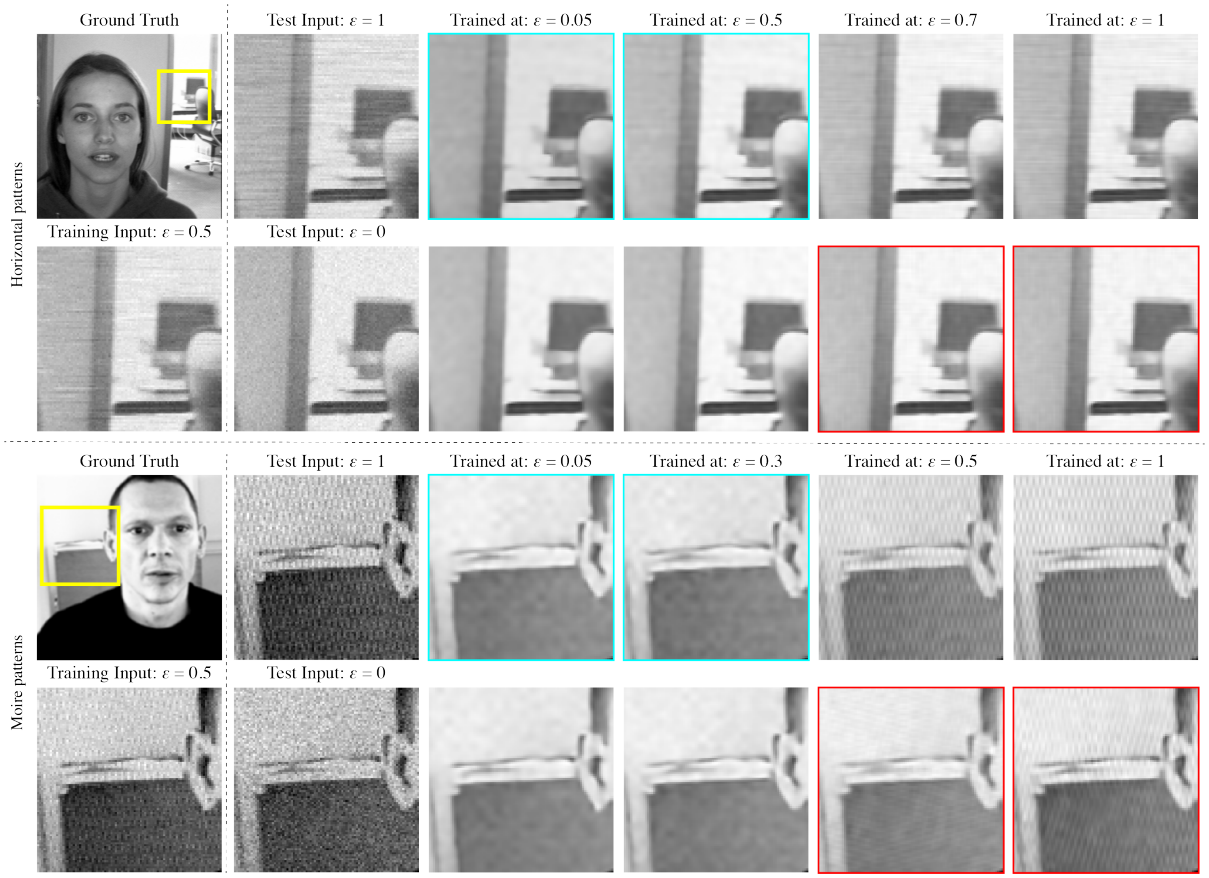

Figure VIII: **Testing structured noise removal with DivNoising at changing artefact abundance.** The top half shows results for horizontal pattern removal and the bottom half for the restoration of images corrupted with Moire patterns. The first column always shows a ground truth image and below a crop of a training image subject to Gaussian and structured noise ($\varepsilon = 0.5$). The second column shows two test inputs, top with highest abundance ($\varepsilon = 1$), bottom without structured noise ($\varepsilon = 0$). The remaining 4 columns show results obtained with DivNoising networks trained on input data exposed to various abundances ε. Cyan frames in the top row indicate cases where DivNoising successfully removes structured noise. Red frames, in the bottom row, indicate cases where restorations show structured noise patterns despite the input not containing them (hallucinated artefacts).

Appendix B (Continued)

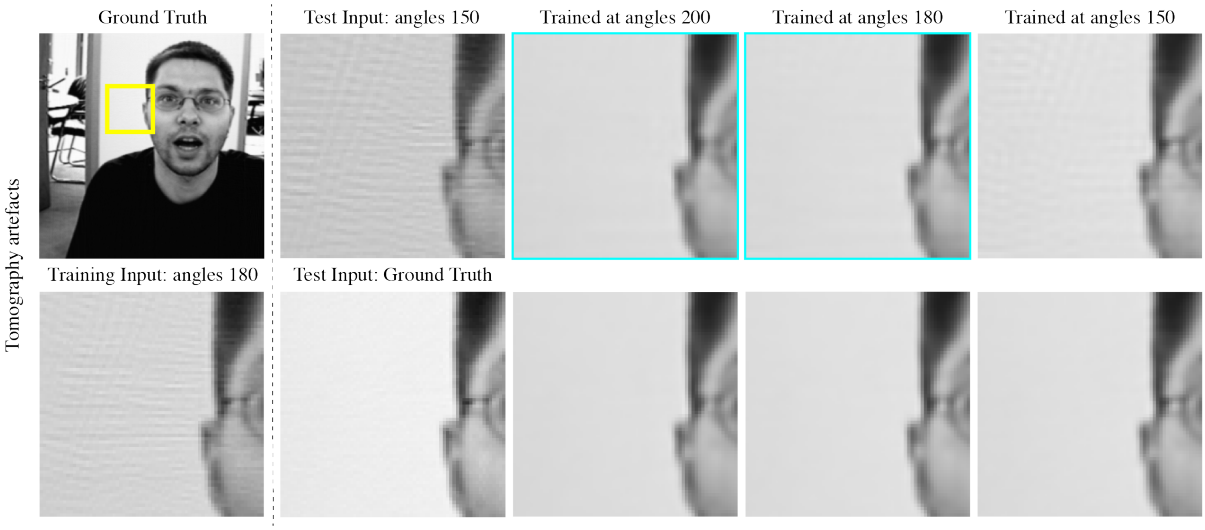

Figure IX: **Testing tomography artefact removal with vanilla VAEs with varying abundance of artefacts.** The first column shows a ground truth image and below a crop of a training image subject to computed tomography (CT) artefacts simulated by using only 180 projection angles for reconstruction. The second column shows two test inputs, the top one with the artefacts simulated by using 150 projection angles, while the one below does not contain this structured noise at all (ground truth). The remaining 4 columns show results obtained with vanilla VAE networks trained on input data exposed to tomography artefacts due to reconstruction with different number of projection angles. Cyan frames in the top row indicate cases where the vanilla VAE successfully removes the artefacts. The bottom row indicates denoising results for the same networks when the corresponding input does not contain artefacts.

Appendix C

Supplementary Material for Chapter 6

C.1 DSB Results with Increasingly Many GT labels

The DSB dataset we have used offers the possibility to run DenoiSeg with arbitrary many segmentation GT labels. While DenoiSeg is intended in cases where the amount of such labels is very limited, in Figure X we plot the segmentation results of DenoiSeg, the sequential baseline, as well as the baseline as defined in the main text.

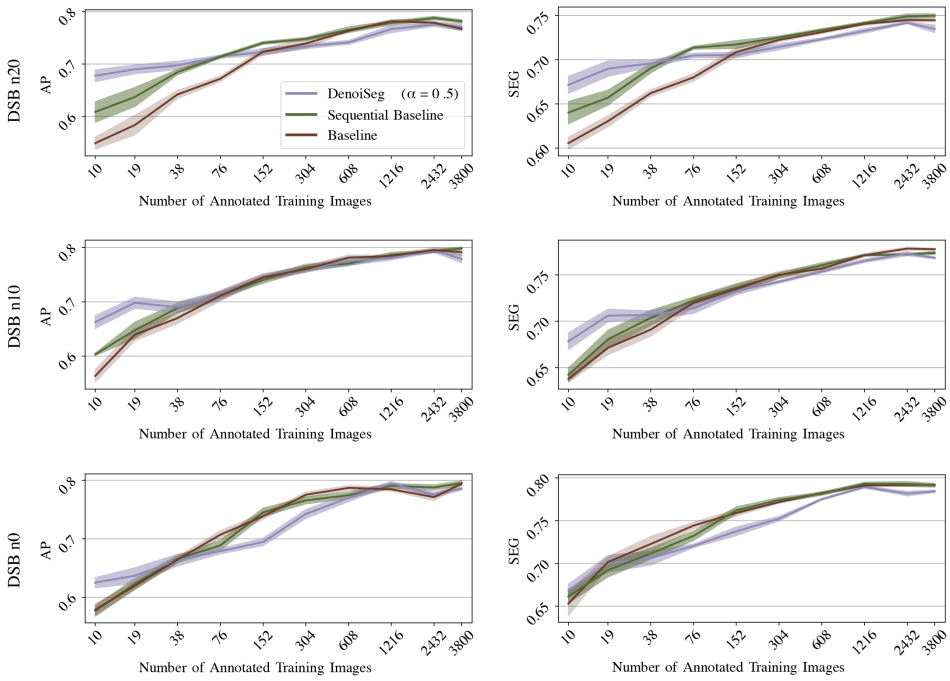

Figure X: **Extended version of Figure 6.4.** Results for DSB n0, n10 and n20, evaluated with Average Precision (AP) [119] and SEG-Score [134]. DenoiSeg outperforms both baseline methods, mainly when only limited segmentation ground truth is available. Note that the advantage of our proposed method for this dataset is at least partially compromised when the image data is not noisy (row 3).

Appendix C (Continued)

As expected, with additional labels, the advantage of also seeing noisy images decreases, leading to similarly good results for all compared methods. It is still reassuring to see that the performance of DenoiSeg is still essentially on par with the results of a vanilla U-Net that does not perform the joint training we propose.

C.2 Our Baseline *vs.* Vanilla 3-class U-Net

The baseline method we used in this work is, as explained in the main text, a DenoiSeg network with α being set to 0. This is, in fact, very similar to using a vanilla 3-class U-Net. While we are still feeding noisy images, we are not backpropagating any denoising loss, meaning that only the data for which segmentation labels exist will contribute to the training. The one difference is, that some of the hyperparameters (number of epochs, adaptation of learning rate, *etc.*) will slightly diverge in these two baseline setups. Figure XI shows that these subtle differences are in fact not making any practical differences.

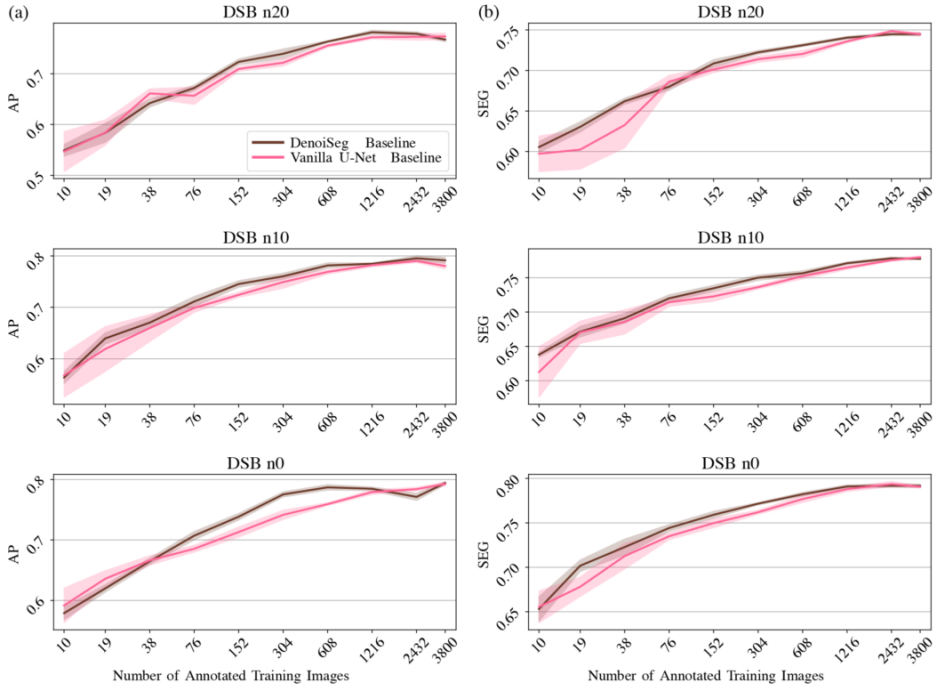

Figure XI: **Comparison of vanilla U-Net with our DenoiSeg $\alpha = 0$ baseline for DSB datasets.** Our DenoiSeg $\alpha = 0$ baseline is at least as good or better than the vanilla U-net baseline both in terms of Average Precision (AP) [119] and SEG-Score [134] metrics. Hence, we establish a stronger baseline with DenoiSeg $\alpha = 0$ and measure our performance against this baseline (see Figure X,Figure 6.3 and Figure 6.5.)

CITED LITERATURE

1. Grady, C. L., McIntosh, A. R., Rajah, M. N., and Craik, F. I.: Neural correlates of the episodic encoding of pictures and words. Proceedings of the National Academy of Sciences, 95(5):2703–2708, 1998.

2. Hutmacher, F.: Why is there so much more research on vision than on any other sensory modality? Frontiers in psychology, 10:2246, 2019.

3. DiCarlo, J. J., Zoccolan, D., and Rust, N. C.: How does the brain solve visual object recognition? Neuron, 73(3):415–434, 2012.

4. Van Valen, D. A., Kudo, T., Lane, K. M., Macklin, D. N., Quach, N. T., DeFelice, M. M., Maayan, I., Tanouchi, Y., Ashley, E. A., and Covert, M. W.: Deep learning automates the quantitative analysis of individual cells in live-cell imaging experiments. PLoS computational biology, 12(11):e1005177, 2016.

5. Hayakawa, T., Prasath, V. S., Kawanaka, H., Aronow, B. J., and Tsuruoka, S.: Computational nuclei segmentation methods in digital pathology: a survey. Archives of Computational Methods in Engineering, pages 1–13, 2019.

6. Li, F., Zhou, X., Ma, J., and Wong, S. T.: Multiple nuclei tracking using integer programming for quantitative cancer cell cycle analysis. IEEE transactions on medical imaging, 29(1):96–105, 2009.

7. Witte, M., Jaspers, S., Wenck, H., Rübhausen, M., and Fischer, F.: Noise reduction and quantification of fiber orientations in greyscale images. PloS one, 15(1):e0227534, 2020.

8. Kindle, L., Kakadiaris, I., Ju, T., and Carson, J.: A semiautomated approach for artefact removal in serial tissue cryosections. Journal of microscopy, 241(2):200–206, 2011.

9. Goubran, M., Leuze, C., Hsueh, B., Aswendt, M., Ye, L., Tian, Q., Cheng, M. Y., Crow, A., Steinberg, G. K., McNab, J. A., et al.: Multimodal image registration and connectivity analysis for integration of connectomic data from microscopy to mri. Nature communications, 10(1):1–17, 2019.

10. Xue, Y. and Ray, N.: Cell detection in microscopy images with deep convolutional neural network and compressed sensing. arXiv preprint arXiv:1708.03307, 2017.

11. Brooks, T. and Barron, J. T.: Learning to synthesize motion blur. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6840–6848, 2019.

12. Jiang, C., Liao, J., Dong, P., Ma, Z., Cai, D., Zheng, G., Liu, Y., Bu, H., and Yao, J.: Blind deblurring for microscopic pathology images using deep learning networks. arXiv preprint arXiv:2011.11879, 2020.

13. Farahani, N., Parwani, A. V., and Pantanowitz, L.: Whole slide imaging in pathology: advantages, limitations, and emerging perspectives. Pathology and Laboratory Medicine International, 7:23–33, 2015.

14. Haider, S., Cameron, A., Siva, P., Lui, D., Shafiee, M., Boroomand, A., Haider, N., and Wong, A.: Fluorescence microscopy image noise reduction using a stochastically-connected random field model. Scientific reports, 6(1):1–16, 2016.

15. Bernacki, J.: Digital camera identification based on analysis of optical defects. Multimedia Tools and Applications, 79(3):2945–2963, 2020.

16. Rabbani, M.: Jpeg2000: Image compression fundamentals, standards and practice. Journal of Electronic Imaging, 11(2):286, 2002.

17. Broaddus, C., Krull, A., Weigert, M., Schmidt, U., and Myers, G.: Removing structured noise with self-supervised blind-spot networks. In 2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI), pages 159–163, 2020.

18. Boas, F. E., Fleischmann, D., et al.: Ct artifacts: causes and reduction techniques. Imaging Med, 4(2):229–240, 2012.

19. Luu, K., Le, T. H. N., Seshadri, K., and Savvides, M.: Facecut-a robust approach for facial feature segmentation. In 2012 19th IEEE International Conference on Image Processing, pages 1841–1844. IEEE, 2012.

20. Brostow, G. J., Fauqueur, J., and Cipolla, R.: Semantic object classes in video: A high-definition ground truth database. Pattern Recognition Letters, 30(2):88–97, 2009.

21. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3213–3223, 2016.

22. Arroyo, R., Yebes, J. J., Bergasa, L. M., Daza, I. G., and Almazán, J.: Expert video-surveillance system for real-time detection of suspicious behaviors in shopping malls. Expert systems with Applications, 42(21):7991–8005, 2015.

23. Ma, C., Huang, J.-B., Yang, X., and Yang, M.-H.: Hierarchical convolutional features for visual tracking. In Proceedings of the IEEE international conference on computer vision, pages 3074–3082, 2015.

24. Kiy, K.: Segmentation and detection of contrast objects and their application in robot navigation. Pattern Recognition and Image Analysis, 25(2):338–346, 2015.

25. DeSouza, G. N. and Kak, A. C.: Vision for mobile robot navigation: A survey. IEEE transactions on pattern analysis and machine intelligence, 24(2):237–267, 2002.

26. Pham, D. L., Xu, C., and Prince, J. L.: Current methods in medical image segmentation. Annual review of biomedical engineering, 2(1):315–337, 2000.

27. Schmidt, U., Weigert, M., Broaddus, C., and Myers, G.: Cell detection with star-convex polygons. In International Conference on Medical Image Computing and Computer-Assisted Intervention, pages 265–273. Springer, 2018.

28. Jalal, A. and Kamal, S.: Real-time life logging via a depth silhouette-based human activity recognition system for smart home services. In 2014 11th IEEE International conference on advanced video and signal based surveillance (AVSS), pages 74–80. IEEE, 2014.

29. Mihailidis, A., Carmichael, B., and Boger, J.: The use of computer vision in an intelligent environment to support aging-in-place, safety, and independence in the home. IEEE Transactions on information technology in biomedicine, 8(3):238–247, 2004.

30. Kremer, J., Stensbo-Smidt, K., Gieseke, F., Pedersen, K. S., and Igel, C.: Big universe, big data: machine learning and image analysis for astronomy. IEEE Intelligent Systems, 32(2):16–22, 2017.

31. Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., et al.: Fiji: an open-source platform for biological-image analysis. Nature methods, 9(7):676–682, 2012.

32. Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., Van Der Laak, J. A., Van Ginneken, B., and Sánchez, C. I.: A survey on deep learning in medical image analysis. Medical image analysis, 42:60–88, 2017.

33. Wojnar, L.: Image analysis: applications in materials engineering. Crc Press, 2019.

34. Shah, V.: Image processing and its military applications. Defence science journal. Delhi, 37(4):457–468, 1987.

35. Richards, J. A. and Richards, J.: Remote sensing digital image analysis, volume 3. Springer, 1999.

36. Weigert, M., Schmidt, U., Boothe, T., Müller, A., Dibrov, A., Jain, A., Wilhelm, B., Schmidt, D., Broaddus, C., Culley, S., et al.: Content-aware image restoration: pushing the limits of fluorescence microscopy. Nature methods, 15(12):1090, 2018.

37. Zhang, Y., Zhu, Y., Nichols, E., Wang, Q., Zhang, S., Smith, C., and Howard, S.: A poisson-gaussian denoising dataset with real fluorescence microscopy images. In CVPR, 2019.

38. Buchholz, T.-O., Jordan, M., Pigino, G., and Jug, F.: Cryo-care: Content-aware image restoration for cryo-transmission electron microscopy data. In 2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019), pages 502–506. IEEE, 2019.

39. Buchholz, T.-O., Krull, A., Shahidi, R., Pigino, G., Jékely, G., and Jug, F.: Content-aware image restoration for electron microscopy. In Three-Dimensional Electron Microscopy, eds. G. Pigino and T. Müller-Reichert, pages 277–289. Academic Press, July 2019.

40. Krull, A., Buchholz, T.-O., and Jug, F.: Noise2void-learning denoising from single noisy images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2129–2137, 2019.

41. Krull, A., Vicar, T., and Jug, F.: Probabilistic noise2void: Unsupervised content-aware denoising. arXiv preprint arXiv:1906.00651, 2019.

42. Laine, S., Karras, T., Lehtinen, J., and Aila, T.: High-quality self-supervised deep image denoising. In Advances in Neural Information Processing Systems, pages 6968–6978, 2019.

43. Tomasi, C. and Manduchi, R.: Bilateral filtering for gray and color images. In Sixth international conference on computer vision (IEEE Cat. No. 98CH36271), pages 839–846. IEEE, 1998.

44. Buades, A., Coll, B., and Morel, J.-M.: A non-local algorithm for image denoising. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), volume 2, pages 60–65. IEEE, 2005.

45. Dabov, K., Foi, A., Katkovnik, V., and Egiazarian, K.: Image denoising by sparse 3-d transform-domain collaborative filtering. IEEE Transactions on image processing, 16(8):2080–2095, 2007.

46. Milanfar, P.: A tour of modern image filtering: New insights and methods, both practical and theoretical. IEEE signal processing magazine, 30(1):106–128, 2012.

47. Burger, H. C., Schuler, C. J., and Harmeling, S.: Image denoising: Can plain neural networks compete with bm3d? In 2012 IEEE conference on computer vision and pattern recognition, pages 2392–2399. IEEE, 2012.

48. Jain, V. and Seung, S.: Natural image denoising with convolutional networks. Advances in neural information processing systems, 21:769–776, 2008.

49. Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L.: Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. IEEE Transactions on Image Processing, 26(7):3142–3155, 2017.

50. Zhang, K., Zuo, W., Chen, Y., Meng, D., and Zhang, L.: Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. IEEE Transactions on Image Processing, 26(7):3142–3155, 2017.

51. Xie, J., Xu, L., and Chen, E.: Image denoising and inpainting with deep neural networks. Advances in neural information processing systems, 25:341–349, 2012.

52. Weigert, M., Royer, L., Jug, F., and Myers, G.: Isotropic reconstruction of 3D fluorescence microscopy images using convolutional neural networks. arXiv, April 2017.

53. Krull, A., Vicar, T., Prakash, M., Lalit, M., and Jug, F.: Probabilistic Noise2Void: Unsupervised Content-Aware Denoising. Front. Comput. Sci., 2:60, February 2020.

54. Lehtinen, J., Munkberg, J., Hasselgren, J., Laine, S., Karras, T., Aittala, M., and Aila, T.: Noise2Noise: Learning image restoration without clean data. In International Conference on Machine Learning, 2018.

55. Batson, J. and Royer, L.: Noise2self: Blind denoising by self-supervision. In International Conference on Machine Learning, 2019.

56. Long, J., Shelhamer, E., and Darrell, T.: Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3431–3440, 2015.

57. Ulyanov, D., Vedaldi, A., and Lempitsky, V.: Deep image prior. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 9446–9454, 2018.

58. Prakash, M., Lalit, M., Tomancak, P., Krul, A., and Jug, F.: Fully unsupervised probabilistic noise2void. In 2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI), pages 154–158. IEEE, 2020.

59. Quan, Y., Chen, M., Pang, T., and Ji, H.: Self2self with dropout: Learning self-supervised denoising from single image. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 1890–1898, 2020.

60. Xie, Y., Wang, Z., and Ji, S.: Noise2same: Optimizing a self-supervised bound for image denoising. arXiv preprint arXiv:2010.11971, 2020.

61. Khademi, W., Rao, S., Minnerath, C., Hagen, G., and Ventura, J.: Self-supervised poisson-gaussian denoising. arXiv preprint arXiv:2002.09558, 2020.

62. Goncharova, A. S., Honigmann, A., Jug, F., and Krull, A.: Improving blind spot denoising for microscopy. In European Conference on Computer Vision, pages 380–393. Springer, 2020.

63. Ronneberger, O., P.Fischer, and Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In Medical Image Computing and Computer-Assisted Intervention (MICCAI), volume 9351 of LNCS, pages 234–241. Springer, 2015. (available on arXiv:1505.04597 [cs.CV]).

64. Kingma, D. P. and Ba, J.: Adam: A method for stochastic optimization. 3rd international conference on learning representations, iclr 2015. arXiv preprint arXiv:1412.6980, 9, 2015.

65. Kingma, D. P. and Welling, M.: Auto-encoding variational bayes. In 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, eds. Y. Bengio and Y. LeCun, 2014.

66. Rezende, D. J., Mohamed, S., and Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. arXiv preprint arXiv:1401.4082, 2014.

67. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1):1929–1958, 2014.

68. Gal, Y. and Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In international conference on machine learning, pages 1050–1059, 2016.

69. Lakshminarayanan, B., Pritzel, A., and Blundell, C.: Simple and scalable predictive uncertainty estimation using deep ensembles. In Advances in neural information processing systems, pages 6402–6413, 2017.

70. Böhm, V., Lanusse, F., and Seljak, U.: Uncertainty quantification with generative models. arXiv preprint arXiv:1910.10046, 2019.

71. Sensoy, M., Kaplan, L., Cerutti, F., and Saleki, M.: Uncertainty-aware deep classifiers using generative models. arXiv preprint arXiv:2006.04183, 2020.

72. Im, D. I. J., Ahn, S., Memisevic, R., and Bengio, Y.: Denoising criterion for variational auto-encoding framework. In Thirty-First AAAI Conference on Artificial Intelligence, 2017.

73. Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A.: Extracting and composing robust features with denoising autoencoders. In Proceedings of the 25th international conference on Machine learning, pages 1096–1103, 2008.

74. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.-A., and Bottou, L.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. Journal of machine learning research, 11(12), 2010.

75. Jiao, J., Bao, L., Wei, Y., He, S., Shi, H., Lau, R., and Huang, T. S.: Laplacian denoising autoencoder. arXiv preprint arXiv:2003.13623, 2020.

76. Kohl, S., Romera-Paredes, B., Meyer, C., De Fauw, J., Ledsam, J. R., Maier-Hein, K., Eslami, S. A., Rezende, D. J., and Ronneberger, O.: A probabilistic u-net for segmentation of ambiguous images. In Advances in Neural Information Processing Systems, pages 6965–6975, 2018.

77. Kohl, S. A., Romera-Paredes, B., Maier-Hein, K. H., Rezende, D. J., Eslami, S., Kohli, P., Zisserman, A., and Ronneberger, O.: A hierarchical probabilistic u-net for modeling multi-scale ambiguities. arXiv preprint arXiv:1905.13077, 2019.

78. Baumgartner, C. F., Tezcan, K. C., Chaitanya, K., Hötker, A. M., Muehlematter, U. J., Schawkat, K., Becker, A. S., Donati, O., and Konukoglu, E.: Phiseg: Capturing uncertainty in medical image segmentation. In International Conference on Medical Image Computing and Computer-Assisted Intervention, pages 119–127. Springer, 2019.

79. Nazabal, A., Olmos, P. M., Ghahramani, Z., and Valera, I.: Handling incomplete heterogeneous data using vaes. Pattern Recognition, page 107501, 2020.

80. Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R. H., and Levine, S.: Stochastic variational video prediction. arXiv preprint arXiv:1710.11252, 2017.

81. Balakrishnan, G., Dalca, A. V., Zhao, A., Guttag, J. V., Durand, F., and Freeman, W. T.: Visual deprojection: Probabilistic recovery of collapsed dimensions. In Proceedings of the IEEE International Conference on Computer Vision, pages 171–180, 2019.

82. Kingma, D. P. and Welling, M.: An introduction to variational autoencoders. Foundations and Trends® in Machine Learning, 12(4):307–392, 2019.

83. Huang, H., He, R., Sun, Z., Tan, T., et al.: Introvae: Introspective variational autoencoders for photographic image synthesis. In Advances in neural information processing systems, pages 52–63, 2018.

84. Fan, Y., Wen, G., Li, D., Qiu, S., Levine, M. D., and Xiao, F.: Video anomaly detection and localization via gaussian mixture fully convolutional variational autoencoder. Computer Vision and Image Understanding, page 102920, 2020.

85. Cheng, Y.: Mean shift, mode seeking, and clustering. IEEE transactions on pattern analysis and machine intelligence, 17(8):790–799, 1995.

86. Faraji, H. and MacLean, W. J.: Ccd noise removal in digital images. IEEE Transactions on image processing, 15(9):2676–2685, 2006.

87. Jezierska, A., Chaux, C., Pesquet, J.-C., and Talbot, H.: An em approach for poisson-gaussian noise modeling. In 2011 19th European Signal Processing Conference, pages 2244–2248. IEEE, 2011.

88. Grimson, W. E. L. and Grimson, W.: From images to surfaces: A computational study of the human early visual system, volume 4. MIT press Cambridge, MA, 1981.

89. Li, S. Z.: Markov random field models in computer vision. In European conference on computer vision, pages 361–370. Springer, 1994.

90. Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological), 58(1):267–288, 1996.

91. Van Rijsbergen, C. J.: Information retrieval. Citeseer, 1979.

92. Jaccard, P.: Étude comparative de la distribution florale dans une portion des alpes et des jura. Bull Soc Vaudoise Sci Nat, 37:547–579, 1901.

93. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L.: Microsoft coco: Common objects in context. In European conference on computer vision, pages 740–755. Springer, 2014.

94. Emre Akbas, C., Ulman, V., Maska, M., Jug, F., and Kozubek, M.: Automatic fusion of segmentation and tracking labels. In Proceedings of the European Conference on Computer Vision (ECCV), pages 0–0, 2018.

95. Yokota, T., Hontani, H., Zhao, Q., and Cichocki, A.: Manifold modeling in embedded space: A perspective for interpreting" deep image prior". arXiv preprint arXiv:1908.02995, 2019.

96. Tachella, J., Tang, J., and Davies, M.: Cnn denoisers as non-local filters: The neural tangent denoiser. arXiv preprint arXiv:2006.02379, 2020.

97. Prakash, M., Krull, A., and Jug, F.: Fully unsupervised diversity denoising with convolutional variational autoencoders. In International Conference on Learning Representations, 2021.

98. Delbracio, M., Talebi, H., and Milanfar, P.: Projected distribution loss for image enhancement. arXiv preprint arXiv:2012.09289, 2020.

99. Rudin, L. I., Osher, S., and Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D: nonlinear phenomena, 60(1-4):259–268, 1992.

100. Golub, G. H., Hansen, P. C., and O'Leary, D. P.: Tikhonov regularization and total least squares. SIAM journal on matrix analysis and applications, 21(1):185–194, 1999.

101. Bruckstein, A. M., Donoho, D. L., and Elad, M.: From sparse solutions of systems of equations to sparse modeling of signals and images. SIAM review, 51(1):34–81, 2009.

102. Zoran, D. and Weiss, Y.: From learning models of natural image patches to whole image restoration. In 2011 International Conference on Computer Vision, pages 479–486. IEEE, 2011.

103. Romano, Y., Elad, M., and Milanfar, P.: The little engine that could: Regularization by denoising (red). SIAM Journal on Imaging Sciences, 10(4):1804–1844, 2017.

104. Sønderby, C. K., Raiko, T., Maaløe, L., Sønderby, S. K., and Winther, O.: Ladder variational autoencoders. arXiv preprint arXiv:1602.02282, 2016.

105. Maaløe, L., Fraccaro, M., Liévin, V., and Winther, O.: Biva: A very deep hierarchy of latent variables for generative modeling. arXiv preprint arXiv:1902.02102, 2019.

106. Vahdat, A. and Kautz, J.: Nvae: A deep hierarchical variational autoencoder. arXiv preprint arXiv:2007.03898, 2020.

107. Child, R.: Very deep {vae}s generalize autoregressive models and can outperform them on images. In International Conference on Learning Representations, 2021.

108. Liévin, V., Dittadi, A., Maaløe, L., and Winther, O.: Towards hierarchical discrete variational autoencoders. In 2nd Symposium on Advances in Approximate Bayesian Inference, 2019.

109. Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S.: Generating sentences from a continuous space. arXiv preprint arXiv:1511.06349, 2015.

110. Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M.: Improving variational inference with inverse autoregressive flow. arXiv preprint arXiv:1606.04934, 2016.

111. Chen, X., Kingma, D. P., Salimans, T., Duan, Y., Dhariwal, P., Schulman, J., Sutskever, I., and Abbeel, P.: Variational lossy autoencoder. arXiv preprint arXiv:1611.02731, 2016.

112. Kak, A. C., Slaney, M., and Wang, G.: Principles of computerized tomographic imaging, 2002.

113. Ramesh, G., Srinivasa, N., and Rajgopal, K.: An algorithm for computing the discrete radon transform with some applications. In Fourth IEEE Region 10 International Conference TENCON, pages 78–81. IEEE, 1989.

114. Bruyant, P. P., Sau, J., and Mallet, J.-J.: Streak artifact reduction in filtered backprojection using a level line–based interpolation method. Journal of Nuclear Medicine, 41(11):1913–1919, 2000.

115. Jug, F., Pietzsch, T., Preibisch, S., and Tomancak, P.: Bioimage informatics in the context of drosophila research. Methods, 68(1):60–73, 2014.

116. Caicedo, J. C., Roth, J., Goodman, A., Becker, T., Karhohs, K. W., Broisin, M., Molnar, C., McQuin, C., Singh, S., Theis, F. J., et al.: Evaluation of deep learning strategies for nucleus segmentation in fluorescence images. Cytometry Part A, 95(9):952–965, 2019.

117. Moen, E., Bannon, D., Kudo, T., Graf, W., Covert, M., and Van Valen, D.: Deep learning for cellular image analysis. Nature methods, pages 1–14, 2019.

118. Razzak, M. I., Naz, S., and Zaib, A.: Deep learning for medical image processing: Overview, challenges and the future. In Classification in BioApps, pages 323–350. Springer, 2018.

119. Schmidt, U., Weigert, M., Broaddus, C., and Myers, G.: Cell detection with star-convex polygons. In Medical Image Computing and Computer Assisted Intervention -

MICCAI 2018 - 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part II, pages 265–273, 2018.

120. Stringer, C., Michaelos, M., and Pachitariu, M.: Cellpose: a generalist algorithm for cellular segmentation. bioRxiv, 2020.

121. Hirsch, P., Mais, L., and Kainmueller, D.: Patchperpix for instance segmentation. arXiv preprint arXiv:2001.07626, 2020.

122. Lalit, M., Tomancak, P., and Jug, F.: Embedding-based instance segmentation of microscopy images. arXiv preprint arXiv:2101.10033, 2021.

123. Shorten, C. and Khoshgoftaar, T. M.: A survey on image data augmentation for deep learning. Journal of Big Data, 6(1):60, 2019.

124. Zhao, A., Balakrishnan, G., Durand, F., Guttag, J. V., and Dalca, A. V.: Data augmentation using learned transformations for one-shot medical image segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8543–8553, 2019.

125. Ihle, S. J., Reichmuth, A. M., Girardin, S., Han, H., Stauffer, F., Bonnin, A., Stampanoni, M., Pattisapu, K., Vörös, J., and Forró, C.: Unsupervised data to content transformation with histogram-matching cycle-consistent generative adversarial networks. Nature Machine Intelligence, pages 1–10, 2019.

126. Osokin, A., Chessel, A., Carazo Salas, R. E., and Vaggi, F.: Gans for biological image synthesis. In Proceedings of the IEEE International Conference on Computer Vision, pages 2233–2242, 2017.

127. Sandfort, V., Yan, K., Pickhardt, P. J., and Summers, R. M.: Data augmentation using generative adversarial networks (cyclegan) to improve generalizability in ct segmentation tasks. Scientific reports, 9(1):1–9, 2019.

128. Doersch, C., Gupta, A., and Efros, A. A.: Unsupervised visual representation learning by context prediction. In CVPR, 2015.

129. Zamir, A. R., Sax, A., Shen, W., Guibas, L. J., Malik, J., and Savarese, S.: Taskonomy: Disentangling task transfer learning. In CVPR, 2018.

130. Falk, T., Mai, D., Bensch, R., Çiçek, Ö., Abdulkadir, A., Marrakchi, Y., Böhm, A., Deubner, J., Jäckel, Z., Seiwald, K., et al.: U-net: deep learning for cell counting, detection, and morphometry. Nature methods, 16(1):67, 2019.

131. Chen, H., Qi, X., Yu, L., and Heng, P.-A.: Dcan: deep contour-aware networks for accurate gland segmentation. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pages 2487–2496, 2016.

132. Guerrero-Pena, F. A., Fernandez, P. D. M., Ren, T. I., Yui, M., Rothenberg, E., and Cunha, A.: Multiclass weighted loss for instance segmentation of cluttered cells. In 2018 25th IEEE International Conference on Image Processing (ICIP), pages 2451–2455. IEEE, 2018.

133. Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A.: The pascal visual object classes (voc) challenge. International journal of computer vision, 88(2):303–338, 2010.

134. Ulman, V., Maška, M., Magnusson, K. E., Ronneberger, O., Haubold, C., Harder, N., Matula, P., Matula, P., Svoboda, D., Radojevic, M., et al.: An objective comparison of cell-tracking algorithms. Nature methods, 14(12):1141, 2017.

135. Liu, D., Wen, B., Liu, X., Wang, Z., and Huang, T. S.: When image denoising meets high-level vision tasks: A deep learning approach. arXiv preprint arXiv:1706.04284, 2017.

136. Wang, S., Wen, B., Wu, J., Tao, D., and Wang, Z.: Segmentation-aware image denoising without knowing true segmentation. arXiv preprint arXiv:1905.08965, 2019.

137. Prakash, M., Buchholz, T.-O., Lalit, M., Tomancak, P., Jug, F., and Krull, A.: Leveraging self-supervised denoising for image segmentation, 2019.

138. Zheng, H., Yang, L., Chen, J., Han, J., Zhang, Y., Liang, P., Zhao, Z., Wang, C., and Chen, D. Z.: Biomedical image segmentation via representative annotation. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 5901–5908, 2019.

139. Dutta, S., Djabrayan, N. J.-V., Torquato, S., Shvartsman, S. Y., and Krajnc, M.: Self-similar dynamics of nuclear packing in the early drosophila embryo. Biophysical journal, 117(4):743–750, 2019.

140. Hatipoglu, N. and Bilgin, G.: Cell segmentation in histopathological images with deep learning algorithms by utilizing spatial relationships. Medical & biological engineering & computing, 55(10):1829–1848, 2017.

141. Weigert, M., Schmidt, U., Haase, R., Sugawara, K., and Myers, G.: Star-convex polyhedra for 3d object detection and segmentation in microscopy. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pages 3666–3673, 2020.

142. Hirsch, P. and Kainmueller, D.: An auxiliary task for learning nuclei segmentation in 3d microscopy images. In Medical Imaging with Deep Learning, pages 304–321. PMLR, 2020.

143. Stringer, C., Wang, T., Michaelos, M., and Pachitariu, M.: Cellpose: a generalist algorithm for cellular segmentation. Nature Methods, 18(1):100–106, 2021.

144. Prakash, M., Buchholz, T.-O., Lalit, M., Tomancak, P., Jug, F., and Krull, A.: Leveraging self-supervised denoising for image segmentation. In 2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI), pages 428–432. IEEE, 2020.

145. Buchholz, T.-O., Prakash, M., Schmidt, D., Krull, A., and Jug, F.: Denoiseg: joint denoising and segmentation. In European Conference on Computer Vision, pages 324–337. Springer, 2020.

146. Rumberger, J. L., Mais, L., and Kainmueller, D.: Probabilistic deep learning for instance segmentation. In European Conference on Computer Vision, pages 445–457. Springer, 2020.

147. Akram, S. U., Kannala, J., Eklund, L., and Heikkilä, J.: Cell segmentation proposal network for microscopy image analysis. In Deep Learning and Data Labeling for Medical Applications, pages 21–29. Springer, 2016.

148. Sadanandan, S. K., Ranefall, P., Le Guyader, S., and Wählby, C.: Automated training of deep convolutional neural networks for cell segmentation. Scientific reports, 7(1):1–7, 2017.

149. Al-Kofahi, Y., Zaltsman, A., Graves, R., Marshall, W., and Rusu, M.: A deep learning-based algorithm for 2-d cell segmentation in microscopy images. BMC bioinformatics, 19(1):1–11, 2018.

150. Moshkov, N., Mathe, B., Kertesz-Farkas, A., Hollandi, R., and Horvath, P.: Test-time augmentation for deep learning-based cell segmentation on microscopy images. Scientific reports, 10(1):1–7, 2020.

151. Yang, L., Zhang, Y., Chen, J., Zhang, S., and Chen, D. Z.: Suggestive annotation: A deep active learning framework for biomedical image segmentation. In International conference on medical image computing and computer-assisted intervention, pages 399–407. Springer, 2017.

152. Xu, X., Lu, Q., Yang, L., Hu, S., Chen, D., Hu, Y., and Shi, Y.: Quantization of fully convolutional networks for accurate biomedical image segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8300–8308, 2018.

153. Molnar, C., Jermyn, I. H., Kato, Z., Rahkama, V., Östling, P., Mikkonen, P., Pietiäinen, V., and Horvath, P.: Accurate morphology preserving segmentation of overlapping cells based on active contours. Scientific reports, 6(1):1–10, 2016.

154. Wang, X., He, W., Metaxas, D., Mathew, R., and White, E.: Cell segmentation and tracking using texture-adaptive snakes. In 2007 4th IEEE International Symposium on Biomedical Imaging: From Nano to Macro, pages 101–104. IEEE, 2007.

155. Bredies, K. and Wolinski, H.: An active-contour based algorithm for the automated segmentation of dense yeast populations on transmission microscopy images. Computing and Visualization in Science, 14(7):341–352, 2011.

156. Li, F., Zhou, X., Zhao, H., and Wong, S. T.: Cell segmentation using front vector flow guided active contours. In International Conference on Medical Image Computing and Computer-Assisted Intervention, pages 609–616. Springer, 2009.

157. Gamarra, M., Zurek, E., Escalante, H. J., Hurtado, L., and San-Juan-Vergara, H.: Split and merge watershed: A two-step method for cell segmentation in fluorescence microscopy images. Biomedical Signal Processing and Control, 53:101575, 2019.

158. Kachouie, N. N., Fieguth, P., and Jervis, E.: Watershed deconvolution for cell segmentation. In 2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pages 375–378. IEEE, 2008.

159. Koyuncu, C. F., Arslan, S., Durmaz, I., Cetin-Atalay, R., and Gunduz-Demir, C.: Smart markers for watershed-based cell segmentation. PloS one, 7(11):e48664, 2012.

160. Sharif, J. M., Miswan, M., Ngadi, M., Salam, M. S. H., and bin Abdul Jamil, M. M.: Red blood cell segmentation using masking and watershed algorithm: A preliminary study. In 2012 International Conference on Biomedical Engineering (ICoBE), pages 258–262. IEEE, 2012.

161. Jeong, H.-J., Kim, T.-Y., Hwang, H.-G., Choi, H.-J., Park, H.-S., and Choi, H.-K.: Comparison of thresholding methods for breast tumor cell segmentation. In Proceedings of 7th International Workshop on Enterprise networking and Computing in Healthcare Industry, 2005. HEALTHCOM 2005., pages 392–395. IEEE, 2005.

162. Chau, Z. H., Paranawithana, I., Yang, L., and Tan, U.-X.: Plant cell segmentation with adaptive thresholding. In 2018 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP), pages 1–6. IEEE, 2018.

163. Espinoza, E., Martinez, G., Frerichs, J.-G., and Scheper, T.: Cell cluster segmentation based on global and local thresholding for in-situ microscopy. In 3rd IEEE International Symposium on Biomedical Imaging: Nano to Macro, 2006., pages 542–545. IEEE, 2006.

164. Al-Hafiz, F., Al-Megren, S., and Kurdi, H.: Red blood cell segmentation by thresholding and canny detector. Procedia Computer Science, 141:327–334, 2018.

165. Piórkowski, A.: A statistical dominance algorithm for edge detection and segmentation of medical images. In Conference of Information Technologies in Biomedicine, pages 3–14. Springer, 2016.

166. Bensch, R. and Ronneberger, O.: Cell segmentation and tracking in phase contrast images using graph cut with asymmetric boundary costs. In 2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI), pages 1220–1223. IEEE, 2015.

167. Leskó, M., Kato, Z., Nagy, A., Gombos, I., Török, Z., Vigh Jr, L., and Vígh, L.: Live cell segmentation in fluorescence microscopy via graph cut. In 2010 20th International Conference on Pattern Recognition, pages 1485–1488. IEEE, 2010.

168. Yang, H.-F. and Choe, Y.: Cell tracking and segmentation in electron microscopy images using graph cuts. In 2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, pages 306–309. IEEE, 2009.

169. Browet, A., De Vleeschouwer, C., Jacques, L., Mathiah, N., Saykali, B., and Migeotte, I.: Cell segmentation with random ferns and graph-cuts. In 2016 IEEE International Conference on Image Processing (ICIP), pages 4145–4149. IEEE, 2016.

170. Arganda-Carreras, I., Kaynig, V., Rueden, C., Eliceiri, K. W., Schindelin, J., Cardona, A., and Sebastian Seung, H.: Trainable weka segmentation: a machine learning tool for microscopy pixel classification. Bioinformatics, 33(15):2424–2426, 2017.

171. Prasath, V. S., Kassim, Y. M., Oraibi, Z. A., Guiriec, J.-B., Hafiane, A., Seetharaman, G., and Palaniappan, K.: Hep-2 cell classification and segmentation using motif texture patterns and spatial features with random forests. In 2016 23rd International Conference on Pattern Recognition (ICPR), pages 90–95. IEEE, 2016.

172. Sommer, C., Straehle, C., Koethe, U., and Hamprecht, F. A.: Ilastik: Interactive learning and segmentation toolkit. In 2011 IEEE international symposium on biomedical imaging: From nano to macro, pages 230–233. IEEE, 2011.

173. Iglesias, J. E. and Sabuncu, M. R.: Multi-atlas segmentation of biomedical images: a survey. Medical image analysis, 24(1):205–219, 2015.

174. Warfield, S. K., Zou, K. H., and Wells, W. M.: Simultaneous truth and performance level estimation (staple): an algorithm for the validation of image segmentation. IEEE transactions on medical imaging, 23(7):903–921, 2004.

175. Rohlfing, T., Russakoff, D. B., and Maurer, C. R.: Performance-based classifier combination in atlas-based image segmentation using expectation-maximization parameter estimation. IEEE transactions on medical imaging, 23(8):983–994, 2004.

176. Weisenfeld, N. I. and Warfield, S. K.: Softstaple: Truth and performance-level estimation from probabilistic segmentations. In 2011 IEEE International Symposium on Biomedical Imaging: From Nano to Macro, pages 441–446. IEEE, 2011.

177. Landman, B. A., Asman, A. J., Scoggins, A. G., Bogovic, J. A., Xing, F., and Prince, J. L.: Robust statistical fusion of image labels. IEEE transactions on medical imaging, 31(2):512–522, 2011.

178. Akhondi-Asl, A., Hoyte, L., Lockhart, M. E., and Warfield, S. K.: A logarithmic opinion pool based staple algorithm for the fusion of segmentations with associated reliability weights. IEEE transactions on medical imaging, 33(10):1997–2009, 2014.

179. Akhondi-Asl, A. and Warfield, S. K.: Simultaneous truth and performance level estimation through fusion of probabilistic segmentations. IEEE transactions on medical imaging, 32(10):1840–1852, 2013.

180. Langerak, T. R., van der Heide, U. A., Kotte, A. N., Viergever, M. A., Van Vulpen, M., and Pluim, J. P.: Label fusion in atlas-based segmentation using a selective and iterative method for performance level estimation (simple). IEEE transactions on medical imaging, 29(12):2000–2008, 2010.

181. Ding, Z., Han, X., and Niethammer, M.: Votenet: A deep learning label fusion method for multi-atlas segmentation. In International Conference on Medical Image Computing and Computer-Assisted Intervention, pages 202–210. Springer, 2019.

182. Ding, Z. and Niethammer, M.: Votenet++: Registration refinement for multi-atlas segmentation. arXiv preprint arXiv:2010.13484, 2020.

183. Bixby, B.: The gurobi optimizer. Transp. Re-search Part B, 41(2):159–178, 2007.

184. Lima, R. and Seminar, E.: Ibm ilog cplex-what is inside of the box. In Proc. 2010 EWO Seminar, pages 1–72, 2010.

185. Hutschenreiter, L., Haller, S., Feineis, L., Rother, C., Kainmüller, D., and Savchynskyy, B.: Fusion moves for graph matching. arXiv preprint arXiv:2101.12085, 2021.

186. Haller, S., Prakash, M., Hutschenreiter, L., Pietzsch, T., Rother, C., Jug, F., Swoboda, P., and Savchynskyy, B.: A primal-dual solver for large-scale tracking-by-assignment. In International Conference on Artificial Intelligence and Statistics, pages 2539–2549. PMLR, 2020.

187. Bártová, E., Šustáčková, G., Stixová, L., Kozubek, S., Legartová, S., and Foltánková, V.: Recruitment of oct4 protein to uv-damaged chromatin in embryonic stem cells. PLoS One, 6(12):e27281, 2011.

188. Liu, D., Zhang, D., Song, Y., Zhang, C., Zhang, F., O'Donnell, L., and Cai, W.: Nuclei segmentation via a deep panoptic model with semantic feature fusion. In Proceedings of the 28th International Joint Conference on Artificial Intelligence, pages 861–868. AAAI Press, 2019.

189. Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A.: beta-vae: Learning basic visual concepts with a constrained variational framework. Iclr, 2(5):6, 2017.

190. Lee, W., Kim, D., Hong, S., and Lee, H.: High-fidelity synthesis with disentangled representation. In European Conference on Computer Vision, pages 157–174. Springer, 2020.

191. Tran, L., Yin, X., and Liu, X.: Disentangled representation learning gan for pose-invariant face recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1415–1424, 2017.

192. Becker, S. and Hinton, G. E.: Self-organizing neural network that discovers surfaces in random-dot stereograms. Nature, 355(6356):161–163, 1992.

193. Hadsell, R., Chopra, S., and LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), volume 2, pages 1735–1742. IEEE, 2006.

194. Shoshan, A., Bhonker, N., Kviatkovsky, I., and Medioni, G.: Gan-control: Explicitly controllable gans. arXiv preprint arXiv:2101.02477, 2021.

195. Yüksel, O. K., Simsar, E., Er, E. G., and Yanardag, P.: Latentclr: A contrastive learning approach for unsupervised discovery of interpretable directions. arXiv preprint arXiv:2104.00820, 2021.

196. Ren, X., Yang, T., Wang, Y., and Zeng, W.: Do generative models know disentanglement? contrastive learning is all you need. arXiv preprint arXiv:2102.10543, 2021.

197. Deng, Y., Yang, J., Chen, D., Wen, F., and Tong, X.: Disentangled and controllable face image generation via 3d imitative-contrastive learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5154–5163, 2020.

198. Zhou, R., Helou, M. E., Sage, D., Laroche, T., Seitz, A., and Süsstrunk, S.: W2s: A joint denoising and super-resolution dataset. arXiv preprint arXiv:2003.05961, 2020.

199. LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.

200. Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., and Ha, D.: Deep learning for classical japanese literature. arXiv preprint arXiv:1812.01718, 2018.

201. Marsh, R.: The beetle. Broadview Press, 2004.

202. BioID Face Database | Face Detection Dataset | facedb.

203. Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., Tinevez, J.-Y., White, D. J., Hartenstein, V., Eliceiri, K., Tomancak, P., and Cardona, A.: Fiji: an open-source platform for biological-image analysis. Nature Methods, 9(7):676–682, July 2012.

204. Roth, S. and Black, M. J.: Fields of experts: A framework for learning image priors. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), volume 2, pages 860–867. IEEE, 2005.

205. Doersch, C.: Tutorial on variational autoencoders. arXiv preprint arXiv:1606.05908, 2016.

206. Karras, T., Aila, T., Laine, S., and Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196, 2017.

207. Ioffe, S. and Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International conference on machine learning, pages 448–456. PMLR, 2015.

208. Mohan, S., Kadkhodaie, Z., Simoncelli, E. P., and Fernandez-Granda, C.: Robust and interpretable blind image denoising via bias-free convolutional neural networks. arXiv preprint arXiv:1906.05478, 2019.

Selbstständigkeitserklärung

1. Hiermit versichere ich, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.
2. Bei der Auswahl und Auswertung des Materials sowie bei der Herstellung des Manuskripts habe ich Unterstützungsleistungen von folgenden Personen erhalten: (keine)
3. Weitere Personen waren an der geistigen Herstellung der vorliegenden Arbeit nicht beteiligt. Insbesondere habe ich nicht die Hilfe eines kommerziellen Promotionsberaters in Anspruch genommen. Dritte haben von mir weder unmittelbar noch mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.
4. Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt und ist auch noch nicht veröffentlicht worden.
5. Ich bestätige, dass ich die geltende Promotionsordnung der Fakultät Informatik der Technischen Universität Dresden anerkenne.

Mangal Prakash
April 1, 2022
Dresden