DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Robotics, Cognition, Intelligence

3D Instance Segmentation of an Unlabeled Modality via Cyclic Segmentation GANs

Leander Lauenburg

DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Robotics, Cognition, Intelligence

3D Instance Segmentation of an Unlabeled Modality via Cyclic Segmentation GANs

3D-Instanz-Segmentierung einer nicht gelabelten Modalität mittels zyklischer Segmentierungs-GANs

Author:	Leander Lauenburg
Supervisor:	PD Dr. Tobias Lasser
Advisor:	Zudi Lin, PhD
Submission Date:	15.09.2022

I confirm that this master's thesis in robotics, cognition, intelligence is my own work and I have documented all sources and material used.

Munich, 15.09.2022 Leander Lauenburg

Copyright

Acknowledgments

First and foremost, I would like express my gratitude for Ph.D. Zudi Lin, who supervised my thesis. Thank you for sharing your expertise and advice with me. You taught me a lot professionally and helped me grow personally. Your skills and work ethic were an inspiration every step of the way. I also thank Prof. Hanspeter Pfister for enabling my collaboration with Harvard's Visual Computing Group (VCG) and granting me access to Harvard's resources. I am grateful to the entire VCG for their continued support. Research is much more fun with a great team. I would also like to thank Prof. Edward Boyden, who leads MIT's Synthetic Neurobiology Group and especially Ph.D. Ruihan Zhang for the close and insightful collaboration. Further, I am grateful for PD Dr. Tobias Lasser's formal supervision of my master's thesis. Tobias, your general support for students like me who write their thesis abroad is deeply appreciated. Finally, I want to acknowledge the German Academic Exchange Service (DAAD) for supporting me through a fellowship within the IFI program.

Abstract

In this work and its companion paper, we propose a novel *Cyclic Segmentation* Generative Adversarial Network (**CySGAN**) for 3D domain adaptive instance segmentation. We developed CySGAN for nuclei segmentation in connectomics. Instance segmentation of 3D data is critical for connectomics as the field aims to identify and track the entire neural structures of the human brain. While manual segmentation is not feasible, automatic annotation usually requires training data with expert annotations that are expensive and time-consuming to obtain. In addition, imaging methods with sufficient resolution for connectomics and available annotated data generally have weak points like high costs and low throughput while suffering from stitching and alignment artifacts. At the same time, new imaging techniques that have the potential to alleviate some of these weak points are slow to be adopted precisely because of the lack of expert annotations. As a result, 3D instance segmentation of unlabeled image domains is desirable. However, existing methods are predominantly 2D and designed for semantic segmentation. Moreover, almost all existing methods segment a new modality either by using pre-trained models optimized on diverse training data or by performing image translation and segmentation sequentially with two relatively independent networks. CySGAN performs image translation and instance segmentation simultaneously using a unified weight-sharing network. The weight-sharing between the segmentation and the translation component has multiple benefits. First, both components are aware of the final objective at all times. Second, the segmentation introduces structural constraints on the image transformation. Third, the computational complexity is significantly lower compared to sequential models. At the time of inference, we can even remove the image transformation layer, reducing the computational cost of CySGAN to that of a standard segmentation model. We build up on CycleGAN's adversarial and cyclic consistency losses for the image translation. We facilitate the image translation through implicit structural constraints induced by the segmentation components. The segmentation builds on supervised losses that leverage the segmentation from the labeled domain. We complement the supervised losses with self-supervised structural consistency and structure-based adversarial objectives, leveraging unlabeled images of the target domain. We evaluate our method on a newly collected densely annotated Expansion Microscopy (ExM) nuclei dataset that matches a publicly available Electron Microscopy (EM) nuclei dataset. The unpaired EM and the ExM datasets both image parts of the zebrafish brain. The ExM domain is our target domain and the EM is our source domain.

Our CySGAN outperforms both pre-trained and sequential models, as well as a modified 2D semantic segmentation model that we manually adapted to 3D instance segmentation due to the lack of available frameworks. The implementation of CySGAN and the newly collected ExM dataset named *NucExM* are publicly available at `https://connectomics-bazaar.github.io/proj/CySGAN/index.html`.

Nomenclature

X	X (referring to EM) domain comprised of images I_X and segmentation S_X
Y	Y (referring to ExM) domain comprised of images I_Y
I_X	Images from the X domain
I_Y	Images from the Y domain
S_X	Instance segmentation masks from the X domain
S_Y	Instance segmentation masks from the Y domain
$I_{X'}$	Synthetic X images - images transferred from the Y domain to the X domain
$I_{Y'}$	Synthetic Y images - images transferred from the X domain to the Y domain
x_i	An explicit image from $I_{X'}$, i.e., $x_i \sim I_X$
y_i	An explicit image from Y, i.e., $y_i \sim I_Y$
$\hat{x}_i$	An explicit image from $I_{X'}$, i.e., $\hat{x}_i \sim I_{X'}$
$\hat{y}_i$	An explicit image from $I_{Y'}$, i.e., $\hat{y}_i \sim I_{Y'}$
x_s	A ground truth instance segmentation from S_X domain, i.e., $x_s \sim S_X$
$\hat{x}_s$	A generated instance segmentation for an image of I_X or $I_{X'}$, i.e., $\hat{x}_s \sim F(I_X)_{[S]}$ or $\hat{x}_s \sim G(F(I_X)_{[I]})_{[S]}$
$\hat{y}_s$	A generated instance segmentation for an image of I_Y or $I_{Y'}$, i.e., $\hat{y}_s \sim F(I_Y)_{[S]}$ or $\hat{y}_s \sim F(G(I_Y)_{[I]})_{[S]}$
x_i^*	In context of the augmentation schema, an explicit image from I_X that is not augmented, i.e., $x_i^* \sim I_X$
y_i^*	In context of the augmentation schema, an explicit image from I_Y that is not augmented, i.e., $y_i^* \sim I_Y$
F	A generator transferring images from the source to the target domain ($X \to Y$) - also called forward generator
G	A generator transferring images from the target to the source domain ($Y \to X$) - also called backward generator

D_X^I Discriminator for the images in the X domain

D_X^S Discriminator for the concatenation of the instance representation maps in the X domain

D_Y^I Discriminator for the images in the Y domain

D_Y^S Discriminator for the concatenation of the instance representation maps in the Y domain

Acronyms

AE Autoencoder.

AI Artificial Intelligence.

AN Artificial Neuron.

ANN Artificial Neural Network.

BCE Binary Cross Entropy.

CMSC Cross-modality Semantic Consistency.

CNN Convolutional Neural Network.

CySGAN Cyclic Segmentation Generative Adversarial Network.

DL Deep Learning.

EM Electron Microscopy.

ExM Expansion Microscopy.

FE Feature Engineering.

FFN Flood-filling network.

FOV Field of View.

GAN Generative Adversarial Network.

I2I Image-to-Image.

IMSC Intra-Semantic Consistency Loss.

L1 Absolute Error Loss.

L2 Squared Error Loss.

MAE Mean Absolute Error.

MIT Massachusetts Institute of Technology.

ML Machine Learning.

MSE Mean Square Error.

NMS Non-Maximum Suppression.

POM Predicted Object Map.

RMSE Root Mean Square Error.

SNG Synthetic Neurobiology Group.

SRE Signal to Reconstruction Error Ratio.

SSIM Structural Similarity Index Measure.

SSL Self-Supervised Learning.

U3D-BCD 3D U-Net for instance Segmentation.

VAE Variational Autoencoder.

VCG Visual Computing Group.

Glossary

AE Autoencoder: Self-supervised ANNs comprised of an encoder and a decoder. The encoder extracts a latent representation. The decoder reconstructs the input or an augmented version from the latent representation..

ANN Artificial Neural Network: Nonlinear functions approximator modeled after the biological model of the interconnection of neurons in the nervous system of a living being.

BCE Binary Cross Entropy: A loss function used in binary classification. A sample can belong to one of two possible classes.

CMSC Cross-Modality Semantic Consistency: A loss that ensures that the segmentation of an original image and the segmentation of a corresponding transferred image are as similar as possible.

CNN Convolutional Neural Network: A particular class of artificial neural networks (ANNs) that builds up on the discoveries that the neurons which make up the visual cortex have a small local receptive field. A receptive field is the area of action that an individual sensory neuron reacts to. The sum of the local receptive fields - which partly overlap - makes up the whole visual field (in the case of the visual cortex).

CySGAN Cyclic Segmentation GAN: CNN-based adversarial network that conducts image translation and instance segmentation jointly using a unified framework with weight sharing.

DL Deep Learning: Sub-field of ML that relies on ANNs and eradicates the need for FE.

Domain Transfer In the context of this thesis, a domain transfer refers to the conversion of an image from one domain, e.g., captured with EM, to a second domain, e.g., the domain of ExM. Classically, such a transfer assumes an underlying relationship between the two domains, e.g., when the same object is imaged with different image modalities. .

EM Electron Microscopy: Imaging method that uses a beam of accelerated electrons, instead of the photons used by light microscopy.

ExM Expension Microscopy: Imaging method that introduces a swellable polymer network into a sample. The polymer is then expanded by dialysis in water, resulting in an up

to 4.5-fold mechanical, linear, and isotropic sample expansion. As a result, diffraction-limited light microscopes can effectively reach 4.5 times their usual resolution when combined with ExM.

FFN Flood-Filling Network: CNN-based segmentation model that generates step by step a POM using a moving field of view (FFN).

FOV Field of View: A selected image area that is processed by a FFN at each iteration.

GAN Generative Adversarial Network: Generative models based on an adversarial two-player min-max game between a simultaneously trained generator and discriminator.

I2I Image-to-Image: Generally refers to Image-to-Image (Translation) - translating an image from a source domain to a target domain while attempting to preserve at least some of the context.

IMSC Intra-Modality Semantic Consistency: A cycle-consistency-like loss that ensures that the segmentation of the reconstructed image is as similar as possible to the segmentation of the original image.

Instance Segmentation Identifying and tracing each instance of one or multiple semantic classes within a dataset and assigning them instance-specific labels.

L1 Absolute Error Loss: The absolute differences between the ground truth and the predicted value.

L2 Squared Error Loss: Squared differences between the ground truth and the predicted value.

MAE Mean Absolute Error: Loss that minimizes the error that is the sum of all absolute differences between the ground truth and the predicted value.

MSE Mean Square Error: Loss that minimizes the error that is the sum of all squared differences between the ground truth and the predicted value. The loss is commonly used for regression tasks.

NMS Non-Maximum Suppression: An algorithm that suppresses bounding boxes that overlap with other bounding boxes by a given threshold and have a lower confidence rate.

POM Predicted Object Map: A segmentation mask that an FFN incrementally generates.

RMSE Root Mean Square Error: A measure of the difference between samples derived by taking the root of the MSE.

Semantic Segmentation Identifying and grouping semantically similar objects in images and assigning them class-specific labels.

SNG Synthetic Neurobiology Group: MIT's Synthetic Neurobiology Group led by Prof. Edward Boyden.

SRE Signal to Reconstruction Error Ratio: A method of measuring the difference between two images by evalutating the error in relation to the signal's power. This makes images with different brightness comparable. A high value refers to high similarity.

SSIM Structural Similarity Index Measure: A method used to measure the similarity between two images. A high value refers to high similarity.

SSL Self-Supervised Learning: A learning form that works without explicitly annotated labels by exploiting the data's inherent structure as supervisory signals. Often used to initialize the weights of ANNs.

U3D-BCD A 3D UNet used for instance, segmentation that generates the three representation maps: background (B), contour (C), and distance map (D).

VAE Variational Autoencoder: A AE whose latent space follows a standard normal distribution. The encoder outputs a mean and a variance.

VCG Visual Computing Group: Harvard's Visual Computing Group led by Prof. Hanspeter Pfister.

List of Figures

List of Tables

Contents

1. Introduction

This thesis is based on a collaboration between Hanspeter Pfister's Visual Computing Group (VCG) at Harvard and Massachusetts Institute of Technology's (MIT) Synthetic Neurobiology Group (SNG) led by Edward Boyden, a co-developer of the ExM method. Our work aims to perform instance segmentation for the unlabeled ExM modality. The goal is to eliminate the expensive and time-consuming step of collecting manual expert annotations.

1.1. Motivation

Today, our comprehensive understanding of human organs and their functions at the microscopic level enables, for example, the development of effective drugs. However, this generally excludes the brain, as our knowledge of this organ lags far behind the others. According to Jeff Lichtman, director of the Lichtman Laboratory at Harvard, there are two main reasons for this. Firstly, the brain is organized over six orders of magnitude from centimeters down to nanometers [1]. Secondly, unlike other organs, the brain's structure does not simply generate its functionality, but its functionality, in turn, generates its structure [1]. Nevertheless, a deeper understanding of the brain would significantly impact drug development and our knowledge of mental activities, behaviors, and neurological and psychiatric disorders [2].

Connectomics aims to improve our understanding by creating a structural and functional wiring diagram of the brain called the connectome [3, 2]. So far, connectomics relies heavily on EM since the organization of the brain extends down to the nanometer level. However, EM is expensive and slow. Thus, creating large brain tissue datasets using EM while maintaining low error rates remains a challenge [3, 2]. The reasonably new imaging method ExM has the potential to circumvent some weak points of EM. ExM enables diffraction-limited light microscopes to reach effectively seven times their usual resolution [4, 5]. Light microscopes are more available than electron microscopes, have higher throughput, and can handle samples ten times larger than EM. Additionally, the synergy effects, such as the reduction of cutting and alignment errors due to the larger sample size, are considerable [4]. However, in order for ExM to be useful for connectomics, we must be able to create the appropriate instance segmentation masks (in this work, we use the terms "segmentation" and "label" interchangeably). Instance segmentation identifies and traces individual object instances. Such a segmentation builds the base for the structural map of the human connectome. For example, segmenting cell nuclei from volumetric microscopy data enables us to study cellular expression patterns and cell lineages [6]. Manual segmentation of modern brain tissue datasets is not feasible. Abbott *et al.* [7] estimate that approximately 1 million terabytes of data must be acquired and analyzed for the construction of the mouse connectome. The human

brain is 1,000 times larger than that of the mouse and contains 1,000 times as many neurons [7]. The unprecedented data volume and complexity of the data require innovative advances, both computational and algorithmic, in automating data processing and labeling. The currently most promising approaches for automatic segmentation use Artificial Intelligence (AI).

Training an AI for a segmentation task generally requires segmented training data. However, the manual segmentation of even small training datasets in connectomics still requires skilled experts and hundreds of work hours. Accordingly, circumventing the annotation step is highly desirable, and many solution approaches exist. The probably most promising and versatile approaches use domain adaptive segmentation. Domain adaptive segmentation applies strategies to close the domain gap between a labeled training (source) and an unlabeled test (target) dataset to leverage the labels of the former for the segmentation of the latter [8]. Several earlier contributions in domain adaptive segmentation [9, 10, 11, 12, 13] attempted to solve the described problem for other image domains - predominantly computed tomography and magnetic resonance imaging. Those works are primarily 2D and built for semantic segmentation. Moreover, these methods segment new modalities by either using a pre-trained model optimized on diverse training data or performing domain translation and image segmentation sequentially, as shown in Fig. 1.1. The first approach either lacks performance or involves a fine-tuning phase requiring domain-specific labels. The weakness of the sequential approach is that the segmentation depends on a translation model optimized regardless of the downstream task. Additionally, two separate modules increase the pipeline complexity and computational cost.

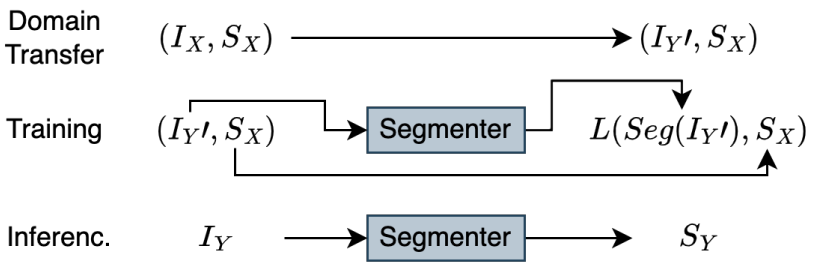
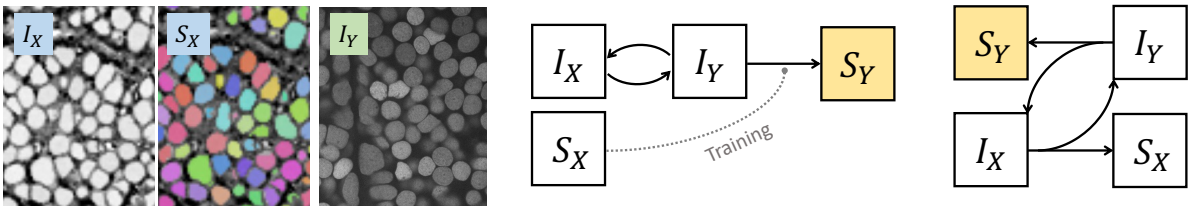


Figure 1.1: Sequential approach: First, domain transfer from I_X to $I_Y\prime$. Second, training a segmentation model on $(I_Y\prime, S_X)$. Third, using the segmentation model to segment I_Y. (I_X, S_X) is a labeled Electron Microscopy (EM) dataset, I_Y is a unlabeled Expansion Microscopy (ExM) dataset, $I_Y\prime$ is an EM dataset styled as an ExM dataset.

We found that the overall performance can be significantly increased by jointly conducting image translation and instance segmentation using a unified framework with weight sharing (see Sec. A.3.2 for an explation of weight sharing). To our knowledge, no relevant work to date performs a 3D domain transformation from EM to ExM or applies weight sharing in a domain adaptive 3D instance segmentation setting. Thus, our work provides connectomics with early access to the relatively new ExM image modality and advances the field of domain adaptive 3D instance segmentation.



(a) Source and target domains (b) CycleGAN + Segm (c) CySGAN (Ours)

Figure 1.2: Overview of task and methods. **(a)** We aim to segment an unlabeled target domain (I_Y) by leveraging the images (I_X) and masks (S_X) in the source domain. Instead of **(b)** conducting image translation (e.g., via CycleGAN [14]) and instance segmentation as two separate steps, we propose **(c)** the CySGAN framework to unify the two functionalities, optimized with both image translation as well as supervised and *semi-supervised* segmentation losses.

1.2. Objective

We have two unpaired sets of images. The first set is a set of segmented images in the EM domain $X = (I_X, S_X)$ (I_X referring to the images, S_X referring to the segmentation masks) with a resolution of $0.51 \times 0.51 \times 0.48 \mu m$. We call the labled EM domain the source domain. The second set is a set of unlabeled images in the ExM domain $Y = (I_Y, \_)$ (I_Y referring to the images, _ referring to the absence of segmentation masks) with an effective resolution of $0.046 \times 0.046 \times 0.357 \mu m$. We call the unlabeled ExM domain the target domain. Both datasets depict part of the zebrafish brain.

We define our objective as follows:

Derive the instance segmentation for the newly acquired and unsegmented ExM data by effectively leveraging the data from the segmented EM domain.

We split the objective into three subtasks:

- Domain Transfer

  Segmentation models show insufficient cross-modality generalization capabilities - their performance generally decreases dramatically as the domain gap between the source and target data increases [9]. Accordingly, the segmentation of an image domain generally requires domain-specific labeled training data. We aim to circumvent the need for domain-specific manual annotations. To this end, we aspire to derive a model that reduces the impact of the domain shift. Following previous methods, we rely on an adversarial domain transfer to close the domain gap. In the context of this thesis, a domain transfer refers to the translation of an image from one image domain to a second image domain. Classically, such a transfer assumes an underlying relationship between the two domains. When applying the domain transfer to the EM images (I_X) we obtain a labeled synthetic ExM dataset $(I_{Y'}, S_X)$.

Therefore, part of our objective is to acquire a translator $F_{I_X \to I_Y}$ that applies an adversarial fitting. The translator learns to map images from the source domain EM to the target domain ExM such that an adversarial trained classifier can not distinguish the output $I_{Y'} = F_{I_X \to I_Y}(I_X)$ from I_Y, with $I_X \in X$ and $I_Y \in Y$ [14].

- Label Transferability

  We can use the synthetic dataset $(I_{Y'}, S_X)$ to train a segmentation model in the ExM domain, an approach that we generally refer to as *domain adaptive segmentation*. However, for domain adaptive segmentation it is not sufficient for the domain translator to trick the discriminator by adapting the style while neglecting instance-specific information. Instead, we require label portability throughout the domain transfer, i.e., a segmentation $x_s \in S_X$ must retain its relevance for a corresponding synthetic image $\hat{y}_i \in I_{Y'}$. Thus, another aspect of our objective is to condition the generator to create a bijective transformation function (a single input is uniquely associated with a single output and vice versa) that maintains the cell instances and their structure between the source and target images [15].

- Instance Segmentation

  The final part of our objective is to join the domain transformation component with a segmentation model. We aim to fuse the components in such a way that they constrain each other and perceive the domain adaptive segmentation as the sole uniform objective - instead of a sequential transformation followed by a segmentation. We will use structural and feature-based losses, weight sharing between the translation and the segmentation components, structural constraints, and a novel augmentation strategy to achieve our objective.

1.3. CySGAN

In this work and the accompanying paper, we propose Cyclic Segmentation Generative Adversarial Network (CySGAN) a cyclic adversarial network that fuses 3D image translation with instance segmentation. We developed CySGAN for segmenting nuclei instances of the fully unlabeled ExM image modality (Fig. 1.2c). Given two image modalities, X and Y, with corresponding labeled and unlabeled datasets, (I_X, S_X) and $(I_Y, \_)$, CySGAN uses domain adaptive segmentation to leverage (I_X, S_X) to learn to segment I_Y. CySGAN follows a cyclic pattern, learning to transfer images from one domain to the opposite domain before reconstructing the original, i.e., $X \to Y \to X$ and $Y \to X \to Y$. This pattern lets us apply a cycle consistency loss [14] between the original and reconstructed images. The cycle consistency loss conditions CySGAN to embed sample-specific information into the translations (see Sec. 3.1 for an explanation of cycle consistency and CycleGAN [14]) and helps us achieve a bijective transformation. CySGAN builds on two multi-task 3D U-Nets [16] $F : I_X \to (I_Y, S_X)$, and $G : I_Y \to (I_X, S_Y)$ (see Sec. A.2.2 for a description of U-Net [17]). Each U-Net takes a single image as input and generates a four-channel output. The first channel is

the domain-translated image. The other three channels are instance representation maps from which we later derive the segmentation using a marker-driven watershed algorithm (MW) (see Sec. A.1.1 for an explanation of the watershed algorithm). Thus, F and G perform two tasks simultaneously, generating the domain transfer and generating the instance representation maps. Accordingly, the segmentation and transformation components within F and G share all their weights, except for a single task-specific output layer. However, while F and G have identical architectures, they are logically separated entities that do not share weights with each other. The fusion of the segmentation and translation components has multiple synergy effects. Most significantly, the domain transformation is aware of the final instance segmentation objective and directly benefits from structural constraints introduced by the instance representation maps. To our knowledge, the only framework that uses a similar weight-sharing strategy as ours is SUSAN [18]. However, SUSAN [18] performs domain adaptive 2D semantic segmentation, not domain adaptive 3D instance segmentation. Moreover, SUSAN [18] only applies image translation and supervised segmentation losses. We extend on SUSAN's [18] losses by introducing structural consistency and segmentation-based adversarial losses. These losses let CySGAN better leverage the unlabeled domain images, connecting ideas from semi-supervised segmentation (see Sec. A.3.4 for an explanation of semi-supervised learning). In addition, we propose a novel cycle consistency strategy for augmentation. Here, we directly integrate data augmentations into the joint image transformation and segmentation and enforce cycle consistency between the reconstructed and the clean images instead of the augmented images. Combined with our model's 3D nature, this enables us to introduce occlusion and blurring to the task of 3D domain transformation without toppling the delicate balance between the generator and the discriminator. Our augmentation strategy acts as regularization and enables the model to restore corrupted regions, improving the model's performance and robustness. The improved performance is consistent with previous works that show the significant impact of augmentations like blurry, noisy, and missing regions on 3D instance segmentation models [19, 6].

1.4. NucEM

To benchmark CySGAN we curated and annotated, in a joint effort between the VCG and SNG, two ExM image volumes from an adult zebrafish brain tissue with dense cell nuclei (I_Y in Fig. 1.2a). The number of segmented cell nuclei totals 18.4K instances. These two volumes are complemented by a publicly available and labeled EM dataset (I_X and S_X in Fig. 1.2a). Segmentation of cell nuclei from volumetric microscopy images is an essential task in studying biological systems, enabling the study of cellular expression patterns, cell lineages, and single cell analysis. Further, cell nuclei segmentation can aid in image registration and alignment. We segment cell nuclei because they are relatively easy to identify, there is only one per cell, and their dimensions are relatively constant, unlike the soma.

2. Background

This chapter reviews the relevant background. First, we introduce the field of connectomics, appropriate imaging methods, and dataset-specific information. Afterwards, we discuss image segmentation and Image-to-Image (I2I) translation along with some of the most relevant base frameworks like Flood-filling network (FFN)s and U3D-BCDs for segmentation and Variational Autoencoder (VAE)s and Generative Adversarial Network (GAN)s for I2I translation. We added further algorithm and concept-specific information in the appendix (see Chap.A). However, we generally referenced these in the appropriate section.

2.1. Connectomics

The brain is the most complex organ in an animal's body, responsible for vital functions such as perception, emotions, and movement control. The diversity of cell types and the wide variation in neuronal cell architectures reflect this complexity and prevent the derivation of transferable general guidelines between different brain regions [20]. Further, while the soma (cell bodies) of human nerve cells are relatively small, their axon (nerve fiber) and dendrites (branched extensions of a nerve cell) can span a connectivity network that adds up to a meter in length [20]. Therefore, it is impossible to infer the full functionality of many neurons based on partial volume analyses, as we have to image the entire brain to capture all their subtle branches and connections. However, imaging the whole human brain with a resolution high enough to resolve all nerve cells' entire geometry is currently infeasible simply due to resource constraints [3]. Moreover, although the structure of the human nervous system might ultimately determine its function, it is necessary to analyze the chemical and electrical activity - like membrane potentials, known as action potentials - to derive the actual functionality [20].

Connectomics studies neural circuits intending to create the human connectome - the structural and functional wiring diagram of the human brain [3, 2] - to grow our knowledge and understanding of mental activities, behaviors, and neurological and psychiatric disorders [2]. Additionally, connectomics can help drug development and even derive new insides for AI development. For example, Convolutional Neural Networks (CNN)s, which have revolutionized machine learning and the field of computer vision, originate from the discovery of the structure and function of locally sensitive and orientation-selective cells in the cat's visual cortex [21]. According to Jeff Lichtman, director of the Lichtman Laboratory at Harvard, creating the connectome requires collecting and merging the following three types of maps:

- A structural map that traces the physical neuron circuits
- A map that observes the electrochemical signal flow

- A map that records how circuits and their signals vary and adjust over time as the circuits undergo changes

Our work focuses on the structural map. The structural map does not only depend on the cellular morphology but cares about the structural connectivity down to the individual synapses. A synapse is a neuronal connection through which a neuron transmits electrical or chemical signals towards other cells. The construction of this map faces three main challenges: imaging, volume reconstruction, and computational segmentation.

2.2. Imaging Methods

A suitable imaging method for connectomics must provide sufficient resolution, have a high throughput, and promote volume reconstruction, i.e., reduce stitching and alignment artifacts.

2.2.1. Electron Microscopy (EM)

EM is an imaging method that uses a beam of accelerated electrons for imaging, in contrast to the photons used by light microscopy. Due to their shorter wavelength, electrons can resolve more delicate features than optical light. Modern EM can magnify objects up to a million times their original size and resolve features smaller than one *nm* [22]. Although EM can only handle small sample sizes, has a low throughput, and is prone to stitching and alignment artifacts, the resolution limits of most other imaging methods make EM often the method of choice for connectomics. In addition, the latest advances in automated serial electron microscopy promise to increase the throughput of EM [2].

2.2.2. Expansion microscopy (ExM)

ExM is a reasonably new imaging modality developed at MIT by Fei Chen, Paul W. Tillberg, and Edward S. Boyden [4]. ExM introduces a polyelectrolyte gel as a swellable polymer network into a sample. The polymer is then expanded by dialysis in water, resulting in an up to 7-fold mechanical, linear, and isotropic expansion of the sample [4, 5]. The expansion factor matches the effective resolution increase achieved by diffraction-limited light microscopes during subsequent imaging of the expanded sample [4, 5]. The term ExM comprises the entire processing and imaging step. Light microscopes are more available than electron microscopes, have higher throughput, and can handle samples ten times larger than electron microscopes. The synergy effects, such as the reduction of cutting and alignment errors, are therefore considerable [4]. In the original paper, Chen *et al.* [4] show that ExM achieve a $\sim 70nm$ laterally and $\sim 200nm$ axially resolution in cultured cells and brain tissue. Depending on the probe, the authors derived a measurement error of $1\% - 4\%$ post-ExM [4]. The authors further refined the method since its first publication. In the original work, the maximum isotropic expansion was 4.5 times, in contrast to the 7-fold expansion we have achieved in this work. One advantage of ExM is that it can visualize nanoscale features in samples with scale

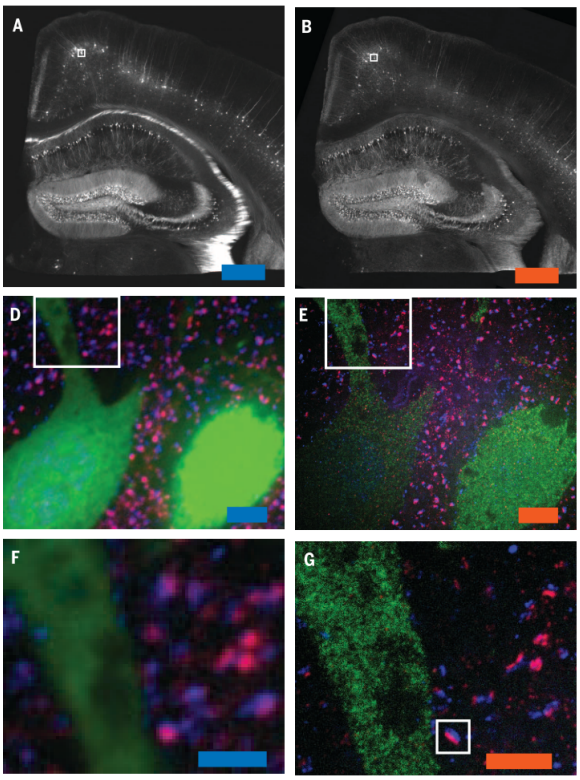


Figure 2.1: Both columns show the same mouse brain slice. The left column shows the specimen pre-expansion the right column shows the specimen post-expansion. The second and third rows show the magnification of the areas marked by the white boxes of the corresponding previous rows. [4, Chen *et al.*, p. 546]

sizes relevant to understanding neural circuits. Fig. 2.1 depicts a side-by-side comparison of a sample pre- and post-expansion.

2.3. 3D Nuclei Image Data

This thesis focuses on the analyses of cell nuclei in the brain of the zebrafish. Since nuclei are 3D structures, analyzing them in a 3D space is desirable. However, the creation and analysis of 3D datasets is much more complex than that of 2D datasets.

2.3.1. Cell Nuclei

A nerve cell's soma is its cell body. The axon is a long nerve cell projection that branches from the soma and transmits signals to neighboring cells. The dendrites are tree-like appendages that receive electrochemical signals from connected cells and transmit them to the soma [23]. Our NucExM dataset images cell nuclei. The nucleus (plural nuclei) is the center of the cell within the soma and contains all of the cell's genetic material [23]. The segmentation of cell

nuclei is a central task in connectomics that enables the study of cellular expression patterns, cell lineages, and single-cell analysis [6]. In addition, we use nuclei segmentation for volume reconstruction by helping with image registration and alignment. Another advantage of segmenting nuclei is that they are comparably easy to identify. There is only one nucleus per cell, and their dimensions are relatively constant, unlike the soma.

2.3.2. Zebrafish

The complexity of the human brain overwhelms our algorithms and storage capacities, and the creation of the human connectome is still a long way off. Therefore, it is common to test new imaging techniques and algorithms on less complex animals for which we can potentially image their entire neuronal structure at a resolution suitable for connectomics. The larva of the zebrafish is particularly well suited for connectomics research because it is transparent and has only about 100,000 neurons. Of these neurons, we can already map and physiologically record 80% in living, behaving animals and produce meaningful maps of neuronal activity [24]. Moreover, the neuroanatomy and neurochemistry of the larval zebrafish resemble that of mammals, and the animal exhibits complex sensory, motor, and cognitive functions [24]. According to Ma *et al.* [25],the derivation of functional connectivity, and thus interpretation of models of behavioral science and disorders in zebrafish is limited by the lack of detailed structural information about neuronal wiring. Ma *et al.* [25] point out that the existing mapping of neuronal wiring in zebrafish is not as comprehensive as other commonly used model organisms such as mice, fruit flies (Drosophila), and nematode (C. Elegans).

2.3.3. 3D Image Data

The connectome's structural map and all its subcomponents are 3D. However, datasets and methods for connectomics are still predominantly 2D. The reasons for this are threefold. First, acquiring 3D datasets is considerably more complex than acquiring 2D data. Second, processing 3D data is significantly more computationally expensive. Third, training 3D models is challenging as the solutions space grows exponentially with each additional dimension [3]. Structural imaging methods with a resolution appropriate for connectomics generally image 2D slices. Creating a 3D rendering from those 2D slices is, in most cases, a registration and alignment problem [3]. The 2D slices tend to be so small and thin that even the slightest interaction leads to distortions that make stitching and alignment artifacts inevitable. Additionally, mechanical limitations in cutting thin films lead to anisotropy, i.e., the resolution of the z-axis lags behind that of the x- and y-axes, where x and y represent the image plane, and z represents the depth axis [3]. Nevertheless, analyzing 3D data instead of 2D data is superior due to the reduced ambiguity through the richer contextual and structural information [12].

2.4. AI-driven Image Analysis

Brain tissue datasets have already begun to exceed petabytes in size, although they often map only a fraction of the neurological structures of a specimen. Shapson-Coe *et al.*[3] recently published the H1 dataset that images just one cubic millimeter of human brain tissue, while the storage requirements of the dataset already reached 1.4 petabytes. In this cubic millimeter of human brain tissue, Shapson-Coe *et al.*[3] segmented more than 130 million individual synapses. Accordingly, manual segmentation of modern brain tissue datasets is not feasible, and AI-driven processing and labeling algorithms have become the norm.

2.4.1. Segmentation

In the context of image analysis, segmentation generally refers to semantic segmentation or instance segmentation. Semantic segmentation identifies and groups semantically similar objects. Instance segmentation identifies and traces individual instances and assigns them instance-specific labels. In neural circuit reconstruction, we focus on instance segmentation as we want to reconstruct the circuit diagram based on all its individual components. Instance segmentation in connectomics is particularly challenging because we need to track cells in their entirety, including all axons or dendrites, to derive reliable circuit diagrams [26].

Flood-filling network

The FFN architecture, introduced by Januszewski *et al.*, is a well-known CNN segmentation model layout in connectomics. A flood-filling network has two input channels - one input channel for the 3D image data and one input channel for the Predicted Object Map (POM). The CNN's task is to create the POM step by step using a moving Field of View (FOV). We pass the previously created POM, the 3D image data, and an updated FOV to the model at each iteration. The voxel values of the POM range from 0 to 1, indicating whether a voxel belongs to the currently segmented object (1) or not (0). Before starting the segmentation, Januszewski *et al.* cleverly initialize the whole POM with object seeds, placing the seeds far from the object boundaries to prevent merging. Each object seed is processed individually. In the first iteration, we pass the model an image subvolume with the FOV centered around one of the object seeds and the corresponding POM with the object seed at its center. The model now estimates the values of the POM for the current FOV. If a voxel in the FOV is sufficiently far from the current center and assigned a value greater than 0.9, it gets queued as the center for a new FOV. The model repeats this process until the queue is empty. In the end, we apply a threshold to the POM that determines whether or not it contains an object [26]. The success of FFN builds on its ability to reduce segmentation splits and merges. A split occurs when the network identifies a single instance as two or more individual instances. A merge occurs when the network merges two or more instances into a single instance. The latter being considerably harder to proofread [26]. To reduce the number of merges, we can sample different object seeds, change the order in which the seeds are processed, or process the data in different resolutions. Januszewski *et al.* decreased the number of mergers in their

sample data by a factor of 82 while increasing the number of splits by only a factor of 2 [26]. On the other hand, we can reduce splits by identifying areas in which the proximity of two objects breaches a given threshold, reseeding the center of those two objects, and rerunning the FFN on those. If the created POMs overlap sufficiently, we merge the objects.

U3D-BCD

While a FFN computes the segmentation mask directly, a more common approach in connectomics is to derive intermediate representations from which we later infer the segmentation using, e.g., the watershed transformation (see Sec.A.1.1 for the watershed algorithm) [27]. Intermediate representations commonly comprise boundary [28, 17, 6] and affinity maps [29, 19, 6]. However, various fusions of representation maps have been tested [30, 31]. One such promising approach is U3D-BCD developed by Zudi *et al.* U3D-BCD is a 3D U-Net (see Sec. A.2.2 for U-Net) that generates three representation maps: the background (B), contour (C), and distance map (D) [6]. U3D-BCD builds on the fact that existing methods focus on the objects and neglect the background. Thus all background pixels are handled equally. Zudi *et al.* show that the segmentation model learns a refined background and foreground discrimination by introducing the distance map. The background and contour maps are learned via a Binary Cross Entropy (BCE) loss (see Sec. A.4.4 for BCE) since they are ultimately classifications (background or foreground and object boundary or non-object boundary). The distance map is learned via Mean Square Error (MSE) (see Sec. A.4.4 for MSE) regression and is defined as:

$$f(x_i) = \begin{cases} +\operatorname{dist}(x_i, B)/\alpha, & \text{if } x \in F. \\ -\operatorname{dist}(x_i, F)/\beta, & \text{if } x \in B. \end{cases} \tag{2.1}$$

with F and B denoting foreground and background masks, x_i a single pixel, and α and β as scaling parameters used to control the distance range [6]. In a consecutive step, the authors derive the instance segmentation from the instance representation maps using a watershed transform. The benefit of the U3D-BCD over the FFN is that it is way less computationally expensive and complex. In addition, U-Nets are very versatile and can also be used, for example, as the generator of a GAN (see Sec. 2.4.2).

Cellpose

While FFNs and U3D-BCD can produce incredible results, suitable training data is often the limiting factor. Cellpose is a method that claims to be a generalized deep learning-based segmentation framework that can bypass the need for task-specific training data. According to the authors, training with a versatile core dataset sourced from multiple laboratories using a variety of microscopy modalities and fluorescent markers allows Cellpose to segment a wide range of cell types without retraining or tuning hyperparameters [31]. Cellpose builts on a U-Net (see Sec. A.2.2 for U-Net) that generates two vector flow representations, a vertical and a horizontal flow. In the first iterations, the flow for each pixel within a cell points to adjacent pixels within the cell. In later iterations, the vector flow for pixels associated with

the cell converges toward the center of the cell. Subsequently, the pixels are grouped based on the flow field to form the segmentation. For the training data, the authors place a heat source in the center of each cell and apply a heat diffusion simulation to simulate vector flow. They choose as cell center the pixel closest to the median of the horizontal and vertical positions of all pixels associated with a cell [31]. An interesting design decision made by Cellpose is to apply global average pooling to the smallest convolutional maps to create a style representation and pass this as additional information to all upsampling layers [31]. The idea is that different cell types and imaging methods require different processing methods. The style representation allows the system to align itself accordingly.

StarDist

Like Cellpose, StarDist builds on a U-Net (see Sec. A.2.2 for U-Net) and claims to be a generalized deep-learning segmentation method. StarDist generates star-convex polygons as intermediate shape representation [32]. Star-convex polygons contain at least one point within themselves from which their whole boundary is visible. The authors argue that the star-convex polygons are perfectly suitable for approximating roundish shapes of cell nuclei as seen in 2D microscopy images. Their model predicts a star-convex polygon for each pixel by regressing the distance to the object boundary for a predefined set of radial directions. In addition, the model predicts whether a pixel belongs to the background or foreground, discarding star-shaped polygons that belong to background pixels. Next, StarDist applies Non-Maximum Suppression (NMS) to the candidate polygons and their associated object probabilities to determine the final set of polygons representing each object instance [32]. Schmidt *et al.* [33], the authors of StarDist, later extended their system to 3D by using Star-convex Polyhedra.

2.4.2. Image-to-Image Translation (I2I)

While generalized models such as StarDist and Cellpose can show surprisingly good performance, they are often not comparable to supervised methods. A common approach for getting closer to supervised performance in unlabeled modality segmentation is synthesizing a labeled in-domain dataset using a domain transfer. I2I translation models transfer images from a source domain to a target domain while attempting to preserve at least some of the context. Various applications include, for example, style transfer, restoration, and segmentation [34]. For this thesis, we care about domain adaptive segmentation that utilizes I2I translation models to transfer data from a labeled source domain to a target domain. Crucially the transfer has to preserve the underlying data structure and only transfer the style, otherwise, the labels would not be applicable anymore. We generally distinguish between supervised and unsupervised I2I translation. In the supervised setting, we have image pairs - matching images from the different domains. A classic example is images of the same landscape taken at different times of the year. In this case, the I2I translation model must perform a style transfer between the seasons. Such image pairs allow for pixel-wise losses, which evaluate model performance based on how similar the transferred image is to the paired target image.

In most cases, however, obtaining image pairs is difficult or infeasible. For example, EM and ExM require irreversible preprocessing steps and different sample sizes and thicknesses [3]. Therefore, unpaired I2I transfers focus on distribution matching using generative models [34]. The most commonly used generative models for I2I translation are Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) [34].

Variational Autoencoder (VAE)

Autoencoders (AE), initially developed for data compression, are self-supervised Artificial Neural Networks (ANN). AEs comprise two sub-models, an encoder and a decoder. Given some input, the encoder derives a low-dimensional latent space representation via a non-linear data dimensionality reduction. The latent space is a multidimensional space that structures the compressed data using a meaningful internal representation [35]. The decoder takes this latent space representation and reconstructs the original input. AEs learn based on a pixel-wise score between the original and reconstructed image. Accordingly, AEs are unsupervised models that show strong parallels to Self-Supervised Learning (SSL) (see Sec. A.3.3 for a definition of SSL). An optimal AE balances the highest possible data reduction rate against the lowest possible reconstruction error [35]. Nevertheless, in the case of classic AE we do not know how to sample the latent space since the space is neither organized nor structured. VAE extent the AE model by deliberately structuring the latent space to derive a common space suitable for a generation. The goal of VAEs is to encode the source data as a distribution over a latent space - instead of individual points in an unregulated latent space. VAEs accordingly train the encoder to output a mean and variance. However, without regulation, the latent space would likely split into multiple sub-distributions with little to no variation. For effective data generation, we require a complete and continuous latent space. Therefore, we regularize the distribution of a VAEs by forcing it to be close to a standard normal distribution. A well-trained VAE can be used for data generation by sampling a mean and variance from a normal distribution and feeding it to the decoder. Multiple extensions for VAE exist that make them suitable for I2I translation. Conditional VAEs, for example, let the user specify the desired output by conditioning the decoder on a specific data point. Accordingly, conditional VAEs take in a sampled mean, a variance, and an additional data point. In this setting, the mean and variance should only influence the style while the image defines the context [36]. However, although VAEs work very well for data generation, their data transformation capability often lags behind those of GAN-based models. Their use is mainly restricted to paired I2I translation [36, 37, 38, 39]. A further extension of VAEs that makes them suitable for unsupervised I2I translation are VAE-GANs [40, 41, 42]. VAE-GANs improve the generation capability of the VAEs by extending VAEs with a GAN-like discriminator and adversarial loss [40].

Generative Adversarial Networks (GAN)

GANs are generative models that build upon an adversarial two-player min-max game between a simultaneously trained generator and discriminator. The task of the generator is

to generate samples that match a given target distribution. We condition the generator with noise sampled from a fixed noise distribution. The discriminator must distinguish between generated and real data points [43]. The value function $V(G, D)$ modeling the two-player min-max game, as proposed by Goodfellow *et al.*, is defined as follows:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \tag{2.2}$$

D and G are two multilayer perceptrons [43]. G learns to generate samples from the distribution p_g over the data $\boldsymbol{x}$ based on a noise variable $\boldsymbol{z}$ that is restricted through the prior $p_z(\boldsymbol{z})$. $D(\boldsymbol{x})$ outputs a single confidence value based on whether $\boldsymbol{x}$ comes from p_g or from the original data. The optimal D outputs 1 for original data points and 0 for data points generated by G. G is trained to minimize $\log(1 - D(G(\boldsymbol{z})))$, i.e., fool D. D, on the other hand, aims to assign the correct label to both the training examples and the samples from G [43].

3. Related Work

This chapter discusses the relevant I2I translation models, 3D instance segmentation models, and domain adaptive segmentation models. The overall focus is on image adaptation and image segmentation mechanisms used by domain adaptive segmentation models.

3.1. Unpaired Image-to-Image Translation

Translating image data between domains can have many practical advantages like growing existing datasets, increasing the utility of tools by closing domain gaps, or improving the comparability between data. While an I2I translation model performs best when trained on image pairs, obtaining such pairs is often expensive or even infeasible. Accordingly, unpaired I2I translation [44, 14] is an active field of research. Currently, the most prominent methods for unpaired I2I-translation use GANs [43]. GANs comprise a generator that maps source

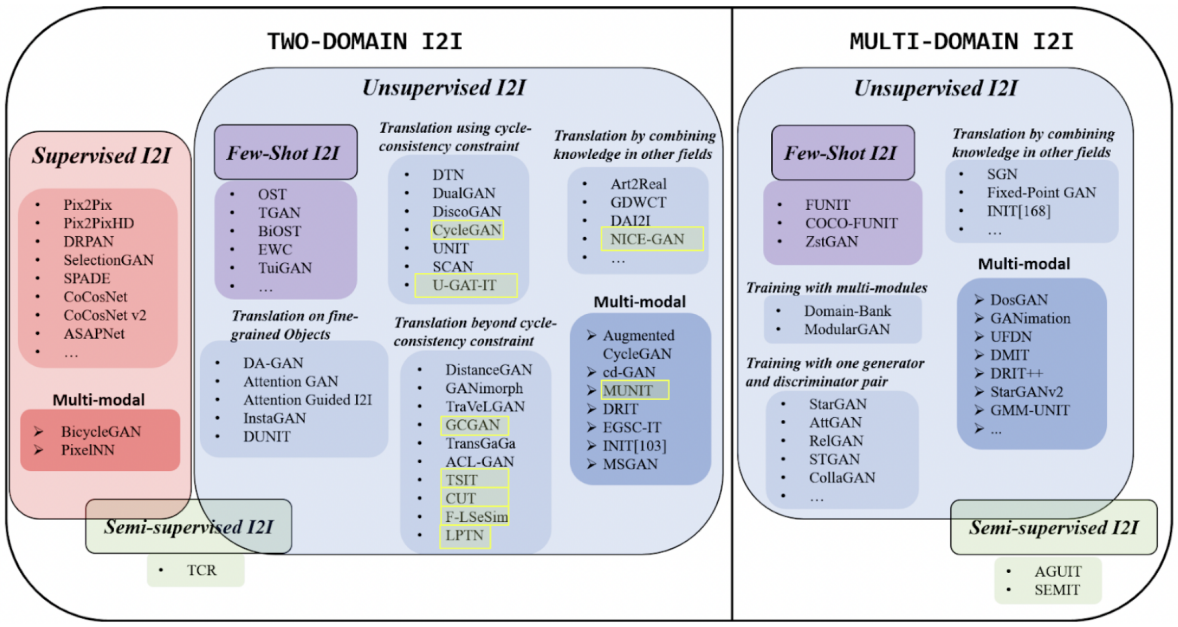


Figure 3.1: Overview of I2I methods from the paper "Image-to-Image Translation: Methods and Applications" by Pang et al. [34]. We tested and compared the methods highlighted in yellow. Our use case requires a two-domain system. We selected the methods based on results, citations, year of publication, code availability, and thematic suitability. [34, Pang *et al.*, p. 2]

images to the target domain and a discriminator. The discriminator decides whether an input image belongs to the real target distribution or is synthesized (see Sec. 2.4.2 for an explanation of GANs).

Figure 3.2: Qualitative results from the successfully tested unsupervised image to image translation models. First column on the left shows the ground truth satellite images. The last column on the right shows the ground truth Google Maps images. The columns in between correspond to the results from the methods indicated above each column.

We tested several models to determine which might be the best suited as the backbone for our system. Taking our cue from the study "Image-to-Image Translation: Methods and Applications" by Pang *et al.*, we selected models based on results, citations, year of publication, code availability, and thematic suitability [34] - we highlighted our model selection in yellow in Fig. 3.1. We evaluated the models based on the quality of the style transfer and the structural consistency. The second point is crucial for our task, as the labels must remain relevant throughout the domain transfer. We selected CycleGAN's paired Maps dataset [14] for the test and training dataset. We did train the models in an unpaired fashion. However, the image pairs allow us to evaluate the quality of the output. Further, the precise structure within the images, like roads and boxy houses, allows us to assess structural consistency throughout the domain transfer. Of the models tested, CycleGAN, F-LSeSim, CUT, FCUT, and GcGAN ran without problems. We got TSIT running after some discussions with the authors. With U-GAT-IT we encountered problems that we could not solve, and unfortunately, the authors did not respond to our requests for help. Our training of NICE-GAN ended in a mode collapse every time (see Sec. A.3.1 for mode collapse), no matter which hyperparameter we chose. LPTN showed inadequate performance. Through a discussion with the authors of LPTN, we learned that their model is better suited for transforming low-frequency components such as illuminations and colors. At the same time, the map dataset requires the transformation of

Table 3.1: **Quantitative unsupervised I2I evaluation.** The table depicts the results for the root mean square error (RMSE), structural similarity index (SSIM), and signal to reconstruction error ratio (SRE) for a selection of I2I methods. The results for each error term where derived by averaging over 52 samples generated by the corresponding models

Model	RMSE	SSIM	SRE
CycleGAN	0.0026	0.996	59.42
F-LSeSim	0.0028	0.995	59.36
CUT	0.0029	0.995	59.08
GcGAN	0.0033	0.993	58.22
TSIT	0.0029	0.995	58.75

high-frequency components.

Figure 3.2 depicts visual results from the (relevant) tested methods. A quantitative evaluation is shown in table 3.1. For the quantitative evaluation, we took the average over 52 samples of the Root Mean Square Error (RMSE), Structural Similarity Index Measure (SSIM), and Signal to Reconstruction Error Ratio (SRE) for each algorithm, respectively. For RMSE, a low value is desirable, while for SSIM and SRE, a high value is desirable. The two best-performing algorithms in our quantitive and qualitative tests are CycleGAN and F-LSeSim, with CycleGAN showing the best overall performance. Due to CycleGAN's overall performance, active development, cycle consistency approach, and broad acceptance in the community, we choose CycleGAN as our model's backbone.

For a more detailed discussion of the current state of the art of I2I translation models, please refer to the review by Pang *et al.* [34].

CycleGAN

Our study of relevant I2I translation methods yielded that CycleGAN [14] is well suited to function as the backbone of our method. CycleGAN achieves impressive performance by ensuring *cycle consistency* when transferring translated images back to the input domain. Further, CycleGAN's active development and broad acceptance in the community lead to constant updates and improvements of the model like shared high-level layers [41] and latent space alignment [45]. In its basic configuration, a GAN realizes a one-to-many mapping (one input can result in many different outputs) or even a many-to-many mapping (many outputs can correspond to many different inputs). However, in our case, we require a bijective transformation function (a single input is uniquely associated with a single output and vice versa) [15]. Otherwise, we will be unable to maintain the relevance of the segmentations throughout the transfer. CycleGAN's cycle consistency lets us supposedly achieve a bijective

translation function that guarantees a meaningful mapping between EM and ExM. Cycle consistency means that, for example, a sentence translated from one language to another and then back to the original language should be the same as the initial sentence [14]. To achieve cycle consistency, CycleGAN extents the classic GAN architecture (see Sec. 2.4.2 for an explanation of GANs) by implementing a second translator $G_{Y \to X}$, that aims to be the inverse of $G_{X \to Y}$ such that $G_{Y \to X}(G_{X \to Y}(I_X)) \approx I_X$ and $G_{X \to Y}(G_{Y \to X}(I_Y)) \approx I_Y$ [14]. We show the original schematic depiction of CycleGAN in Fig. 3.3.

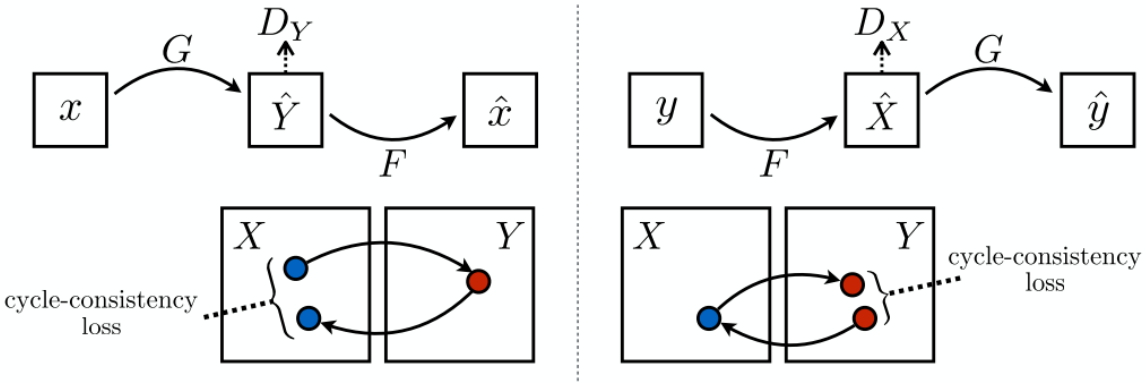


Figure 3.3: The image on the left depicts the forward cycle-consistency loss: $x \to G(x) \to F(G(x)) \approx x$. The image on the right depicts the backward cycle-consistency loss: $y \to F(y) \to G(F(y)) \approx y$. D_Y and D_X are the associated adversarial discriminators. D_Y trains G to translate images from X into images indistinguishable from Y. Same holds for D_X - just the other way around. [15, Zhu *et al.*, p. 3]

Nevertheless, for our objective, it is not sufficient for one generator to enable the other generator to approximate the original input by simply embedding enough information about the source domain. The following example clearly illustrates this:

$$G_{Y \to X}(G_{X \to Y}(I_X)) \approx I_X$$
$$\hat{G}_{Y \to X}(\hat{G}_{X \to Y}(I_X)) \approx I_X$$

The generators of the second equation are defined as $\hat{G}_{X \to Y} = G_{X \to Y} \circ T$ and $\hat{G}_{Y \to X} = G_{Y \to X} \circ T^{-1}$. T is a bijective geometric transformation, T^{-1} is the inverse geometric transformation of T, and $\circ$ is a concatenation operation. We can quickly see that if cycle consistency is given in the first case, it will also be given in the second case since T and T^{-1} cancel each other [9]. However, the underlying structure, which should be domain-invariant, is not necessarily the same between a transferred and the original image due to the geometric transformations T and T^{-1} [9, 10]. Further, Chu *et al.* [46] showed that CycleGAN hides information in nonvisible high-frequency components to achieve cycle consistency - a technique called steganography.

Figure 3.4 depicts samples from the NucEM dataset and corresponding synthesized ExM samples, generated with a vanilla 2D CycleGAN trained on the NucEM and NucExM dataset.

The translations seem reliable. However, all the samples depicted have a high instance distribution density. Further, the samples mainly contain features that are present in both datasets.

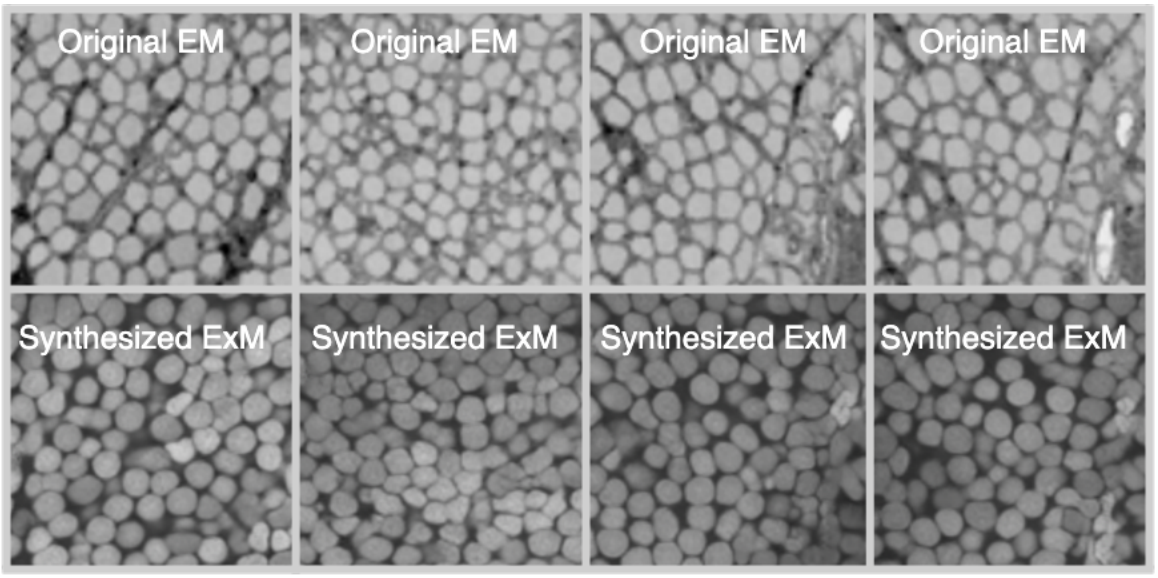


Figure 3.4: Examples generated with a vanilla 2D CycleGAN trained on NucEM and NucExM data. The top row shows samples from the NucEM dataset. The bottom row depicts the corresponding images translated with a vanilla CylceGAN model trained on the NucEM dataset.

Looking instead at the Fig. 3.5, we begin to see the non-bijective and hallucinatory tendencies of CylceGAN. The bone-like structures in the EM samples are not present in the NucExM training dataset. CycleGAN solves this by modeling them using nuclei. For areas without nuclei in the EM samples, CycleGAN begins to hallucinate nuclei to fill the space.

Figure 3.5 shows that the discriminative loss of the vanilla GAN is inherently flawed for our use case. For example, suppose the average object distribution density of the dataset is high. In that case, the translation model will begin to hallucinate objects for samples with lower than average object distribution density. It does so to increase the chance of fooling the discriminator. Figure 3.6 shows this effect clearly. The bottom row depicts the real targets, while the middle and top rows depict the corresponding transferred samples. We generated the top row after fifteen epochs and five decay epochs. In comparison, we generated the middle row after twenty epochs (continuing the training of the previous model from its state before the decay epochs) and ten decay epochs. The figure shows that CycleGAN preserves the data structure more clearly during low epoch rates, where the model still focuses on the general transfer. In contrast, at later epochs, the hallucinated objects become more numerous and have a higher intensity.

As mentioned earlier, in domain-adaptive instance segmentation, the structure within the

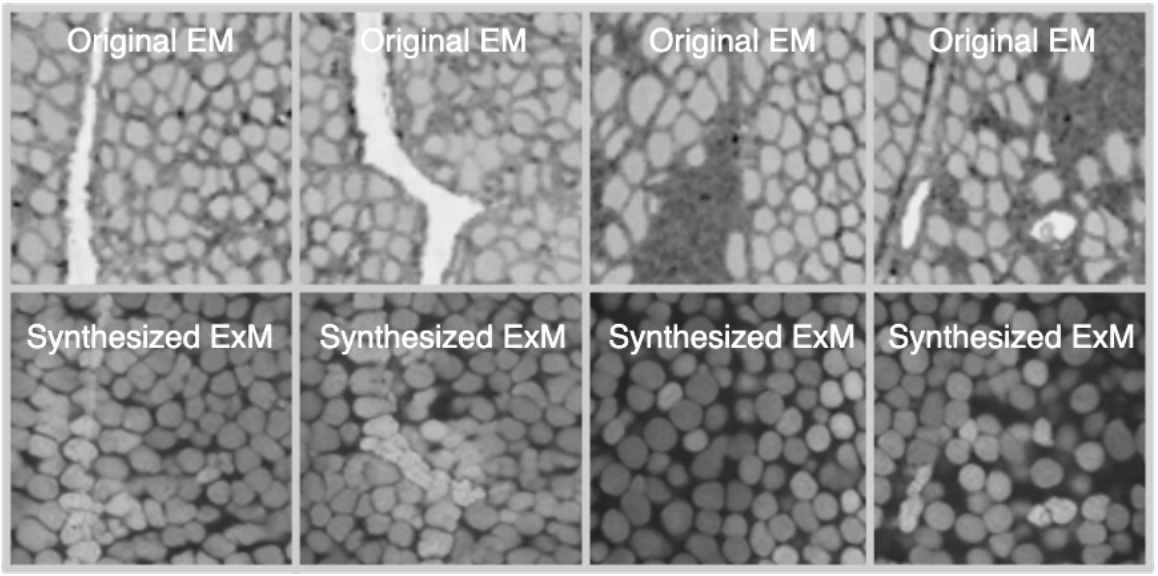


Figure 3.5: Examples generated with a vanilla CycleGAN trained on NucEM and NucExM data. The top row shows samples from the NucEM dataset. The bottom row depicts the corresponding images translated with a vanilla CylceGAN model trained on the NucEM dataset.

data must be preserved throughout the domain transfer for the labels to remain relevant. Accordingly, our analysis of CycleGAN makes it clear that we must either adapt the model's objective or constrain it to meet our use case.

3D CycleGAN

The 3D nature of our data adds a further layer of complexity to the problem. Zhang *et al.* [9] state that from an optimization and memory perspective, the end-to-end training of a 3D network is significantly more complex. However, considering the structural consistency and possible slice-to-slice inconsistencies, the problem would be best handled by a 3D CycleGAN [12]. The 3D rendering, in turn, is prone to alignment and stitching artifacts. In addition, the 3D representation often suffers from the anisotropy of the given imaging techniques. In general, the resolution of the z-axis lags behind that of the x- and y-axes, where x and y represent the image plane, and z represents the depth axis. The resolution deficit is mainly due to mechanical limitations in cutting thin films [3]. Zhang *et al.* [9] have multiple ideas for improving the performance of a 3D CycleGAN. First, they found that both top and bottom layer representations are critical for maintaining anatomical structures. Accordingly, Zhang *et al.* [9] propose to use long-range skip connections in their generators. Secondly, they claim to have achieved better results by applying less greedy downsampling and restricting their max downsampling rate to eight. Thirdly, they propose using nearest neighbor upsampling instead of transpose convolutions. Zhang *et al.* [9] base this adaption on the

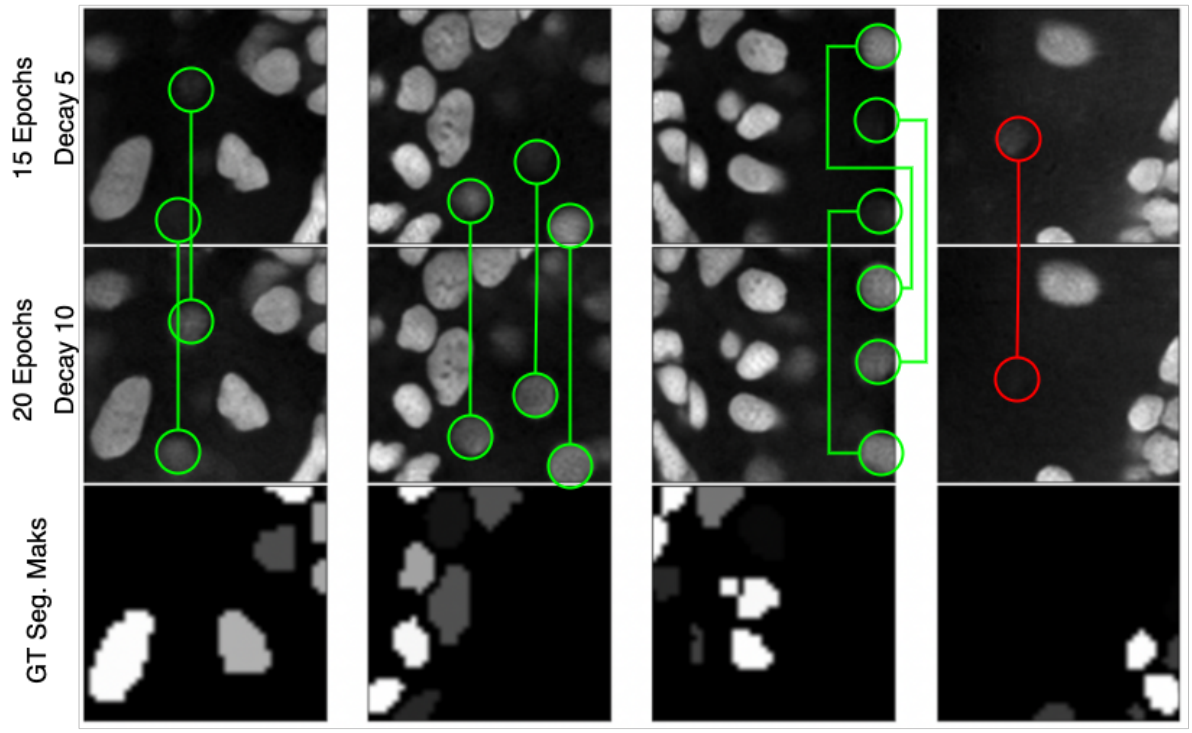


Figure 3.6: Examples generated with a vanilla CycleGAN trained on NucEM and NucExM data. The bottom row shows the real targets. The middle and top rows show the corresponding transferred samples. The top row was generated after fifteen epochs and five decay epochs, the middle row after twenty epochs (continuation of the training of the previous model), and ten decay epochs. Green marks the hallucinations that are stronger in the data generated after the more extended training period, while red marks the hallucinations that are stronger after a shorter training period.

finding that the uneven overlapping of convolutional kernels in transpose-convolutions can cause checkerboard artifacts. An effect that is supposedly even intensified in 3D transpose-convolutions. [47], as well as [11], propose the incorporation of attention gates to sharpen the network's focus on salient features.

For our CySGAN framework, we use the open-source implementation of U3D-BCD [6] (see Sec. 2.4.1 for an explanation of U3D-BCD). U3D-BCD [6] uses residual blocks instead of standalone convolutional layers and uses addition instead of concatenation to save memory. Following Zhang *et al.* we also use long-range skip connections. We use convolutional models with 3D instance norm and leaky Relu activation layers for the discriminators. The generators and discriminators use a mixture of 2D and 3D convolutions to deal with the anisotropy of the data (see Sec. 4.4 for more details).

3.2. Instance Segmentation of 3D Microscopy

Semantic segmentation of microscopy images from bio- and biomedical domains has many practical applications. For example, it plays a crucial role in understanding, diagnosing, and treating diseases [48]. Instance segmentation does not only assign a semantic class but also assigns individual labels to instances of the same class. Therefore, it enables a more refined analysis, such as phenotyping, tracking, and thus, e.g., neuronal circuit reconstruction [49]. However, the dense object distribution and varying instance morphology, particularly in biological data, can make instance segmentation a real challenge. Analyzing volumetric 3D data instead of individual 2D layers can significantly improve the accuracy of instance segmentation. Volumetric 3D data provides additional structural information and lets the model understand the objects better [33, 6, 48]. Nevertheless, 3D networks are hard to optimize and memory intensive [9]. Additionally, 3D renderings suffer from alignment artifacts, stitching artifacts, and anisotropy. These problems originate from the physical limitations of the imaging methods (see Sec. 2.2.1 and Sec. 2.2.2 for an explanation of the limitations of EM and ExM). Accordingly, 3D instance segmentation remains an open field of research. The variety of existing segmentation architecture is immense and, in some cases, can be highly specialized. An example for such a highly specialized method are FFNs [26] (see Sec. 2.4.1 for an explanation of FFNs). However, there seems to be an emerging trend towards a two-step approach for learning-based 3D microscopy segmentation models. In the first step, a CNN-based model derives intermediate instance representation maps. In the second step, the actual instance segmentation is derived by applying watershed transform [27, 50] (see Sec. A.1.1 for an explanation of the watershed algorithm) or graph partition [51]. Instance segmentation frameworks that follow these two-stage approaches vary mainly in the representation maps that they compute. Common intermediate instance representation maps include object boundary [28, 17, 30], affinity [52, 19] (see U3D-BCD in Sec. 2.4.1 for an explanation of the affinity map), star-convex distance [33] (see StarDist in Sec. 2.4.1 for an explanation of the star-convex map), and flow-field [31] maps (see Cellpose in Sec. 2.4.1 for an explanation of the flow-field map). Many frameworks use a combination of multiple of those or other representations [6, 30].

3.3. Domain Adaptive Segmentation

Domain adaptation is the method of choice to achieve close to supervised segmentation performance for unlabeled imaging modalities. Chartsias *et al.* [53] designed a two-stage framework that first translates label images to the unlabeled domain using CycleGAN [14] and then trains a separate segmentation model using the synthesized images and original ground-truth label. CyCADA [45], ICMSC [54], SIFA [10], and EssNet [55] improve the sequential model by jointly optimizing the translation and segmentation networks. However, using two separate models increases the system complexity in training and deployment. For example, the authors of CyCADA [45] stated that although their model is theoretically end-to-end trainable, they had to train it in stages as it was too memory-intensive in practice. Unlike

the mentioned works, we unify image translation and segmentation into a single model, significantly reducing the system's complexity. Since the translation and segmentation layers base their predictions on the same high-level features, CySGAN enforces the consistency between translated images and segmentation maps from an architectural perspective. We base this adaptation on the models SIFA [10] and Segmentation-Enhanced CycleGAN [13]. Both works show that cyclic models can be trained to inherently compute domain-invariant encodings by applying image- and feature-level adaptations. Further, the authors of both models show that we can directly derive the segmentations from these encodings. Thus, we argue that a distinctive segmentation model structure introduces unnecessary redundancy.

To our knowledge, the only existing work that explores simultaneously translating and segmenting with weight sharing is SUSAN [18]. SUSAN [18] is a 2D CycleGAN with two heads. One of those heads is a multi-class soft-max classification head that produces class probabilities for each pixel. The other head is an image translation head with a convolution layer that generates grayscale images at the resolution of the input images. Accordingly, SUSAN's [18] segmentation component and domain adaptation parts share weights. Our work differs from SUSAN in two main aspects. First, SUSAN and all the before-mentioned contributions are for 2D semantic segmentation. Our work focuses on the more challenging 3D instance segmentation. Second, SUSAN only applies supervised segmentation losses to the annotated domain. Our CySGAN leverages structural consistency and segmentation-based adversarial losses for the unlabeled domain in the absence of ground-truth labels.

3.3.1. Image- and Feature-level Adaptation

The segmentation in the classical sequential setup of domain adaptive segmentation - a domain transfer followed by a segmentation - is effectively a well-explored supervised task. Accordingly, relevant contributions generally focus on the domain adaptation step. Specifically, they focus on achieving a bijective transformation to maintain the relevance of the targets $x_s \sim S_x$ for the transferred source images $\hat{y}_i \sim I_y$. The probably most prominent method in domain adaptive 2D and 3D microscopy segmentation is CycleGAN (see Sec. 3.1 for an explanation of CycleGAN). Our analyses of I2I translation methods confirmed CycleGAN to be a good choice as the backbone for our method.

Various adaptations have been proposed for CycleGAN to achieve a real bijective transformation that upholds a sufficient structural consistency for domain adaptive segmentation [13, 10, 15, 11, 45, 54]. Generally, these methods build upon image- and feature-level adaptation.

Image level losses (otherwise also called appearance-level losses) operate on pixel level and focus on style-wise domain adaptation - CycleGAN's [14] cycle consistency loss is a great example. Feature-level adaptation focus on structural consistency through feature alignment. These losses try to preserve the object structures throughout the transformation.

For example, Januszewski and Jain [13] try to achieve structural consistency by introducing an additional feature-level discriminative segmentation loss to CycleGAN. They worked on the same problem as ours but applied it to two different domains. For consistency, we will refer to their labeled dataset as (I_X, S_X) and the unlabeled dataset as $(I_Y, \_)$. The loss proposed by Januszewski and Jain builds on a segmentation network independently trained

on the labeled dataset (I_X, S_X). During the CycleGAN training, they freeze the weights of the segmentation and use it to segment samples from (I_X, S_X) and synthesized samples $G_{Y\to X}(Y) = \widehat{X}$ (of the unlabeled dataset ExM). The authors then introduce an additional discriminator D_S to the CycleGAN setup that distinguishes whether a segmentation originates from (I_X, S_X) or synthesized data $G_{Y\to X}(Y)$. D_S is trained simultaneously with CycleGAN's other components. The adversarial loss of D_S is then backpropagated to $G_{X\to Y}$ and $G_{Y\to X}$ [13]. The error signal has to flow back through the segmenter and the discriminator. Januszewski and Jain, therefore, freeze the weights of the segmenter and discriminator ($D_S(x)$) before propagating the gradient of the loss function back to the generator.

Tomczak *et al.* [15] and Chen *et al.* [10] suggest using a single encoder for the different domains and varying only the decoder in the CycleGAN setup. The idea is that the encoder should encode only the anatomical structure information and thus be universal since this information is domain-independent. The authors condition the segmentation model by applying a pixel-wise cross-entropy loss between the segmentation and the ground truth. [15, 10].

Chen *et al.* [10] also try to achieve a domain invariant encoding with their SIFA model. To this end, in addition to the loss described above and the two classical adversarial CycleGAN losses, they propose two additional losses. The first loss uses the same segmentation model from above (shared weights) to create segmentation masks based on the encodings from samples of both datasets, the labeled and the unlabeled dataset [10]. They then train an additional discriminator simultaneously to the CycleGAN to determine the origin of the generated masks [10]. Chen *et al.* [10] argue that the anatomical shapes are domain invariant. Therefore the discriminator would be unable to distinguish the corresponding segmentation masks if the features extracted from both domains are aligned. For the second loss, Chen *et al.* [10] trains a discriminator that distinguishes whether an image is a synthesized image or a reconstructed image. According to Chen *et al.* [10], the idea behind this loss is that the encoding space is not yet domain invariant if the discriminator has successfully classified the image origin.

Tsai *et al.* [56] do not use a CycleGAN but still apply an adversarial loss similar to Chen *et al.* [10]. They train a single segmentation model to derive segmentation for images of both domains. In addition to a supervised loss between the targets of the source domain and the generated segmentation for the source domain, they train their model using an adversarial loss derived from a discriminator that evaluates from which domain a segmentation originates.

Hoffman *et al.*, the authors of CyCADA, propose a semantic consistency loss based on a noisy classifier [45]. The classifier is pre-trained on the labeled source images. While training the domain transfer model, Hoffman *et al.* use the classifier to classify the synthesized and reconstructed images. The goal is that the classifier classifies the output of the domain transfer model in the same way as the corresponding original images. This idea builds on the knowledge that higher-level layers in CNNs typically extract content information and that the content should change between the domains. Only the lower-level individual pixel values should be adapted [45].

According to Liu *et al.* [57], the authors of CyC-PDAM, they developed the first ever

unsupervised domain adaptation instance segmentation model. CyC-PDAM is a sequential model based on a CycleGAN for the domain transfer and a Mask R-CNN [58] for the instance segmentation. The framework builds up on CyCADA. Besides the original CycleGAN and Mask R-CNN losses, the authors also apply two additional domain adaptation losses. The first is an image adaptation loss that evaluates whether the features extracted by Mask R-CNN's FPN belong to the source or target domain. The second is an instance-adaptation loss that evaluates if an instance segmentation belongs to the source or target domain. Liu *et al.* [57] observe that Mask R-CNN leads to a domain bias in the semantic-level features as the model focuses on local features and not global semantic context. To combat the issue, they additionally generate the semantic segmentation mask from the features extracted by Mask R-CNN. A third adversarial loss then evaluates the origin of the semantic loss and backpropagates it to the generator [57].

Zeng *et al.*, the authors of ICMSC, propose a sequential model consisting of a CycleGAN and two segmentation models - one per domain [54]. They observe that previous works are missing a mechanism that guarantees consistent semantics between original images and cycle reconstructions. The cycle consistency loss ensures that an original image and its reconstructed counterpart are as similar as possible. However, this does not necessarily apply to the segmentation derived from them. Therefore, Zeng *et al.* propose an Intra-Semantic Consistency Loss (IMSC) loss that makes the reconstructed images' segmentations match the original images' segmentation. The IMSC loss is applied in both directions of the cycle [54]. Further, Zeng *et al.* observe that the segmentation of an original image and a corresponding domain-translated image should also be the same. Following this line of thought, they propose a Cross-modality Semantic Consistency (CMSC) loss that ensures that the segmentation of an original image and the segmentation of a corresponding domain translated image are as close as possible [54].

SUSAN [18] uses two additional image adaptation losses, besides the GAN and cycle consistency losses, for their two-headed CycelGAN-based adaptive domain segmentation model described earlier (Sec. 3.3). The first loss evaluates the pixel-wise similarity between the targets of the labeled domain and the segmentation generated by the generator that transfers from the labeled to the unlabeled domain. The second loss evaluates the pixel-wise similarity between the targets of the labeled domain and the segmentation generated by the second generator during the reconstruction of the source image [18].

[59] take a different approach to ensure structural consistency by applying a MIND-based loss. MIND stands for Modality independent neighborhood descriptor. The descriptor depends on local image structures instead of intensity. It uses a non-local patch-based self-similarity to capture the modality-invariant anatomical structures [59]. MIND derives structural descriptors for patches of an image by comparing them one after the other to all other patches in predefined proximity of the same image. They then derive a loss by comparing the patch-wise descriptors between the original and corresponding synthesized images.

Additionally, Yang *et al.* propose to improve the image-to-image translation through an affine image registration preprocessing step of the unpaired datasets. Image registration tries

to find the best possible alignment of one or multiple object images to a reference image through a compensating transformation. Based on their results, they argue that such a step can significantly improve the performance of unsupervised synthesis methods [59].

The methods described above show how we can reduce the domain shift through a combination of image- and feature-level adaptation. Chen *et al.* [10], Liu *et al.* [57], and Tsai *et al.* [56], e.g., introduce an adversarial loss that distinguishes between the segmentations generated from the labeled domain and the unlabeled domain. This loss is propagated to the encoder of the domain transfer models, facilitating a domain-invariant and feature-based encoding. We apply a version of this loss for output space adaptation by distinguishing the GT - not the targets generated for the labeled domain - from those generated for the unlabeled domain. We chose this configuration as it provides a robust learning signal from the beginning and naturally reduces the adversarial noise in the generated targets. Further, we use the loss for output space adaptation instead of feature adaptation as we perform instance segmentation on data with high instance density.

Tomczak *et al.* [15], and the authors of ICMSC [54], SIFA [10], and SUSAN [18] propose to apply a semantic consistency loss to the first half of the supervised cycle to control the segmentation process. The loss forces the synthesized segmentation masks close to the GT segmentation. SUSAN [18] further introduces the loss to the second half of the supervised cycle by applying them to the reconstructed segmentations and the GT. We adapted this loss for CySGAN. Nevertheless, the loss disproportionately favors the supervised cycle. We, however, ultimately only care about the segmentation performance of the unsupervised cycle. The authors of ICMSC [54] propose an IMSC loss that forces the segmentation of the reconstructed image to be the same as the segmentation of the original image. They apply the IMSC loss in both cycle directions. We adopted the IMSC loss for CySGAN. However, we argue that the unique design of our method makes more effective use of the loss since it does not depend on the performance of a logically separated segmentation network. The loss effectively strengthens the less supervised half of the cycle and stabilizes the training by increasing the generators' codependency.

Liu *et al.* [57] actually propose a unsupervised domain adaptation instance segmentation model. Nevertheless, their model is a complex sequential 2D model that suffers heavily from hallucinations. Accordingly, the translation and segmentation components are independent network components, and the domain adaption model fails in upholding label transferability. Liu *et al.* [57] combat the hallucinations using an auxiliary nuclei inpainting mechanism that manually removes the nuclei that appear in the synthesized images and do not have a corresponding segmentation. CySGAN is more compact and eliminates the need for such an inpainting mechanism due to its structural conditioning of domain matching induced by weight sharing.

Although our structural consistency loss and segmentation-based adversarial loss share similar ideas with existing works like ICMSC and SIFA, there are two significant distinctions. First, we show that those losses work with a 3D domain adaptation model that jointly conducts image translation and instance segmentation with weight sharing. Second, the referred works usually only use output space adaptation or target-based structural consistency losses. At the

same time, we combine both to better leverage the unlabeled target domain images during training.

3.3.2. Structural Conditioning

The primary purpose of the aforementioned image- and feature-adaptation losses is to achieve a bijective transformation, preserve the structure, and suppress hallucinations [14, 18, 18]. While the performance impact of hybrid representations on instance segmentation has already been proven [6], its impact on domain transfers, as an additional structural constraint, has so far not been explored. CySGAN generates the background, contour, and distance map simultaneously with the domain transformations. Our results indicate that by doing so, we impose structural constraints on the model (see Sec. 6.2 for the results). We argue that those constraints force the model to learn a deeper understanding of the objects, reducing blurring and merging in our challenging, cluttered dataset (see Sec. 4.4 for implementation details).

3.4. Augmentation

Augmentation for domain transformation typically plays a secondary role and is limited to affine transformations and contrast adaptations [10, 54]. Most of the heretofore-mentioned models do not apply augmentation at all. However, segmentation is known to benefit enormously from augmentation [19, 6]. In particular, we are unaware of any work that uses occlusions and blurring for domain adaptation - although this comparison may not be fair as existing methods are predominantly 2D. We implemented an augmentation scheme that masks each slice of a subvolume independently (see Sec. 4.4 for implementation details). At the same time, we do not corrupt consecutive slices to allow recovery. We argue that through occlusions and blurring, the model learns to pay more attention to the direct relationship between layers, i.e., the 3D nature of the data (see Sec. 4.4 for an explanation of our augmentation schema).

4. Method

In this chapter we first give an overview of the CySGAN framework (Sec. 4.1) and present the image translation (Sec. 4.2) and segmentation objectives (Sec. 4.3). Afterwards we describe the precise implementation (Sec. 4.4).

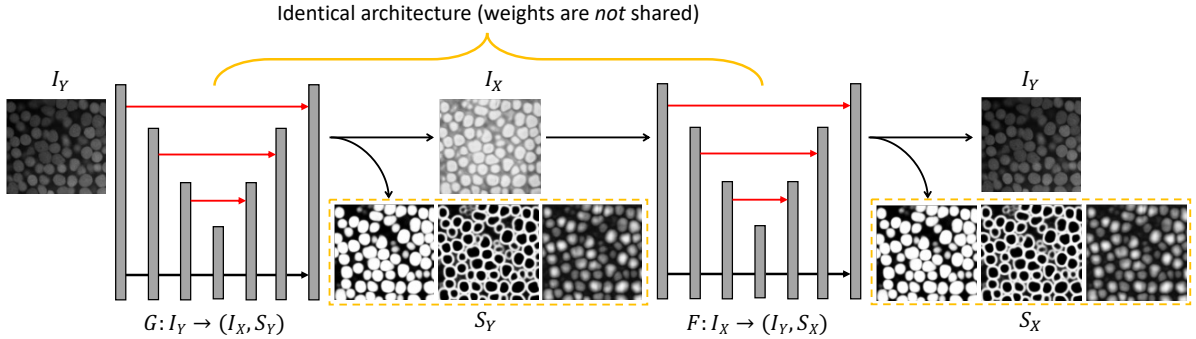


Figure 4.1: Architecture details corresponding to the unsupervised direction of CySGAN's transformation and segmentation cycle ($Y \rightarrow X \rightarrow Y$). Given an image sampled from I_Y, the generator G predicts both the transferred image in I_X and the BCD segmentation representations S_Y. Then the generator F takes only the translated image as input and predicts both the reconstructed image and segmentation representations. The two generators have exactly the same architecture, but the weights are *not* shared as they are optimized to translate images in different domains. Only the generator G is needed to segment I_Y images at inference time (the output channel for translation can also be removed).

4.1. The CySGAN Framework

We have two datasets from different domains: a source dataset $X = (I_X, S_X)$, I_X denoting the image and S_X the corresponding segmentation labels, and a target dataset $Y = (I_Y, \_)$, I_Y denoting the image and _ the absence of segmentation labels. Our *cyclic segmentation* GAN (CySGAN) generates the instance segmentation's S_Y for the unlabeled domain via a domain adaptive segmentation. It does so by using two generators - one for each domain - that simultaneously output intermediate instance representations for the segmentation and translated images (Fig. 1.2c).

$$F : I_X \rightarrow (I_Y, S_X) \qquad G : I_Y \rightarrow (I_X, S_Y) \tag{4.1}$$

Equation 4.1 outlines the in- and output of our generators. The generator F outputs $[\hat{y}_i, \hat{x}_s] = F(x_i)$, where $x_i \sim I_X$ is an image from the source domain, $\hat{y}_i$ is the synthesized image in the target domain, and $\hat{x}_s$ is the concatenation of the intermediate instance representations. The output $[\hat{y}_i, \hat{x}_s]$ is the concatenation along the channel dimension of the synthesized image and the intermediate instance representations. For a clear distinction and later reference we also denote $\hat{y}_i = F(x_i)_{[\mathrm{I}]}$, the image transformation, and $\hat{x}_s = F(x_i)_{[\mathrm{S}]}$, the generated instance representation. CySGAN does not directly output the instance segmentation. Instead it follows the U3D-BCD (see Sec.2.4.1 for U3D-BCD) implementation and generates the three instance representations *binary foreground mask* (B), *instance contour map* (C), and *signed distance transform* (D). We later derive the instance segmentation masks from these instance representations using a marker-driven watershed algorithm (MW) (see Sec. A.1.1 for the watershed algorithm). Accordingly, each generator output has four channels - one channel for the transferred image and three channels for the corresponding instance representations. Therefore, and in contrast to the classic CycleGAN [14] implementation, $G(F(x_i))$ and $F(G(y_i))$ are not a valid expressions for CySGAN. CySGAN's generators only take in a single image while outputting the concatenation of the translated image and instance representations. Figure 4.1 depicts the unsupervised direction of CySGAN's transformation and segmentation cycle ($Y \to X \to Y$).

The two generators F and G are multi-task networks as they simultaneously transfer an image and predict the corresponding instance representation maps. Thus, the single generator, F or G, internally shares almost all its weights between the transformation and the segmentation task, apart from single task-specific output layers. Importantly F and G do not share weights with each other. Our multi-task approach is a clear distinction from most existing sequential systems (see Sec. A.3.2 for weight sharing). A sequential framework treats the domain adaptive segmentation task as two separate subtask. The first subtask is to transfer the images from the source domain to the target domain, $I_{Y'} = F(I_X)$. The objective of the first subtask is to make the transferred images indistinguishable from the distribution of the source domain I_Y while maintaining the structural information recorded in S_X. Subsequently, the second subtask is the supervised segmentation of I_Y. The segmentation model is trained on $(I_{Y'}, S_X)$. At inference time, the trained model can then be used to predict S_Y from I_Y (Fig. 1.2b).

CySGAN's multi-task approach has several advantages over the sequential approach:

- **Joint objective**: The transfer components are aware of the fundamental objective and do not propagate unintentional errors to the segmentation components.
- **Structural Constraints**: Due to the weight sharing, the segmentation and transformation build on the same extracted high-level features. The segmentation component thereby regulates the generation process. It explicitly imposes powerful structural constraints, promoting label transferability and thus suppressing hallucination.
- **Lightweight during Training**: The described weight sharing between the segmentation and transformation model significantly decreases the number of parameters compared to a sequential model.

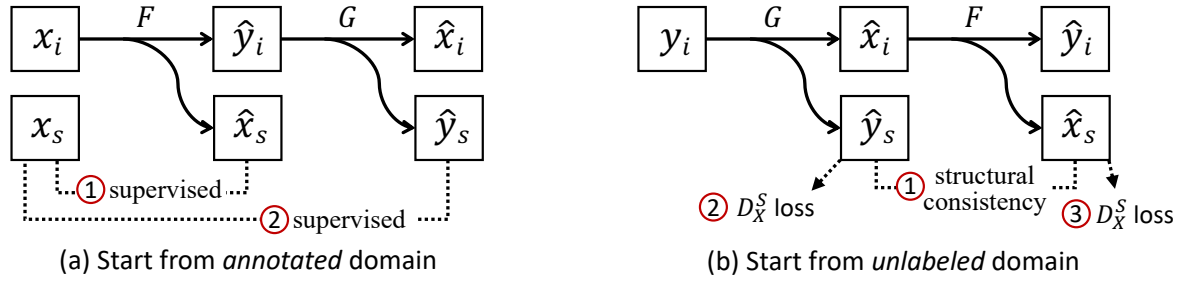


Figure 4.2: Different segmentation losses for two domains. **(a)** For an annotated image in X, we compute the supervised losses of predicted segmentation representations against the label. **(b)** For an unlabeled image in Y, we enforce *structural consistency* between predicted representations (as the underlying structures should be shared) and also segmentation-based adversarial losses to improve the quality of predictions in the absence of paired labels.

- **Lightweight during Deployment**: During inferencing Y, we only require the generator G. Furthermore, we do not require the output layer for the image transformation during inferencing. We can, therefore, remove the layer at deployment, reducing the system complexity of our model to that of a simple vanilla U3D-BCD [6].

The following sections explain the image translation and segmentation objectives that we use to optimize CySGAN. Unlike standard unsupervised image translation, our two image domains are *asymmetric* (X is labeled, Y is unlabeled). Accordingly, we need to distinguish between the domains. Therefore, we apply similar image translation losses but unique segmentation losses to the translation components of the two domains X and Y. The final section of this chapter will outline our implementation, including the architecture and data augmentation schema.

4.2. Image Translation Losses

We can split CySGAN's objectives into image translation and image segmentation. The image translation focuses on domain adaptation. Since the datasets I_X and I_Y are not paired, we have to rely on an adversarial loss for the distribution matching of real and synthesized images. Further, as the domains are *asymmetric* we denote F as the *forward* generator and G as the *backward* generator (Eqn. 4.1). F transfers ($X \to Y \to X$) while G transfers ($Y \to X \to Y$). We formulate the adversarial loss for F as follows:

$$\mathcal{L}_{GAN}(F, D_Y^I) = \log D_Y^I(y_i) + \log(1 - D_Y^I(\hat{y}_i)) \tag{4.2}$$

D_Y^I is the discriminator for the I_Y domain. The discriminator distinguishes between real y_i and synthesized $\hat{y}_i$ images. The synthesized image is the output of the forward generator $\hat{y}_i = F(x_i)_{[\mathrm{I}]}$ corresponding to the real image. The adversarial loss for the image transformation of the backward generator G is symmetrical to the adversarial loss of the forward generator F:

$$\mathcal{L}_{GAN}(G, D_X^I) = \log D_X^I(x_i) + \log(1 - D_X^I(\hat{x}_i)) \tag{4.3}$$

D_X^I is the discriminator for the I_X domain. The discriminator distinguishes between real x_i and synthesized $\hat{x}_i$ images. Following CycleGAN [14] we also apply the *cycle-consistency* loss for both domains' real and reconstructed images.

$$\mathcal{L}_{cyc}(F, G) = \|G(\hat{y}_i)_{[\mathrm{I}]} - x_i\|_1 + \|F(\hat{x}_i)_{[\mathrm{I}]} - y_i\|_1 \tag{4.4}$$

The *cycle-consistency* loss requires our model to learn to reconstruct the real images from the corresponding synthesized images, conditioning the model to incorporate sample-specific information into the transformation. We apply the loss in both directions. Since the original binary cross-entropy GAN loss (Eqn. 4.2) was proven to be unstable, we instead follow the official CycleGAN implementation and optimize the LSGAN [51] loss:

$$\mathcal{L}_{LSGAN}(F, D_Y^I) = \left(D_Y^I(y_i) - 1\right)^2 + \left(D_Y^I(\hat{y}_i) + 1\right)^2 \tag{4.5}$$

LSGAN [51] prevents vanishing gradient and smooths the training process. Again the loss for G is symmetrical:

$$\mathcal{L}_{LSGAN}(G, D_X^I) = \left(D_X^I(x_i) - 1\right)^2 + \left(D_X^I(\hat{x}_i) + 1\right)^2 \tag{4.6}$$

The image translation and image segmentation share the same backbone. However, the output layers are logically separated. Accordingly, the image translation losses update the shared backbone but do not influence the output layers for the segmentation maps and vice versa.

4.3. Instance Segmentation Losses

We build upon two U3D-BCD's [6] to generate the instance segmentation masks. U3D-BCD [6] follows a typical instance segmentation approach for microscopy [31, 33, 30, 6] by first generating instance representations from which the segmentation masks are later derived using a decoding algorithm. Specifically, U3D-BCD [6] generates the *binary foreground mask* (B), *instance contour map* (C), and *signed distance transform* (D) as three output channels using a 3D U-Net [16]. We optimize channels B and C using the BCE (see Sec. A.4.5 for a definition of BCE), while D is regressed using the MSE (see Sec. A.4.4 for a definition of MSE). We use BCE for channels B and C as the binary background mask and contour mask are effectively pixel classifications between background and foreground or contour and not contour. Unlike B and C, we regress the channel D using the MSE loss, as the distance map consists of continuous numerical values.

4.3.1. Labeled Source Domain

We can only apply the supervised instance representation losses to the supervised translation direction ($X \rightarrow Y \rightarrow X$) and not to the unsupervised direction ($Y \rightarrow X \rightarrow Y$). For the

supervised direction, the joint instance representation loss of the forward generator F, given an image-label pair (x_i, x_s) sampled from (I_X, S_X), is:

$$\begin{aligned}\mathcal{L}_{seg}(F) = \mathcal{L}_{bce}\left(F(x_i)^B_{[S]}, x_s^B\right) + \mathcal{L}_{bce}\left(F(x_i)^C_{[S]}, x_s^C\right) \\ + \|F(x_i)^D_{[S]} - x_s^D\|_2^2\end{aligned} \tag{4.7}$$

with $x_s = [x_s^B, x_s^C, x_s^D]$ the channel wise concatenation of three instance representations. The corresponding joint instance representation loss for the supervised direction of the backward generator G, given an image-label pair $(\hat{y}_i, x_s)$ sampled from $(F(I_X)_{[I]}, S_X)$, is:

$$\begin{aligned}\mathcal{L}_{seg}(G) = \mathcal{L}_{bce}\left(G(\hat{y}_i)^B_{[S]}, x_s^B\right) + \mathcal{L}_{bce}\left(G(\hat{y}_i)^C_{[S]}, x_s^C\right) \\ + \|G(\hat{y}_i)^D_{[S]} - x_s^D\|_2^2\end{aligned} \tag{4.8}$$

with $\hat{y}_s = [\hat{y}_s^B, \hat{y}_s^C, \hat{y}_s^D]$ the channel wise concatenation of three instance representations. Accordingly the segmentation loss $\mathcal{L}_{seg}(F)$ for the forward generator and the segmentation loss $\mathcal{L}_{seg}(G)$ (based on synthesized $\hat{y}_i$) for the backward generator are optimized by directly comparing $\hat{x}_s$ and $\hat{y}_s$ to x_s, with $x_s \sim S_X$ (① and ② in Fig. 4.2a). The beauty of this setup is that we do not only train the forward generator F in a supervised manner but also the backward generator G. We can do so because the underlying structure of an image should not change throughout the domain transfer, and the ground truth maintains its relevance for the reconstruction, i.e., given an image-label pair (x_i, x_s) sampled from (I_X, S_X):

$$F(x_i)_{[S]} \approx x_s \approx G(F(x_i)_{[I]})_{[S]} \tag{4.9}$$

Should we want to use CySGAN with a different set of instance representations, we simply need to provide the appropriate ground truth representations during training [1]. For a different number of instance representations, we need to adjust the number of instance representation heads of the generators and the input channels of the discriminators.

In the following section, we describe the losses that we applied to the unsupervised direction of the cycle ($Y \to X \to Y$).

4.3.2. Unlabeled Target Domain

For the unsupervised direction of the cycle ($Y \to X \to Y$), we rely on structural consistency and structure-based adversarial losses while counting on the supervised cycle to mediate the general segmentation task. We observe that the segmentation results of both generators, $\hat{y}_s$ and $\hat{x}_s$ (① Fig. 4.2b,) should be the same, as the underlying structure of a sample is consistent throughout the cycle. Accordingly we introduce the *structural consistency* loss $\mathcal{L}_{sc}(F, B)$:

$$\mathcal{L}_{sc}(F, G) = \|G(y_i)_{[S]} - F(G(y_i)_{[I]})_{[S]}\|_1 \tag{4.10}$$

[1] For example, SUSAN [18] apply the supervised segmentation losses for 2D semantic masks with pixel-wise class annotations.

Furthermore, we note that the segmentation masks of both domains should have a similar distribution as they both mask nuclei. Accordingly, we enforce distributional similarity of the unsupervised generated instance representations $\hat{y}_s \sim G(I_Y)_{[S]}$ and $\hat{x}_s \sim F(G(I_Y)_{[I]})_{[S]}$ with a sample of S_X using structure-based adversarial losses (② and ③ in Fig. 4.2b). We donate the structure-based adversarial losses as $\mathcal{L}_{LSGAN}(G, D_X^S)$:

$$\mathcal{L}_{LSGAN}(G, D_X^S) = \Big(D_X^S(x_s) - 1\Big)^2 + \Big(D_X^S(\hat{y}_S) + 1\Big)^2 \tag{4.11}$$

and $\mathcal{L}_{LSGAN}(F, D_X^S)$:

$$\mathcal{L}_{LSGAN}(F, D_Y^S) = \Big(D_Y^S(x_s) - 1\Big)^2 + \Big(D_Y^S(\hat{x}_s) + 1\Big)^2 \tag{4.12}$$

The discriminators D_Y^S and D_X^S take the concatenation of all three instance representations as input and process them in a single pass. Processing the instance representations together emphasizes their correlation and reduces the system's complexity by eliminating the need for three independent discriminators. The architecture of D_X^S is almost identical to the image discriminators except for the number of input channels. Importantly, these losses require a resolution match between both domains, i.e., the dimensions of the instances in both datasets must agree.

We follow semi-supervised paradigms by using unpaired and unlabeled images in CySGAN's optimization schema (see Sec. A.3.4). In the absence of paired labels for I_Y, our presented structural consistency and segmentation-based adversarial losses stabilize and structure the image generation and instance segmentation. The proposed structural consistency loss strengthens the backward generator G by coupling its segmentation capabilities more strongly with those of the forward generator F. The forward generator F is generally more robust and precise because it benefits more directly from the supervised part of the cycle than G. Accordingly, our structural consistency loss stabilizes the segmentation generation and suppresses hallucinations or unintended omissions. We use our structure-based adversarial loss for output space adaptation. The loss helps to converge the label distributions of the source and the target domain [56]. The idea is that although individual images are very different, the associated masks are fundamentally similar in structure and share many common features, such as spatial layout and local context. The loss balances the *structural consistency* losses by ensuring appearance vise consistency.

CySGAN's unified translation segmentation framework is suitable for various semi-supervised learning objectives incorporating unlabeled and unpaired images in the optimization. For example, one can also introduce augmentation consistency [60] to the input of the unlabeled domain. Our proposed structural consistency loss emphasizes the concept of using unlabeled images in a unified translation segmentation framework.

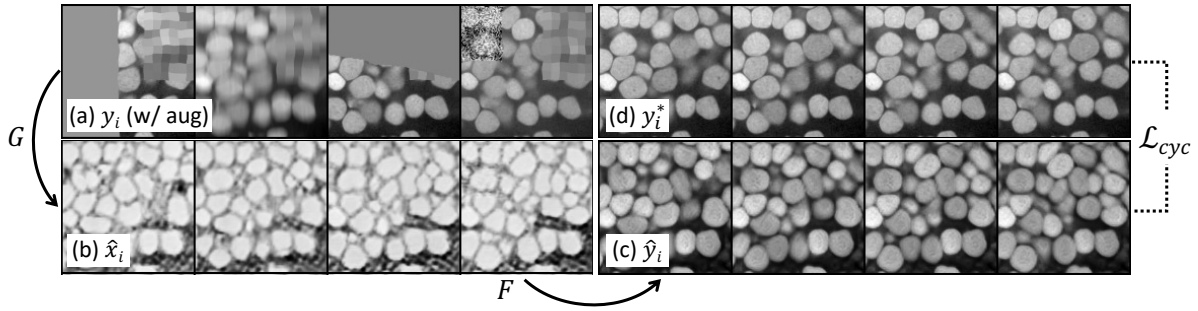


Figure 4.3: Restore augmented regions with an adapted cycle-consistency strategy. We show four consecutive slices of **(a)** augmented real I_Y input, **(b)** synthesized I_X volume, **(c)** reconstructed I_Y volume and **(d)** real I_Y volume w/o augmentations. By forcing the cycle consistency of (c) to (d), the model learns to restore corrupted regions using the 3D context.

4.4. Implementation

4.4.1. Full Objective

The full objective ($\mathcal{L}$) of CySGAN is the sum of losses in Sec. 4.2 and 4.3 for the whole cycle, which is

$$\begin{aligned}\mathcal{L} = &\underbrace{\mathcal{L}_{GAN}(F, D_Y^I) + \mathcal{L}_{GAN}(G, D_X^I) + \mathcal{L}_{cyc}(F, G)}_{\text{image-to-image translation}} \\ &+ \underbrace{\mathcal{L}_{seg}(F) + \mathcal{L}_{seg}(G)}_{\text{supervised segmentation}} \\ &+ \underbrace{\mathcal{L}_{sc}(F, G) + \mathcal{L}_{GAN}(G, D_X^S) + \mathcal{L}_{GAN}(F, D_X^S)}_{\text{semi-supervised segmentation}}\end{aligned} \tag{4.13}$$

For this thesis, we assign a uniform weight to all losses. We included the the effect of our semi-supervised segmentation loss in our ablation study (see Seg. 6.2) by testing CySGAN with and without it.

4.4.2. Augmentation-Aware Cycle Consistency

U3D-BCD [6] uses several training enhancements such as random missing, blurring, and noisy regions (Fig. 4.3a). We preserve them in CySGAN to achieve a better segmentation quality. Existing domain adaptation approaches generally do not apply augmentations [45, 55], only apply affine geometric transformations such as reflections, rotations, and scaling [10], and in single cases may apply contrast adaptations [54]. However, the applied augmentation strategies only serve to grow the training dataset. We are unaware of any domain adaptation method that applies occlusions or blurs. This is not surprising since a discriminator can easily distinguish between synthetic and real images when the former contains visible image adjustments. In the original CycleGAN framework, such augmentations would significantly

upset the balance of the min-max game between the GANs' simultaneously trained generators and discriminators.

To overcome this problem, we propose to use augmented training images as input but apply all the losses between the model's output and the non-augmented version of the input. We thus update, e.g., the cyclic consistency (Gln. 4.4) loss (in the case that we input augmented samples) to:

$$\mathcal{L}_{cyc}(G, F) = \|G(\hat{y}_i)_{[\mathrm{I}]} - x_i^*\|_1 + \|F(\hat{x}_i)_{[\mathrm{I}]} - y_i^*\|_1 \tag{4.14}$$

with augmented samples y_i and x_i, there synthesised versions in the opposing domain $\hat{x}_i$ and $\hat{y}_i$, and their reconstructed version in the original domain $G(\hat{y}_i)_{[\mathrm{I}]}$ and $F(\hat{x}_i)_{[\mathrm{I}]}$. Note that while $\hat{y}_i$ and $\hat{x}_i$ is the synthesised augmented version of y_i, x_i we do not calculate the reconstruction $\mathcal{L}_{cyc}(F, G)$ of $\hat{y}_i$ to y_i and $\hat{x}_i$ to x_i. Instead we enforce the similarity to the clean versions y_i^* and x_i^* (Fig. 4.3d). Accordingly, $\hat{y}_i$ and $\hat{x}_i$, although coming from the augmented input y_i and x_i, should no longer contain augmentations. The augmentation schema is visualized in Fig. 4.3, depicting samples from the unsupervised cycle ($Y \to X \to Y$). G transfers augmented y_i to $\hat{x}_i$ and F reconstructs $\hat{x}_i$ to $\hat{y}_i$. We then apply the cycle consistency loss to the reconstructed image $\hat{y}_i$ and the original non-augmented image y_i^*

The main benefit of using our augmentation-aware cycle consistency strategy is that both generators learn to restore corrupted regions using 3D context. In our augmentation scheme, we mask each layer independently. However, we do not mask consecutive layers, as we want to preserve neighborhoods from which the model can recover the information. We show in the ablation studies that this strategy significantly affects the region-adaptive segmentation performance.

4.4.3. Network Details and Optimization

CySGAN uses two identical 3D U-Nets [16] for the forward generator F and the backward generator G. Importantly, both generators have the same architecture but do not share weights. The 3D U-Nets have a single input channel for the image and four output channels, one for the translated version of the image and three for the instance representations BCD. The architecture of CySGANs U3D-BCD generators is depicted in figure 4.4.

As shown in the architectural layout, we use strided convolutions for downsampling and trilinear interpolation for upsampling. The outer building blocks - excepting the input and output layer - of the encoding and decoding path use a kernel size of $(1, 3, 3)$, a stride of $(1, 2, 2)$, and padding of $(0, 1, 1)$. Therefore, their convolutional layers are effectively 2D. We choose this configuration because of the strong anisotropy of our data. The inner two building blocks of the encoding and the decoding path use a kernel size of $(3, 3, 3)$, a stride of $(2, 2, 2)$, and padding of $(1, 1, 1)$. The isotropic configuration of the inner layers reflects the effect of the increasing receptive field of the successive convolutional layers, which balances the resolution divergence. Except for the output layer, we do not use the bias terms in our convolutions since the consecutive batch normalizations would eradicate the effect. We based our discriminators on a similar convolutional 2D/3D hybrid configuration as our generators as depicted by the architectural layout in figure 4.5. The image discriminators D_X and D_Y

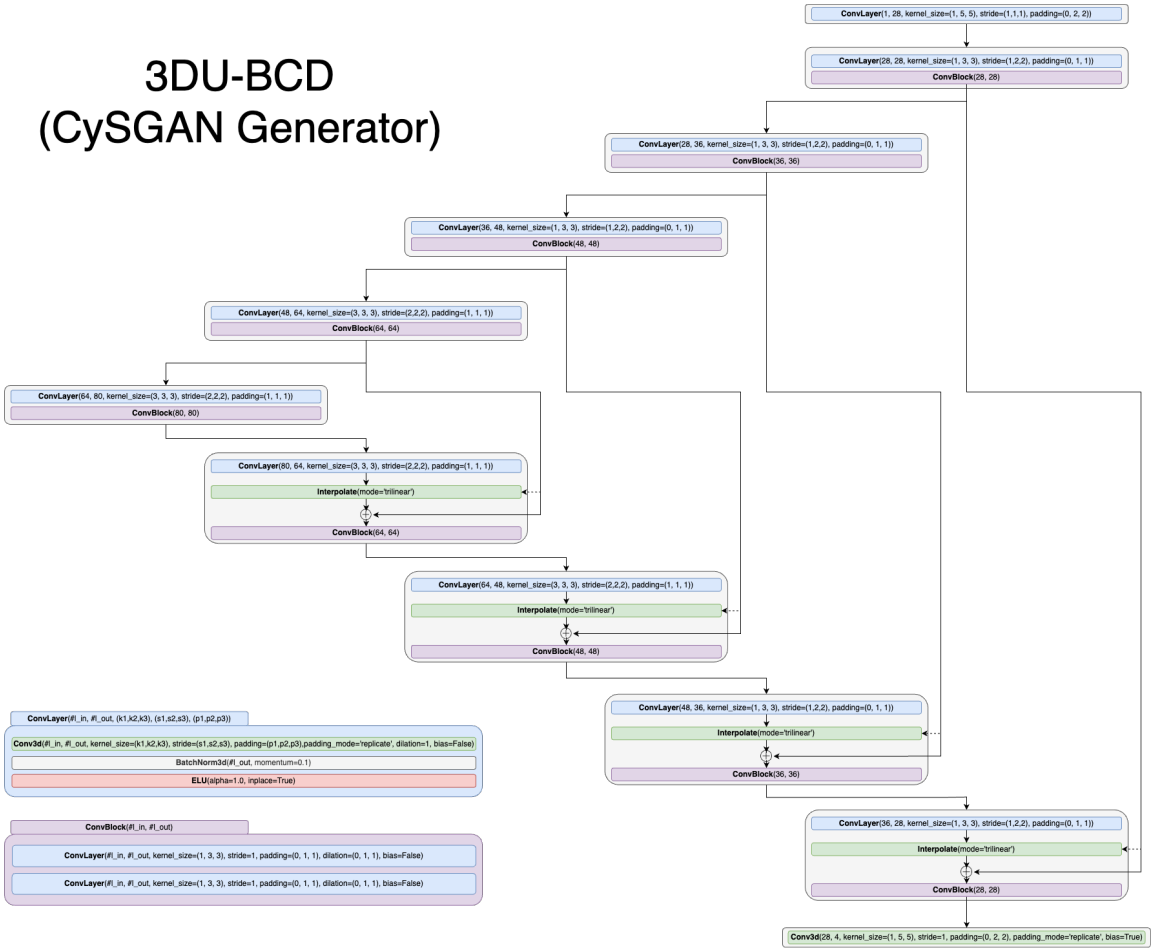


Figure 4.4: Architecture layout of CySGAN's 3DU-BCD generators. The layer specification correspond to common PyTorch layer names. The legend in the bottom left depicts the two general building blocks "ConvLayer" and "ConvBlock".

have a single input channel for the grayscale images. The segmentation discriminators D_S have three input channels for the BCD representations. Each discriminator has five layers. The first four layers are each a stack of a 3D strided convolution, a 3D instance norm, and a leaky Relu as non-linear activation. The fifth layer is a single 3D convolution that outputs a single-channel feature map representing the *realness* of corresponding input patches. The first four layers do not use biases due to the use of instance norm, while the fifth one does use biases. The layers have 64, 64, 128, 128, and 256 filters. For the first three layers the kernels are of shape $(1, 5, 5)$, with a stride of $(1, 2, 2)$, and padding of $(0, 2, 2)$, making the 3D convolutions effectively 2D. The fourth 3D convolution uses an isotropic kernel of shape $(3, 3, 3)$, with a stride of $(2, 2, 2)$, and padding of $(1, 1, 1)$. The final 3D convolutional layers uses an isotropic kernel of shape $(3, 3, 3)$, with a stride of $(1, 1, 1)$, and padding of $(1, 1, 1)$. The single-channel feature map outputted by our discriminators and representing the *realness* of corresponding

input patches is based on PatchGAN [44]. PatchGAN assesses the generator's performance on the scale of local image patches to evaluate high-frequency components better and prohibit blurring (see Sec. A.2.1 for a more detailed explanation of PatchGAN). This approach assumes sufficient independence between pixels separated by more than a patch diameter [44]. We optimize the framework using the LSGAN objective (Eqn. 4.5) as it was proven to stabilize the training, as opposed to the BCE GAN loss (Eqn. 4.2). One minor but important detail is that we detach the synthesized images when computing the segmentation losses. We do this for two reasons. First, we want to prevent updating the first generator twice in one cycle. Second, and more importantly, we want to avoid the segmentation objective affecting the image translation results. The first generator in the cycle outputs the synthesized image through its transfer layer. The second generator takes the synthesized image, reconstructs the original, and generates the instance representations. Suppose we do not detach the synthesized image before generating and computing the losses for the instance representations. In that case, we will pass the segmentation-based losses through the first generator's image transfer layer. We trained CySGAN for 10^6 iterations. For the optimizer, we used AdamW [27], with an initial learning rate of 2×10^{-3} (decreased with cosine annealing) and a batch size of 8 using 4 NVIDIA V100 GPUs.

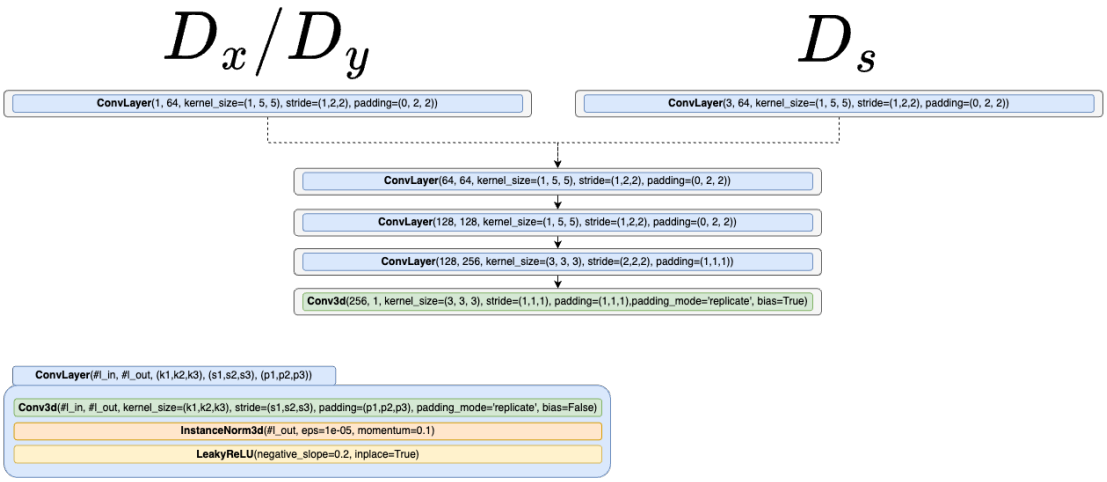
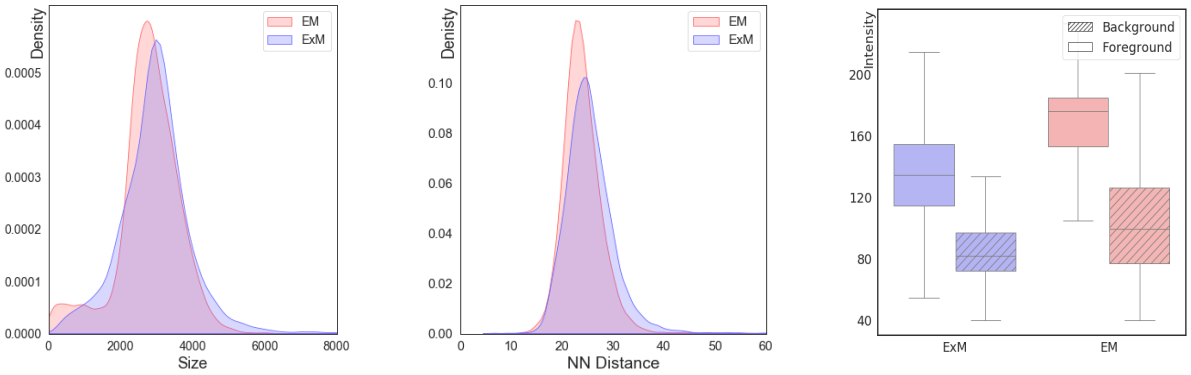


Figure 4.5: Architecture layout of CySGAN's discriminators. D_y and D_x are the discriminators for the domain-translated images of the Y and the X domains. D_S is the discriminator for the domain-invariant instance representation maps. The layer specifications correspond to common PyTorch layer names. The legend in the bottom left depicts the general building block "ConvLayer".

5. Datasets

Existing work in unsupervised domain-adaptive segmentation is predominantly 2D, performs semantic segmentation, and focuses on a small selection of image modalities. This heterogeneity manifests in a severe lack of datasets for 3D instance segmentation. In particular, matching datasets mapping similar content with different modalities are hard to come by. Therefore, as part of this thesis and the companion paper, we release a fully annotated dataset with dense 3D neuronal nuclei instances (Fig. 5.2) that matches the public available NucMM-Z EM volume from the NucMM dataset [6].



(a) Instance Size (b) Nearest-Neighbor Distance (c) Voxel Intensity

Figure 5.1: Statistics of the source (EM) and target (ExM) datasets. We show (**a**) the distribution of instance size (in terms of voxels) and (**b**) the distance between adjacent nuclei centers. The density plots are normalized by the total number of instances in each volume. We also show (**c**) the voxel intensity distribution in object (foreground) and non-object (background) regions for both volumes. The domain gap is characterized by different intensity distributions and contrast.

5.1. NucExM Dataset (Target)

NucExM comprises two volumes that image part of a day seven post-fertilization (dpf) zebrafish brain. We recorded both volumes using ExM with confocal microscopy as the imaging method. A confocal microscope is a light microscope that scans a probe instead of imaging the entire probe at once. Scanning the probe allows the microscope to use a highly focused beam of light that increases optical resolution and contrast. The two NucExM volumes

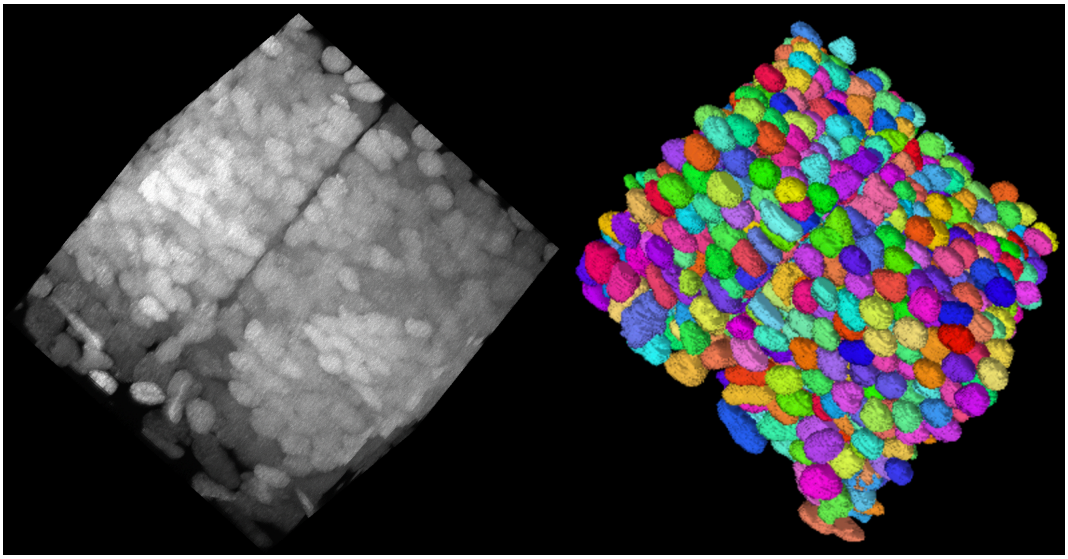


Figure 5.2: Visualization of the NucExM dataset. We sample a sub-volume of size $(1024, 1024, 100)$ from the V_1 volume of NucExM. (Left) The expansion microscopy (ExM) image volume visualized using *Napari*. (Right) The corresponding 3D segmentation masks visualized using *Neuroglancer*.

have an anisotropic resolution of $0.325 \times 0.325 \times 2.5\ \mu m$ in (x, y, z) order. We achieved an approximate tissue expansion factor of 7.0 using ExM. Thus the effective resolution of the data is $0.046 \times 0.046 \times 0.357\ \mu m$. Both volumes span $2048 \times 2048 \times 255$ voxels that together contain 18.4K nuclei (9.6K + 8.8K). For training and inferencing, we downsample both volumes by a factor of four along the x and the y axes to $512 \times 512 \times 255$ to save computational cost. Table 5.1 summarizes the dataset's metadata.

5.1.1. Source Dataset

As our source dataset, we use the publicly available EM volume NucMM-Z from the NucMM dataset [6]. NucMM-Z images nearly an entire brain of an adult zebrafish using EM and, therefore, presents an excellent match to our NucExM dataset. The resolution of the dataset is $0.48 \times 0.48 \times 0.48 \mu m$. We handled the severe resolution mismatch between NucExM and NucEM by cropping a $200 \times 200 \times 255$ subvolume from NucMM-Z and upsampling it to $512 \times 512 \times 255$. The cropped NucMM-Z subvolume contains 12K neuronal nuclei instances. We upsampled the corresponding instance masks using nearest-neighbor upsampling followed by Gaussian filtering to smooth the boundaries and thresholding to recover the binary representation.

5.1.2. Datasets Comparison

We use a cropped portion of the NucMM-Z as our source dataset and both NucExM volumes as our target dataset. Both datasets image the nuclei in the zebrafish brain. The source dataset was acquired using EM, while we imaged the target dataset applying ExM and using a confocal microscope.

Figure 5.1 shows a qualitive comparision of the source and the target dataset regarding, instance size (Fig. 5.1a), nearest-neighbor distance between nuclei centers (Fig. 5.1b), and intensity and contrast difference of object and non-object voxels (Fig. 5.1c). We upsampled the source dataset by a factor of 2.56 along the x and the y axes. We downsampled the target dataset by a factor of four along the x and the y axes. After scaling, the densities of instance size (Fig. 5.1a) and nearest-neighbor distance between the nuclei centers (Fig. 5.1b) of both datasets are nearly aligned. The alignment helps CySGAN segment the 3D neuronal nuclei instances in a domain-adaptive setting, as the model can focus on appearance and general structure without having to consider scaling. Thus, the domain gap is characterized by the different intensity and contrast of object and non-object voxels (Fig. 5.1c). Although the gap appears minor, we show in experiments (Sec.6) that we can not resolve the difference by using conventional appearance-level matching approaches such as histogram matching.

Sample	#Volumes	Volume Size (each)	Resolution (μm)	Ex. Ratio	#Instances
Zebrafish Brain	2	2048×2048×255	0.325×0.325×2.5	7.0	9.6K+8.8K

Table 5.1: **NucExM dataset metadata.** We curated and densely annotated a *neuronal nuclei* segmentation dataset with two ExM volumes of zebrafish. The tissue was expanded by about 7× to increase resolution.

5.1.3. Evaluation Metric

To evaluate the performance of our method, we choose to use average precision (AP) as the evaluation metric as it is common practice in instance segmentation [61, 62]. Specifically, we choose AP with an IoU threshold of 0.5 (AP-50) for our 3D volume data, i.e., an instances segmentation is evaluated as correct in case its IoU with the ground truth segmentation is at least 0.5. We rely on Wei's *et al.* [30] existing public implementation with improved efficiency for 3D volumes that first computes the 3D bounding boxes for all the instance segmentations before calculating the AP.

6. Experiments

6.1. Methods in Comparison

To put CySGAN's performance in perspective, we compare it to several state-of-the-art frameworks. Like CySGAN, those frameworks have to segment the target data without seeing any domain-specific labels during training or testing. We categorize the selected models into three groups: generalist models, appearance-level adaptation models, and feature-level adaptation models.

6.1.1. Generalist models

Generalist models train on diverse and extensive training datasets that cover different image modalities, resolutions, and contrasts. We chose Cellpose [31] (see Sec. 2.4.1 for Cellpose) and StarDist [33] (see Sec. 2.4.1 for StarDist) for comparison based on their performance and the distribution of their training data. For example, Stringer et al. trained Cellpose on a dataset with over 70k segmented objects, primarily nuclei [31]. We used the official implementation for both models, as referenced in their papers.

6.1.2. Appearance-level adaptation

Appearance-level adaptation methods (otherwise also called image-level adaptation) work at the pixel-level, decreasing the domain gap by aligning the source with the target distribution or vice versa. As a first step, such methods generally transfer target images to the source domain. In a second step, they then use either an adversarial loss to evaluate the distribution match or an L1-like loss for a direct comparison (see Sec. A.4.1 for L1 and MAE loss). The direct comparison is, e.g., possible when using image pairs or applying cycle consistency. However, existing methods are almost exclusively 2D and developed for semantic segmentation [45,

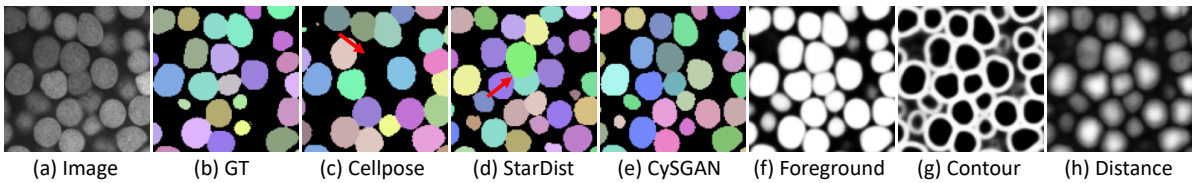


Figure 6.1: Visual comparisons of segmentation results. **(a)** ExM image, **(b)** ground-truth instances, **(c)** Cellpose [31], **(d)** StarDist [33] and **(e)** CySGAN results. We also show **(f-h)** predicted segmentation representations of U3D-BCD used in CySGAN.

63, 55]. Accordingly, we had to adapt existing frameworks to our adversarial 3D instance segmentation task. We implemented two sequential setups. The first uses histogram matching for the appearance-level adaptation (see appendix Sec. A.1.2 for histogram matching). The second uses CycleGAN [14] (see Sec. 3.1 for a review of CycleGAN) for the appearance-level adaptation. In both cases we used U3D-BCD [6] (see background Sec. 2.4.1 for a summary of U3D-BCD) for segmentation. We tested both transfer directions for completeness. For the first direction, $I_X \rightarrow I_Y$ we transferred I_X to $I_{Y'}$, using the synthesized images to train a segmentation model in the target domain. For the second direction, $I_Y \rightarrow I_X$ we transferred I_Y to $I_{X'}$, training a segmentation model in the source domain. Generally, we prefer the first direction as the second requires that we transfer the target data to the source domain during testing. In contrast, the first direction does not use the transfer model during testing.

6.1.3. Feature-level adaptation

Compared to the pixel-level losses used in appearance-level adaptation, feature-level adaptation models map the source and target distributions in their embedding space. Feature-level adaptation methods that operate on the embedding space build on the observation that spacial features are domain invariant. Accordingly, the embeddings of images from different domains that depict similar objects should align if generated with the same domain invariant encoder [10]. Again, we had problems finding a model for 3D instance segmentation and therefore had to implement our own.

We base our implementation on the same high-level ideas as the implementation of the feature-level adaptation model proposed by Tsai *et al.* [56]. Tsai *et al.* use a single segmentation model that predicts the segmentation for the source and the target images. We train the segmentation model using a supervised cross-entropy loss for the source images, an appearance-level adversarial loss, and a feature-level adversarial loss. The appearance-level adversarial loss matches the distribution of source and target images, i.e., it tries to determine the origin of a generated segmentation. The feature-level adversarial loss aligns the features in the embedding space, i.e., it evaluates whether an embedding originates from a source or a target sample. Our 3D instance segmentation adaptation of the model uses the same U3D-BCD segmentation model as in our CySGAN - without the added image transformation layer. Accordingly, we apply the first adversarial and cross-entropy losses to the BCD instance representation maps. We use the same discriminator for the adversarial losses as in our CySGAN framework. All other training details, like augmentations, are kept the same.

6.2. Results

The NucExM dataset that we published as part of this work comprises two volumes, V_1, and V_2. To evaluate the generalization ability of CySGAN and the models described above, we use V_1 for training while running inferencing on both V_1 and V_2. We are still in the unsupervised setting, so we only use the images of V_1 and not its labels for the ExM domain. Table 6.1 depicts the results of CySGAN and the aforementioned frameworks. From the

Table 6.1: **Benchmark results on the NucExM dataset.** We compare both pretrained segmentation networks and translation-segmentation models using the AP scores. In the two-step approaches, we use U3D-BCD [6] for segmentation. **Bold** and underlined numbers denote the 1st and 2nd results.

Method	Cellpose	StarDist	Feat. DA	Histogram + Segm		CycleGAN + Segm		CySGAN (**Ours**)
				$I_X \to I_Y$	$I_Y \to I_X$	$I_X \to I_Y$	$I_Y \to I_X$	
AP-50 (V_1)	0.644	0.816	0.807	0.774	0.804	<u>0.867</u>	0.772	**0.927**
AP-50 (V_2)	0.765	0.875	0.826	0.795	0.816	<u>0.881</u>	0.777	**0.934**
Average	0.705	0.846	0.817	0.785	0.810	<u>0.874</u>	0.775	**0.931**

Table 6.2: **Ablation study for CySGAN.** The results show obvious performance degradation without using data augmentations, semi-supervised losses, and signed distance map (D), demonstrating the importance of those components for CySGAN. The red number describes the corresponding performance decrease compared to our full CySGAN implementation.

Configuration	w/o Augmentation	w/o Semi-sup Losses	w/ BC only	CySGAN (**Ours**)
AP-50 (V_1)	0.761 (-0.166)	0.878 (-0.049)	0.843 (-0.084)	**0.927**

table, it is apparent that CySGAN is the most successful method. It outperforms the pretrained Cellpose [31] and StarDist [33] models, our feature-level adaptation implementation, and the sequential frameworks that use either histogram matching or CycleGAN for the domain adaptation. The second best-performing framework is the sequential model that uses CycleGAN and U3D-BCD for the $I_X \to I_Y$ transfer direction. This result justifies our choice of CycleGAN as the backbone of CySGAN. CySGAN outperforms CycleGAN+Segm by 5.7%. The results additionally show that for sequential models, the $I_X \to I_Y$ direction generally performs better than the $I_Y \to I_X$ direction.

Figure 6.1 shows visual results for Cellpose, StarDist, and CySGAN. We found that Cellpose often misses more difficult object instances, leading to a higher rate of false negatives. StarDist shows many overlaps and poorly aligned object boundaries. We discovered that StarDist's strong star-convex shape prior often neglects other features such as object boundaries. The three tiles to the right of Fig. 6.1 (f-h) depict the three instance representations used by CySGAN for the generation of the segmentation mask Fig. 6.1e. Figure 6.2 shows example results for the histogram matching. Figure 6.3 shows qualitative results generated by our CySGAN framework.

6.2.1. Ablation Study

To validate and justify CySGAN's design choices, we tested the impact of the augmentation schema (Fig. 4.3), the semi-supervised segmentation losses (Eq. 4.13), and the learned instance representation maps on the segmentation of NucExM's V_1 volume. The results of the ablation

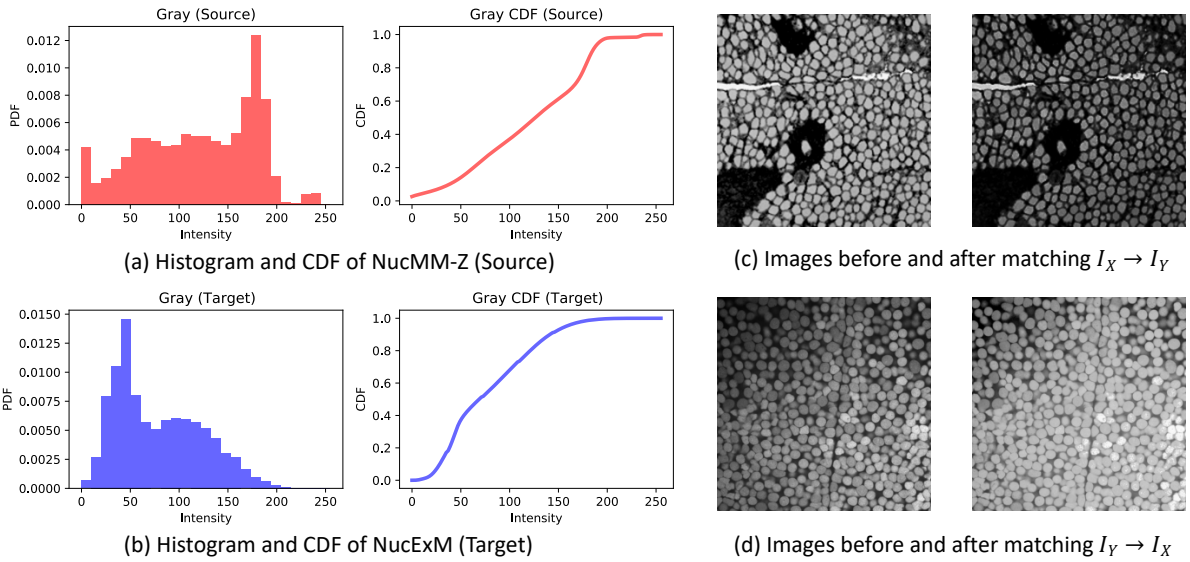


Figure 6.2: Histogram matching between EM and ExM images. We show the histograms and cumulative distribution functions (CDFs) of **(a)** electron microscopy (EM) and **(b)** expansion microscopy (ExM) images. The effect of histogram matching is shown in **(c)** and **(d)** for both matching directions.

study are depicted in Table 6.2. The ablation study shows that the augmentation schema has the most striking impact on CySGAN's performance with a precision increase of 16.6%. We observe that the model is prone to mode collapse without the augmentations, i.e., the output does not reflect the diversity of the input. Additionally, a qualitative analysis indicated a rise in false positives. Accordingly, our augmentation schema does improve not only the performance but also the robustness of the model. Further, the ablation study shows that the semi-supervised segmentation losses improve CySGAN's performance by 4.9%. Without the semi-supervised segmentation losses, the framework is close to a 3D implementation of SUSAN [18] (plus the augmentation schema). Further, we observe that the performance is similar to that of the sequential model that uses CycleGAN for translation and a U3D-BCD network for segmentation. Finally, we tested using U3D-BC instead of U3D-BCD as backbones for CySGAN. U3D-BC only generates the background (B) and contour (C) map, while U3D-BCD additionally generates the distance (D) map. Incorporating D improves CySGAN's performance by 8.4%. This result underlines the impact of the distance-map generation on the instance segmentation of data with high object density. The results of our ablation study support our design decision for CySGAN and allow us to quantify their impact.

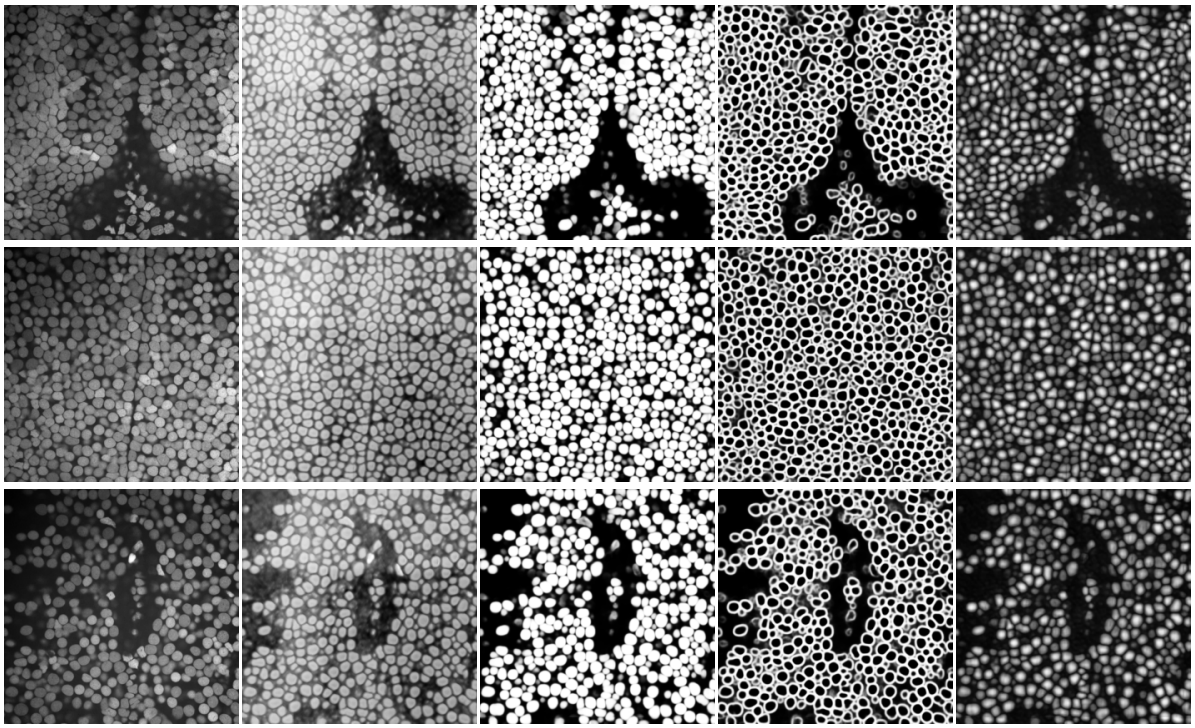


Figure 6.3: Qualitative results of CySGAN. We show multiple slices of **(a)** input NucExM images, as well as **(b)** transferred images, **(c)** predicted binary foreground masks (B), **(d)** predicted instance contour maps (C) and **(e)** predicted distance transform maps (D) of our proposed CySGAN model.

7. Conclusion

In this thesis, we present CySGAN, a unified domain-adaptive segmentation model for segmenting the unlabeled ExM image modality. We optimize CySGAN with image translation losses as well as supervised and semi-supervised instance segmentation losses. Our framework outperforms sequential models that conduct translation and segmentation using separate networks and generalist models that train on diverse and extensive training datasets. In addition, CySGAN represents a significant simplification compared to sequential models, as it unifies image translation and instance segmentation using weight sharing. We also publicly release the NucExM dataset as a testbed for future domain-adaptive 3D instance segmentation models. We discuss our results and outline possible paths for feature work in the following.

7.1. Discussion

We aim to train a segmentation model for the ExM domain. Consequently, we are interested in the three output layers for the instance representation maps of our backward generator G. The challenge here is that the ExM domain is unlabeled and that we do not have a domain-specific supervised signal with which we can train G. Fang Liu, the author of SUSAN [18], suggests training the segmentation capabilities of the backward generator G during the reconstruction step of the supervised cycle ($I_X \to I_Y \to I_X$). Liu assumes that the structural information in the data is domain invariant and therefore preserved during the domain transfer. Accordingly, the instance representation maps that G generates when reconstructing an EM image should match the original ground truth. Thus, we have a supervised loss that can guide G in its segmentation. We adopted this loss as it provides a solid basis to train G's segmentation components. However, SUSAN's [18] configuration neglects the unsupervised cycle ($I_Y \to I_X \to I_Y$), which has to learn to generate the instance segmentation by generalizing across domains. Our results show that we successfully reinforced the unsupervised cycle through our semi-supervised segmentation losses. The structural consistency loss (Equ. 4.10) strengthens the link between the weaker generator G and the more robust generator F. CySGAN's layout provides F with a supervised training signal independent of any prior domain transfer. By requiring structural consistency between the instance representations of F and G during the unsupervised cycle, we enable F to guide G in its segmentation. Moreover, the loss prevents G from hallucinating, as this would lead to a drift in segmentations between F and G. Our output space adaptation loss (Equ. 4.11) directs CySGAN's unsupervised cycle to generate segmentations that share the same distribution as the source segmentation. We claim that this loss helps the model converge and stabilizes the training, particularly during

the first epochs.

We expected the weight sharing to have the most significant impact on the accuracy. Let us summarize how we arrived at this assumption: In the introduction to this paper, we noted that segmentation models, in general, have insufficient cross-domain generalization capabilities. In the related work chapter, we observed that segmentations mainly rely on structural information, which does not depend on the domain - an object's shape and dimension remain the same whether we image it with EM or ExM. We, therefore, deduced that we could circumvent the issue of poor cross-domain generalization by deriving the segmentation from domain invariant encodings, i.e., purely structural information. We assumed that our weight-sharing approach would facilitate the domain invariance as it forces the encoders to focus on the structural information. Nevertheless, the results do not seem to confirm this assumption. The accuracy between *CySGAN w/o semi-supervised segmentation* and the sequential model *CycleGAN + U3D-BCD* differs only by 1.1%. Initially, we thought the BC vs. BCD result confirmed our assumption that weight sharing between the translation and the segmentation components induces strong structural constraints on the translation. Regressing a distance map (D) for an object requires a considerably higher understanding of the object and background than a pixel-wise classification used for the background and contour map. We assumed that this would reflect in the encoding and improve both the generation of the domain-translated images and the generation of the instance representation maps. However, the prior argument indicates that the distance map (D) solely improves the segmentation component. U3D-BCD's original paper [6] supports this argument as it claims an accuracy improvement of 12% from BC to BCD that is close to ours.

Based on our ablation study, we argue that augmentation that includes blurring and occlusions should be a standard tool for domain adaptation. We hypothesize that for a 3D context and model, occlusion and blurring can immensely impact the model's spatial awareness. We base this claim on the observation that the augmentations lead to reduced hallucinations and that the model learns to reconstruct even strongly obscured instances. Conversely, omitting the augmentation schema increases the number of false positives. Additionally, we observed that the augmentation schema increases the model's robustness against mode collapse. Our augmentation schema does not mask consecutive layers to preserve neighborhoods from which the model can recover the information. Hence, we are unsure if using blur and occlusion is also helpful in a 2D context.

Concerning the sequential models' direction-dependent performance, we assume this is due to the segmentation components' robustness to noise arising from the domain transfer. For the direction $I_X \rightarrow I_Y$, we train the segmentation model on synthesized images $I_{Y'}$. These images contain hallucinations and omissions since the domain transfer is not perfect. Accordingly, the segmentation model learns to cope with the corrupted training data during its training process. In contrast, for the direction $I_Y \rightarrow I_X$, we train a segmentation model in the I_X domain. We then segment $I_{X'}$, which we generate during inference by applying the domain transfer to I_Y. In this way, the segmentation model has no chance to learn the imperfect domain transfer.

Additionally, we want to highlight the significance of detaching the synthesized images

when computing the segmentation losses, as described in the implementation section. Neglecting this detail can lead to unexpected results since we pass the segmentation-based losses through the image transfer layer of the first generator.

Finally, the problems we encountered in finding suitable domain-adaptive 3D instance segmentation algorithms to compare and evaluate our CySGAN underscore the urgent need for our method.

7.2. Future Work

The domain gap between the source and target objects is relatively small in our application scenario. Thus, the next logical step is to evaluate our method on modalities with more significant domain gaps.

Further quantitative and qualitative analysis are desirable concerning the impact of our augmentation schema. In particular, our claim of increased spacial awareness needs additional assessment. Applying our augmentation scheme to a simple domain transfer task instead of a domain adaptive segmentation task could help better understand its effect.

While our design reduces complexity and enables a smooth implementation, it is questionable if we exploited its full potential for increasing the segmentation accuracy. It would, therefore, make sense to investigate if the coupling between the translation and the segmentation components is as complete as we assumed. A first step would be to evaluate the sequential setup *CycleGAN+U3D-BC*.

A further enhancement step for our model could be the fusion of the encoders of both generators. Multiple previous works, including SIFA [10], propose this adaptation. Not only does the fusion of the encoders simplify the model, but it can also help generate domain invariant encodings. Although, in our analysis, we derived that the decoders are the central source of hallucinations, calling into question the impact of this adaptation.

A. Appendix

This appendix covers additional background information on algorithms, general Deep Learning (DL) concepts, and loss functions. We linked all of the sections in the relevant spots throughout this thesis.

A.1. Algorithms

A.1.1. Watershed Algorithm

In image processing, the watershed algorithm is a transformation applied to grayscale images. The watershed algorithm interprets the brightness of each point as its elevation and treats the pixel values as local topography. Subsequent flooding of the topographic landscape forms watersheds between contiguous reservoirs, from which we derive the individual water basins. There are several variants of the watershed algorithm. For segmentation, we generally use the marker-driven watershed transformation. In the marker-driven watershed, we flood the basins starting from certain marker positions that we derive from the instance representation maps until they meet with other basins at watershed lines [50, 27]. The U3D-BCD, for example, derives the markers by choosing points with a high foreground probability, a high distance value, and low contour probability [6].

A.1.2. Histogram Matching

A histogram is a block diagram with one block for each quantization level. The height of the block indicates how often the value of the quantization level occurs in the data. For gray images, the quantization level is 256 for the 256 different gray level values. The height of each block corresponds to the number of pixels with the value of the particular quantization level. In other words, a histogram is the intensity distribution of an image [64]. A histogram is not only an analysis tool, but we can also use it for image adaptations like contrast enhancement. We can achieve contrast enhancement via histogram equalization by, e.g., calculating the cumulative distributive function (CDF). To calculate the CDF, we iterate over the histogram and update each quantization level by accumulating its amplitude value with all the amplitude values of the quantization levels to its left. Histogram matching goes one step further and adjusts the contrast of a source image based on the contrast of a target image. Accordingly, histogram matching is a type of distribution matching [65]. For Histrogramm matching, we first derive the histogram for both images and then the CDF of those two histograms. For each quantization level of the source image, we now match the corresponding CDF value with the closest CDF value from the target image. We then replace the CDF value

of the source image with the original unequalized histogram value of the matched CDF value of the target image [65].

A.2. Deep Learning Networks

A.2.1. Markovian discriminator (PatchGAN)

The L2 loss tends to produce blurry results in the image generation task as it penalizes low-frequency errors more strongly than high-frequency errors. The same holds for the L1 loss, albeit considerably less severe. Isola *et al.* [44], the developers of PatchGAN, propose that instead of deriving a new framework that can handle high and low frequencies, we can enforce correctness at low frequencies by shifting the attention to the structure in local image fields and thereby restricting the high-frequency components. To this end, they split the generated data into NxN patches. The discriminator then evaluates the realness of the generation on patch level and outputs the average overall patches as the final result. Isola *et al.* [44] speak of a Markovian discriminator since their framework assumes independence between pixels farther apart than a patch diameter.

A.2.2. U-Net

A U-Net is a CNN model architecture, introduced by Ronneberger *et al.* [17], that follows an encoder-decoder structure. The encoder effectively applies a lossy data compression that downsamples the data using pooling operations while simultaneously growing the number of feature channels. The decoder tries to reverse the compression by typically using up-convolutions and remodeling the original data shape. A fundamental design choice of the U-Net architecture is the concatenation of feature maps from intermediate steps of the encoding path with the input of its mirror-symmetric counterpart from the decoding path [17]. In Ronneberger's *et al.* [17] original implementation, the encoder applies multiple consecutive 3x3 convolutions with Relu activation functions and a single 2x2 pooling layer. Accordingly, the input size is halved at each step. The decoder applies upsampling and a 2x2 convolution, doubling the input size at each step, followed by two 3x3 convolutions and Relu activations. The last layer of the decoder is a 1x1 convolution. It matches the channel number of the output with that of the input.

A.3. Fundamental Concepts (DL)

A.3.1. Mode Collapse

Mode collapse is a common problem in training GANs (see section 2.4.2). The two main evaluation criteria for GANs are that they can fool the discriminator and that the generated samples reflect the diversity of the distribution of the real data. We speak of mode collapse when a model fails at the second criterion. In the case of mode collapse, the generator finds

that it achieves a comparatively low loss with a particular generated sample. It then proceeds to generate only that particular sample [35].

A.3.2. Weight Sharing

Weight sharing effectively means using one network structure for multiple purposes. For example, we could use two different networks to extract a latent space representation for image data from two different domains. However, if we are only interested in domain-invariant structural information, we can use a single network for this task and train it to extract only non-domain-specific components. In such a case, we would say that the extraction networks for both domains are sharing their weights [35]. Further, "weight sharing" indicates that different models share a network structure while having individual subcomponents. Such subcomponents could, for example, be multiple network heads. The head of an ANN comprises one or more task-specific layers that follow the general network structure used for feature extraction. The extracted features are passed to several task-specific layer blocks if a model has multiple heads. For example, we can use the same network to generate an image and a semantic segmentation mask. In this case, the model would need to diverge at least in the output layer; image generation would require a convolutional layer head, while segmentation would require a multi-class soft-max classification head that generates class probabilities for each pixel. The idea behind weight sharing is twofold. On the one hand, it makes the model lighter by reusing the existing infrastructure. On the other hand, many tasks need similar information and can potentially benefit from each other [35].

A.3.3. Self-Supervised Learning

Self-Supervised Learning (SSL) is an intermediate form of supervised and unsupervised learning. In Deep Learning, we train supervised models using labeled data while unsupervised models learn using unlabeled data. An example of an unsupervised task is clustering. SSL is often the first stage of a two-stage training process used to initialize the network and teach it a semantic understanding of the data. A popular form of SSL is contrastive learning, which encourages models to derive more similar encodings for augmentations of the same input compared to augmentations of different inputs [35].

A.3.4. Semi-Supervised Learning

As opposed to SSL, semi-supervised learning uses unlabeled data to complement the supervised learning step instead of acting as a pre-training step for the network initialization. The following is an example of a semi-supervised learning technique: We have two data sets. One small labeled data set and one large unlabeled data set. First, we train our model using the labeled data. Afterward, we predict the labels for unlabeled samples. If the confidence score of a predicted label for an unlabeled sample exceeds a predefined threshold, we assign the label as a pseudo label to that sample and add it to the labeled dataset. If the confidence

score is below the threshold, the sample remains in the unlabeled dataset. We repeat this process for a set number of epochs or until reaching a particular accuracy [35].

A.4. Losses

A.4.1. L1

The Absolute Error Loss (L1) is the absolute difference between the ground truth and the predicted value [35].

$$\text{L1} = y_{\text{ground truth}} - y_{\text{prediction}} \tag{A.1}$$

A.4.2. MAE

The Least Absolute Error (MAE) loss minimizes the error that is the average of the sum of all absolute differences between the ground truth and the predicted value [35].

$$\text{L1} = \frac{1}{n}\sum_{i=1}^{n}\left(y_{\text{ground truth}} - y_{\text{prediction}}\right) \tag{A.2}$$

A.4.3. L2

The Squared Error Loss (L2) is the squared difference between the ground truth and the predicted value [35].

$$\text{L2} = \left(y_{\text{ground truth}} - y_{\text{prediction}}\right)^2 \tag{A.3}$$

A.4.4. MSE

The Mean Square Error (MSE) loss minimizes the error that is the average of the sum of all the squared differences between the ground truth and the predicted value. Accordingly, the MSE focuses on large differences since squaring them inflates their impact. However, this also means that outliers can have an over-proportional impact on loss [35].

$$\text{L2} = \frac{1}{n}\sum_{i=1}^{n}\left(y_{\text{ground truth}} - y_{\text{prediction}}\right)^2 \tag{A.4}$$

A.4.5. BCE

We use the Binary Cross Entropy (BCE) loss for binary classification problems, i.e., problems in which the output has two possible values like "yes" or "no" represented through "1" and "0" [35]. The loss is defined as follows:

$$H_q(p) = -\frac{1}{N}\sum_{i}^{N}(1 - y_i) \cdot \log\left(1 - p\left(y_i\right)\right) + \left(y_i\right) \cdot \log\left(p\left(y_i\right)\right) \tag{A.5}$$

y_i is the ground truth label and $p\left(y_i\right)$ is the model's prediction. If the $y_i = 0$, the second term in the sum becomes zero, reducing the function to:

$$H_q(p) = -\frac{1}{N}\sum_{i}^{N}\log\left(1 - p\left(y_i\right)\right) \tag{A.6}$$

If the $y_i = 1$ the first term in the sum becomes zero and the function is reduced to:

$$H_q(p) = -\frac{1}{N}\sum_{i}^{N}\log\left(p\left(y_i\right)\right) \tag{A.7}$$

Bibliography

[1] J. W. Lichtman and W. Denk. "The big and the small: Challenges of imaging the brain's circuits". In: *Science* 334 (6056 2011), pp. 618–623. ISSN: 10959203. DOI: 10.1126/science.1209168.

[2] J. W. Lichtman, J. Livet, and J. R. Sanes. "A technicolour approach to the connectome". In: *Nature Reviews Neuroscience* 13 (4 2012), p. 217. ISSN: 1471003X. DOI: 10.1038/nrn3217.

[3] A. Shapson-Coe, M. Januszewski, D. R. Berger, A. Pope, Y. Wu, T. Blakely, R. L. Schalek, P. Li, S. Wang, J. Maitin-Shepard, N. Karlupia, S. Dorkenwald, E. Sjostedt, L. Leavitt, D. Lee, L. Bailey, A. Fitzmaurice, R. Kar, B. Field, H. Wu, J. Wagner-Carena, D. Aley, J. Lau, Z. Lin, D. Wei, H. Pfister, A. Peleg, V. Jain, and J. W. Lichtman. "A connectomic study of a petascale fragment of human cerebral cortex". In: *bioRxiv* (2021), p. 2021.05.29.446289. DOI: 10.1101/2021.05.29.446289. URL: https://doi.org/10.1101/2021.05.29.446289.

[4] F. Chen, P. W. Tillberg, and E. S. Boyden. "Expansion microscopy". In: *Science* 347 (6221 2015), pp. 543–548.

[5] L. Lauenburg, Z. Lin, R. Zhang, M. D. Santos, S. Huang, I. Arganda-Carreras, E. S. Boyden, H. Pfister, and D. Wei. "Instance Segmentation of Unlabeled Modalities via Cyclic Segmentation GAN". In: *arxiv* (2022). URL: https://arxiv.org/pdf/2204.03082.pdf.

[6] Z. Lin, D. Wei, M. D. Petkova, Y. Wu, Z. Ahmed, K. S. K, S. Zou, N. Wendt, J. Boulanger-Weill, X. Wang, N. Dhanyasi, I. Arganda-Carreras, F. Engert, J. Lichtman, and H. Pfister. "NucMM Dataset: 3D Neuronal Nuclei Instance Segmentation at Sub-Cubic Millimeter Scale". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention* (2021), pp. 164–174. DOI: 10.1007/978-3-030-87193-2_16. URL: https://connectomics-bazaar.github.io/proj/nucMM/index.html..

[7] L. F. Abbott, D. D. Bock, E. M. Callaway, W. Denk, C. Dulac, A. L. Fairhall, I. Fiete, K. M. Harris, M. Helmstaedter, V. Jain, N. Kasthuri, Y. LeCun, J. W. Lichtman, P. B. Littlewood, L. Luo, J. H. Maunsell, R. C. Reid, B. R. Rosen, G. M. Rubin, T. J. Sejnowski, H. S. Seung, K. Svoboda, D. W. Tank, D. Tsao, and D. C. V. Essen. "The Mind of a Mouse". In: *Cell* 182 (6 Sept. 2020), pp. 1372–1376. ISSN: 10974172. DOI: 10.1016/J.CELL.2020.08.010.

[8] Y. Zou, Z. Yu, V. Kumar, and J. Wang. "Unsupervised Domain Adaptation for Semantic Segmentation via Class-Balanced Self-Training". In: *Proceedings of the European conference on computer vision (ECCV)* (2018), pp. 289–305.

[9] Z. Zhang, L. Yang, and Y. Zheng. "Translating and Segmenting Multimodal Medical Volumes with Cycle- and Shape-Consistency Generative Adversarial Network". In: *Proceedings of the IEEE conference on computer vision and pattern Recognition* (2018), pp. 9242–9251. ISSN: 10636919. DOI: 10.1109/CVPR.2018.00963.

[10] C. Chen, Q. Dou, H. Chen, J. Qin, and P. A. Heng. "Synergistic image and feature adaptation: Towards cross-modality domain adaptation for medical image segmentation". In: *Proceedings of the AAAI conference on artificial intelligence* (2019), pp. 865–872. ISSN: 2159-5399. DOI: 10.1609/aaai.v33i01.3301865.

[11] H. Yang, J. Sun, A. Carass, C. Zhao, J. Lee, and J. L. Prince. "Unsupervised MR-to-CT Synthesis Using Structure-Constrained CycleGAN". In: *IEEE transactions on medical imaging* 39 (12 2020), pp. 4249–4261.

[12] V. Sandfort, K. Yan, P. J. Pickhardt, and R. M. Summers. "Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks". In: *Scientific Reports* 9 (1 Dec. 2019). ISSN: 20452322. DOI: 10.1038/s41598-019-52737-x. URL: /pmc/articles/PMC6858365/.

[13] M. Januszewski and V. Jain. "Segmentation-Enhanced CycleGAN". In: *bioRxiv* (2019). ISSN: 2692-8205. DOI: 10.1101/548081. URL: https://doi.org/10.1101/548081.

[14] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros. "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks". In: *Proceedings of the IEEE International Conference on Computer Vision* 2017-Octob (2017), pp. 2242–2251. ISSN: 15505499. DOI: 10.1109/ICCV.2017.244.

[15] A. Tomczak, S. Ilic, G. Marquardt, T. Engel, F. Forster, N. Navab, and S. Albarqouni. "Multi-task multi-domain learning for digital staining and classification of leukocytes". In: *IEEE Transactions on Medical Imaging* 40 (10 2020), pp. 2897–2910. ISSN: 1558254X. DOI: 10.1109/TMI.2020.3046334.

[16] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger. "3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9901 LNCS (June 2016), pp. 424–432. ISSN: 16113349. DOI: 10.48550/arxiv.1606.06650. URL: https://arxiv.org/abs/1606.06650v1.

[17] O. Ronneberger, P. Fischer, and T. Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *International Conference on Medical image computing and computer-assisted intervention* (Oct. 2015), pp. 234–241. URL: http://lmb.informatik.uni-freiburg.de/.

[18] F. Liu. "SUSAN: segment unannotated image structure using adversarial network". In: *Magnetic Resonance in Medicine* 81 (5 May 2019), pp. 3330–3345. ISSN: 15222594. DOI: 10.1002/mrm.27627.

[19] K. Lee, J. Zung, P. L. Google, V. J. Google, and H. S. Seung. "Superhuman Accuracy on the SNEMI3D Connectomics Challenge". In: *arXiv* (May 2017). DOI: 10.48550/arxiv.1706.00120. URL: https://arxiv.org/abs/1706.00120v1.

[20] J. W. Lichtman and W. Denk. "The big and the small: Challenges of imaging the brain's circuits". In: *Science* 334 (6056 Nov. 2011), pp. 618–623. ISSN: 10959203. DOI: 10.1126/SCIENCE.1209168. URL: https://www.science.org.

[21] D. H. Hubel and T. N. Wiesel. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex". In: *The Journal of Physiology* 160 (1 Jan. 1962), pp. 106–154. ISSN: 1469-7793. DOI: 10.1113/JPHYSIOL.1962.SP006837. URL: https://physoc.onlinelibrary.wiley.com/doi/10.1113/jphysiol.1962.sp006837.

[22] A. Ul-Hamid. *A Beginners' Guide to Scanning Electron Microscopy*. Vol. 1. Springer International Publishing, 2018. DOI: 10.1007/978-3-319-98482-7.

[23] H. J. ten Donkelaar, M. Lammens, and A. Hori. *Clinical Neuro-Embryology - Development and Developmental Disorders of the Human Central Nervous System*. Springer-Verlag Berlin Heidelberg, 2006. ISBN: 3-540-29140-7. URL: https://link.springer.com/content/pdf/10.1007/3-540-34659-7.pdf.

[24] A. Talwar, Z. Lin, D. Wei, Y. Wu, B. Zheng, J. Zhao, W. D. Jang, X. Wang, J. Lichtman, and H. Pfister. "A topological nomenclature for 3D shape analysis in connectomics". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* 2020-June (2020), pp. 4245–4253. ISSN: 21607516. DOI: 10.1109/CVPRW50498.2020.00501.

[25] Q.-Q. Sun, W. Driever, K. G. Pratt, M. Hibi, M. Ma, S. Kler, and Y. A. Pan. "Structural Neural Connectivity Analysis in Zebrafish With Restricted Anterograde Transneuronal Viral Labeling and Quantitative Brain Mapping". In: *Frontiers in Neural Circuits | www.frontiersin.org* 1 (Jan. 2020), pp. 1–13. DOI: 10.3389/fncir.2019.00085. URL: www.frontiersin.org.

[26] M. Januszewski, J. Kornfeld, P. H. Li, A. Pope, T. Blakely, L. Lindsey, J. Maitin-Shepard, M. Tyka, W. Denk, and V. Jain. "High-precision automated reconstruction of neurons with flood-filling networks". In: *Nature methods* 15 (8 2018), pp. 605–610. DOI: 10.1038/s41592-018-0049-4. URL: https://doi.org/10.1038/s41592-018-0049-4.

[27] J. Cousty, G. Bertrand, L. Najman, and M. Couprie. "Watershed cuts: Minimum spanning forests and the drop of water principle". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31 (8 2009), pp. 1362–1374. ISSN: 01628828. DOI: 10.1109/TPAMI.2008.173.

[28] D. C. Cires¸ancires¸an, A. Giusti, L. M. Gambardella, and J. ¨. U. Schmidhuber. "Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images". In: *Advances in neural information processing systems* 25 (2012). URL: http://www.idsia.ch/.

[29] K. Lee, A. Zlateski, A. Vishwanathan, and H. S. Seung. "Recursive Training of 2D-3D Convolutional Networks for Neuronal Boundary Detection". In: *Advances in Neural Information Processing Systems* 28 (2009). URL: http://seunglab.org/data/.

[30] D. Wei, Z. Lin, D. Franco-Barranco, N. Wendt, X. Liu, W. Yin, X. Huang, A. Gupta, W.-D. Jang, X. Wang, I. Arganda-Carreras, J. W. Lichtman, and H. Pfister. "MitoEM Dataset: Large-scale 3D Mitochondria Instance Segmentation from EM Images HHS Public Access". In: *Med Image Comput Comput Assist Interv* 12265 (2020), pp. 66–76. DOI: 10.1007/978-3-030-59722-1_7. URL: https://donglaiw.github.io/page/mitoEM/index.html..

[31] C. Stringer, T. Wang, M. Michaelos, and M. Pachitariu. "Cellpose: a generalist algorithm for cellular segmentation". In: *Nature Methods 2020 18:1* 18 (1 Dec. 2020), pp. 100–106. ISSN: 1548-7105. DOI: 10.1038/s41592-020-01018-x. URL: https://www.nature.com/articles/s41592-020-01018-x.

[32] U. Schmidt, M. Weigert, C. Broaddus, and G. Myers. "Cell Detection with Star-Convex Polygons". In: *International Conference on Medical Image Computing and Computer-Assisted Intervention* (Sept. 2018), pp. 265–273. DOI: 10.1007/978-3-030-00934-2_30. URL: https://doi.org/10.1007/978-3-030-00934-2_30.

[33] M. Weigert, U. Schmidt, R. Haase, K. Sugawara, and G. Myers. "Star-convex Polyhedra for 3D Object Detection and Segmentation in Microscopy". In: *IEEE/CVF* (2020), pp. 3666–3673.

[34] Y. Pang, J. Lin, T. Qin, and Z. Chen. "Image-to-Image Translation: Methods and Applications". In: *IEEE Transactions on Multimedia* (Jan. 2021). URL: https://arxiv.org/abs/2101.08629v2.

[35] F. Chollet. *Deep Learning with Python*. Ed. by T. Arritola, J. Gaines, A. Dragosavljevic, and T. Taylor. First Edit. Manning Publications Co., 2018, p. 384. ISBN: 9781617294433.

[36] K. Sohn, X. Yan, and H. Lee. "Learning Structured Output Representation using Deep Conditional Generative Models". In: *Advances in neural information processing systems* 28 (2015). URL: https://proceedings.neurips.cc/paper/2015/file/8d55a249e6baa5c06772297520da2051-Paper.pdf.

[37] R. Wei and A. Mahmood. "Recent Advances in Variational Autoencoders with Representation Learning for Biomedical Informatics: A Survey". In: *IEEE Access* 9 (2021), pp. 4939–4956. ISSN: 21693536. DOI: 10.1109/ACCESS.2020.3048309.

[38] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. "beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework | OpenReview". In: (2016). URL: https://openreview.net/forum?id=Sy2fzU9gl.

[39] S. Zhao, J. Song, and S. Ermon. "InfoVAE: Balancing Learning and Inference in Variational Autoencoders". In: *Proceedings of the aaai conference on artificial intelligence* 33 (1 2018), pp. 5885–5892.

[40] A. Boesen, L. Larsen, S. K. Sønderby, H. Larochelle, O. Winther, and O. Dk. "Autoencoding beyond pixels using a learned similarity metric". In: *nternational conference on machine learning* (June 2016), pp. 1558–1566.

[41] M.-Y. Liu, T. Breuel, and J. Kautz. "Unsupervised Image-to-Image Translation Networks". In: *Advances in neural information processing systems* 30 (2018). URL: https://github.com/mingyuliutw/unit..

[42] L. Ma, X. Jia, S. Georgoulis, T. Tuytelaars, L. V. Gool, E. Zurich, H. Noah, and A. Lab. "Exemplar Guided Unsupervised Image-to-Image Translation with Semantic Consistency". In: *arXiv* (May 2019).

[43] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. "Generative Adversarial Nets". In: *Communications of the ACM* 11 (Oct. 2015), pp. 139–144. URL: http://www.github.com/goodfeli/adversarial.

[44] P. Isola, J.-Y. Zhu, T. Zhou, A. A. Efros, and B. A. Research. "Image-to-Image Translation with Conditional Adversarial Networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 1125–1134. URL: https://github.com/phillipi/pix2pix..

[45] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. I. Openai, K. Saenko, A. A. Efros, and T. D. Bair. "Cycada: Cycle-consistent adversarial domain adaptation". In: *International conference on machine learning* (July 2017), pp. 1989–1998.

[46] C. Chu, A. Zhmoginov, and M. Sandler. "CycleGAN, a Master of Steganography". In: *arXiv* (Dec. 2017). URL: http://arxiv.org/abs/1712.02950.

[47] H. Sun, R. Fan, C. Li, Z. Lu, K. Xie, X. Ni, and J. Yang. "Imaging Study of Pseudo-CT Synthesized From Cone-Beam CT Based on 3D CycleGAN in Radiotherapy". In: *Frontiers in Oncology* 11 (Mar. 2021), p. 436. ISSN: 2234943X. DOI: 10.3389/fonc.2021.603844.

[48] A. Wang, Q. Zhang, Y. Han, S. Megason, S. Hormoz, K. R. Mosaliganti, J. C. Lam, and V. O. Li. "A novel deep learning-based 3D cell segmentation framework for future image-based disease detection". In: *Scientific Reports 2022 12:1* 12 (1 Jan. 2022), pp. 1–15. ISSN: 2045-2322. DOI: 10.1038/s41598-021-04048-3. URL: https://www.nature.com/articles/s41598-021-04048-3.

[49] L. Chen, M. Strauch, and D. Merhof. "Instance segmentation of biomedical images with an object-aware embedding learned with local constraints". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11764 LNCS (2019), pp. 451–459. ISSN: 16113349. DOI: 10.1007/978-3-030-32239-7_50/FIGURES/4. URL: https://link.springer.com/chapter/10.1007/978-3-030-32239-7_50.

[50] A. Zlateski and H. S. Seung. "Image Segmentation by Size-Dependent Single Linkage Clustering of a Watershed Basin Graph". In: *arXiv* (May 2015). DOI: 10.48550/arxiv.1505.00249. URL: https://arxiv.org/abs/1505.00249v1.

[51] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley. "Least Squares Generative Adversarial Networks". In: *Proceedings of the IEEE International Conference on Computer Vision* 2017-October (Nov. 2016), pp. 2813–2821. ISSN: 15505499. DOI: 10.48550/arxiv.1611.04076. URL: https://arxiv.org/abs/1611.04076v3.

[52] S. C. Turaga, K. L. Briggman, M. Helmstaedter, W. Denk, and H. S. Seung. "Maximin affinity learning of image segmentation". In: *Advances in Neural Information Processing Systems 22 - Proceedings of the 2009 Conference* (Nov. 2009), pp. 1865–1873. DOI: 10.48550/arxiv.0911.5372. URL: https://arxiv.org/abs/0911.5372v1.

[53] A. Chartsias, T. Joyce, R. Dharmakumar, and S. A. Tsaftaris. "Adversarial Image Synthesis for Unpaired Multi-Modal Cardiac Data". In: *International workshop on simulation and synthesis in medical imaging* (Sept. 2017), pp. 3–13.

[54] G. Zeng, T. D. Lerch, F. Schmaranzer, G. Zheng, J. Burger, K. Gerber, M. Tannast, K. Siebenrock, and N. Gerber. "ICMSC: Intra-and Cross-modality Semantic Consistency for Unsupervised Domain Adaptation on Hip Joint Bone Segmentation". In: *arXiv* (Dec. 2020). URL: https://arxiv.org/pdf/2012.12570.pdf.

[55] Y. Huo, Z. Xu, S. Bao, A. Assad, R. G. Abramson, and B. A. Landman. "Adversarial Synthesis Learning Enables Segmentation Without Target Modality Ground Truth". In: *IEEE 15th international symposium on biomedical imaging* (Apr. 2017), pp. 1217–1220.

[56] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker. "Learning to Adapt Structured Output Space for Semantic Segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition* (2020), pp. 7472–7481.

[57] D. Liu, D. Zhang, Y. Song, F. Zhang, L. O'Donnell, H. Huang, M. Chen, and W. Cai. "Unsupervised Instance Segmentation in Microscopy Images via Panoptic Domain Adaptation and Task Re-weighting". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (May 2020), pp. 4242–4251. ISSN: 10636919. DOI: 10.48550/arxiv.2005.02066. URL: https://arxiv.org/abs/2005.02066v1.

[58] K. He, G. Gkioxari, P. Dollar, and R. Girshick. "Mask R-CNN". In: *Proceedings of the IEEE international conference on computer vision* (2017), pp. 2961–2969.

[59] H. Yang, J. Sun, A. Carass, C. Zhao, J. Lee, Z. Xu, and J. Prince. "Unpaired Brain MR-to-CT Synthesis using a Structure-Constrained CycleGAN". In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support* (Sept. 2018), pp. 174–182.

[60] K. Sohn, D. Berthelot, C. L. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel. "FixMatch: Simplifying semi-supervised learning with consistency and confidence". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 596–608. ISSN: 10495258. URL: https://github.com/google-research/fixmatch..

[61] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, D. A. R&d, and T. U. Darmstadt. "The Cityscapes Dataset for Semantic Urban Scene Understanding". In: *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 3213–3223. URL: www.cityscapes-dataset.net.

[62] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. "Microsoft COCO: Common Objects in Context". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 8693 LNCS (PART 5 May 2014), pp. 740–755. ISSN: 16113349. DOI: 10.48550/arxiv.1405.0312. URL: https://arxiv.org/abs/1405.0312v3.

[63] C. Chen, Q. Dou, H. Chen, J. Qin, and P. A. Heng. "Unsupervised Bidirectional Cross-Modality Adaptation via Deeply Synergistic Image and Feature Alignment for Medical Image Segmentation". In: *IEEE transactions on medical imaging* 39 (7 Feb. 2020), pp. 2494–2505. URL: http://arxiv.org/abs/2002.02255.

[64] B. Jähne. *Digitale Bildverarbeitung*. Vol. 6. Springer, July 2005. ISBN: 3-540-24999-0.

[65] C. F. Hildebolt, R. K. Walkup, G. L. Conover, N. Yokoyama-Crothers, T. Q. Bartlett, M. W. Vannier, M. K. Shrout, and J. J. Camp. "Histogram-matching and histogram-flattening contrast correction methods: a comparison." In: *Dentomaxillofacial Radiology* 25 (1 Jan. 2014), pp. 42–47. ISSN: 0250832X. DOI: 10.1259/DMFR.25.1.9084285. URL: https://www.birpublications.org/doi/10.1259/dmfr.25.1.9084285.