From Specific to Universal: One Biomedical Image Segmentation Model To Rule Them All

by

Seyed Alireza Vaezi

(Under the Direction of Shannon Quinn)

Abstract

This section of the page is where your abstract will go. Fill in this spot in the dissertation.tex file with your abstract. Doctoral 350 words max, 150 max for masters.

Index words: [PUT YOUR LIST OF INDEX WORDS HERE SEPERATED BY COMMAS]

From Specific to Universal: One Biomedical Image Segmentation Model To Rule Them All

by

Seyed Alireza Vaezi

M.S., Iran University of Science and Technology (IUST), Iran, 2016

A Dissertation Submitted to the Graduate Faculty of the
University of Georgia in Partial Fulfillment of the Requirements for the Degree.

Doctor of Philosophy of in Computer Science

Athens, Georgia

2025

From Specific to Universal: One Biomedical Image Segmentation Model To Rule Them All

by

Seyed Alireza Vaezi

Major Professor: Shannon Quinn

Committee: Hamid R. Arabnia
Tianming Liu
Kyle Johnsen

Electronic Version Approved:

Ron Walcott
Dean of the Graduate School
The University of Georgia
Month Year

Dedication

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

THIS PAGE IS OPTIONAL

Acknowledgments

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

THIS PAGE IS OPTIONAL

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Image segmentation started with traditional methods and algorithms such as thresholding, watershed, and optical flow, which rely on pixel intensity. Traditional rule-based methods like Thresholding and Watershed analyze pixel values to identify borders and boundaries within areas of interest. Machine learning-based methods - such as Support Vector Machines (SVM), random forests, and contrastive learning - involve the use of statistical machine learning models and increased the popularity and applicability of segmentation. Finally, deep learning-based methods leverage neural networks to learn hierarchical feature representation from raw images without requiring manual feature engineering. This process saw significant improvement with the introduction of Convolutional Neural Networks (CNN). CNNs are trained to detect features in regions of interest, enabling them to perform similarly on new images. Segmentation techniques can be divided into three categories: supervised, semi-supervised, and unsupervised.

U-Net and its variants are extensively adopted in biomedical image segmentation for their capability to automatically extract features from images without manual intervention or preprocessing. They can learn high-level semantic information and low-level spatial information from large-scale data. U-Net architectures are categorized based on their design and functionality. Basic U-Nets, like the original U-Net and 3D U-Net, are foundational, with the latter extending to 3D data for volumetric segmentation useful in CT and MRI scans. Advanced U-Nets include Attention U-Net, which uses attention mechanisms for precision; Inception U-Net, capturing multi-scale information through varied kernel sizes; Residual U-Net, which incorporates residual connections to aid deep network training; and Dense U-Net, promoting feature reuse via dense connections. Currently, CNNs represent the state-of-the-art in image segmentation, with U-Net being the predominant architecture, especially in the field of biomedical image segmentation. These advancements have greatly improved the accuracy and efficiency of biomedical image analysis, making it an essential tool in various fields of research.

Biomedical images come in a vast variety of formats, types, and modalities. The modalities in medical imaging include computed tomography (CT), magnetic resonance imaging (MRI), positron emission tomography (PET), and ultrasound, while microscopy modalities include fluorescent microscopy, bright-field, lens-free microscopy, light microscopy, volume electron microscopy, and phase contrast microscopy, just to name a few. Similarly, due to the variety of biological structures, segmentation targets can vary

from nuclei and cell membranes to organelles such as mitochondria, cilia, tumors, and lesions, as well as blood vessels, bone, and brain structures. This diversity in imaging techniques and segmentation targets highlights the need for specialized and customizable deep learning models in biomedical applications.

The Segment Anything Model (SAM) can segment an object within an image using user inputs, including a single point, multiple points, an entire mask, a bounding box, or textual descriptions. This functionality is based on the model's inherent ability to recognize objects, which enables it to segment unfamiliar object types without further training, effectively supporting zero-shot learning. Furthermore, the effectiveness of SAM is enhanced by its specialized architecture and the use of a significantly large dataset.

1.1 Challenges

Despite their success, CNN methods face challenges including poor generalizability, limited transferability, and the complexity of model development as well as fine-tuning pre-trained models in biomedical applications. This is due to the fact that manual labeling of data in biomedicine requires expert knowledge and is a costly and time-consuming task, making large and quality annotated datasets scarce. As a result, there exists a vast variety of deep learning models, each tailored to a specific modality and target structure. Unsupervised methods, on the other hand, do not require pre-training or an existing dataset and rely on domain-specific rules and heuristics. Although these methods exhibit less accuracy than CNN methods, they excel in reproducibility and generalizability as they do not depend on prior data knowledge. These different approaches to image segmentation provide a range of options for researchers to choose from, depending on their specific needs and resources.

In the biomedical field, where labeled data is often scarce and costly to obtain, several solutions have been proposed to augment and utilize available data effectively. These include semi-supervised learning, which utilizes both labeled and unlabeled data to enhance learning accuracy by leveraging the data's underlying distribution. Active learning focuses on selectively querying the most informative data points for expert labeling, optimizing the training process by using the most valuable examples. Data augmentation techniques, such as image transformations and synthetic data generation through Generative Adversarial Networks, increase the diversity and volume of training data, enhancing model robustness and reducing overfitting. Transfer learning transfers knowledge from one task to another, minimizing the need for extensive labeled data in new tasks. Self-supervised learning creates its labels by defining a pretext task, like predicting the position of a randomly cropped image patch, aiding in the learning of useful data representations. Additionally, few-shot, one-shot, and zero-shot learning techniques are designed to operate with minimal or no labeled examples, relying on generalization capabilities or metadata for making predictions about unseen classes.

Generalizability refers to the trained model's ability to perform well on unseen data outside of the training set. It is a crucial aspect of machine learning, particularly in biomedical applications, where variations in image acquisition conditions, tissue types, and other factors can be substantial. Data augmentation techniques and transfer learning are two excellent methods to overcome overfitting and improve general-

izability where the training data is small. While transfer learning is a powerful technique for leveraging pre-trained models to boost performance, especially in scenarios with limited data, it does come with its own set of challenges and limitations such as domain mismatch, risk of overfitting, computational demands, and potential biases from the source dataset.

Reproducibility refers to obtaining consistent results using the same input data, computational steps, methods, and conditions of analysis. This concept is key in scientific research to ensure that outcomes can be reliably replicated under the same conditions, fostering trust and confidence in the findings. Reproducibility is influenced by various factors including dataset variability, model architecture specifics, optimization procedures, and computational infrastructure. Apart from the loss of validity of a scientific method, non-reproducibility can lead to wasted resources, stalled scientific progress, erroneous conclusions, and significant ethical concerns. To ensure reproducibility in deep learning for medical image segmentation, Renard et al. advocate for comprehensive documentation, standardized practices, fixed random seeds, cross-validation, multiple evaluation metrics, and sharing of source code and dependencies.

Deep learning models are effective across various applications but their usability depends on several factors such as the complexity of the task at hand, data availability, and the extent of necessary model customization. For users who prefer straightforward applications, ease of use is crucial. They benefit from methods that do not require extensive modifications or tuning to achieve optimal results. Incorporating an intuitive graphical user interface (GUI) and ensuring interactivity can enhance the usability of these tools, making them more accessible to non-expert users, such as biologists, who need practical, ready-to-use solutions without the intricacies of model adjustments.

Chapter 2

A Novel Pipeline for Cell Instance Segmentation, Tracking and Motility Classification of Toxoplasma Gondii in 3D Space

Toxoplasma gondii is the parasitic protozoan that causes disseminated toxoplasmosis, a disease that is estimated to infect around one-third of the world's population. While the disease is commonly asymptomatic, the success of the parasite is in large part due to its ability to easily spread through nucleated cells. The virulence of T. gondii is predicated on the parasite's motility. Thus the inspection of motility patterns during its lytic cycle has become a topic of keen interest. Current cell tracking projects usually focus on cell images captured in 2D which are not a true representation of the actual motion of a cell. Current 3D tracking projects lack a comprehensive pipeline covering all phases of preprocessing, cell detection, cell instance segmentation, tracking, and motion classification, and merely implement a subset of the phases. Moreover, current 3D segmentation and tracking pipelines are not targeted for users with less experience in deep learning packages. Our pipeline, TSeg, on the other hand, is developed for segmenting, tracking, and classifying the motility phenotypes of T. gondii in 3D microscopic images. Although TSeg is built initially focusing on T. gondii, it provides generic functions to allow users with similar but distinct applications to use it off-the-shelf. Interacting with all of TSeg's modules is possible through our Napari plugin which is developed mainly off the familiar SciPy scientific stack. Additionally, our plugin is designed with a user-friendly GUI in Napari which adds several benefits to each step of the pipeline such as visualization and representation in 3D. TSeg proves to fulfill a better generalization, making it capable of delivering accurate results with images of other cell types.

2.1 Introduction

Quantitative cell research often requires the measurement of different cell properties including size, shape, and motility. This step is facilitated using segmentation of imaged cells. With fluorescent markers, computational tools can be used to complete segmentation and identify cell features and positions over time. 2D measurements of cells can be useful, but the more difficult task of deriving 3D information from cell images is vital for metrics such as motility and volumetric qualities.

Toxoplasmosis is an infection caused by the intracellular parasite Toxoplasma gondii. T. gondii is one of the most successful parasites, infecting at least one-third of the world's population. Although Toxoplasmosis is generally benign in healthy individuals, the infection has fatal implications in fetuses and immunocompromised individuals [48]. T. gondii's virulence is directly linked to its lytic cycle which is comprised of invasion, replication, egress, and motility. Studying the motility of T. gondii is crucial in understanding its lytic cycle in order to develop potential treatments.

For this reason, we present a novel pipeline to detect, segment, track, and classify the motility pattern of T. gondii in 3D space. One of the main goals is to make our pipeline intuitively easy to use so that the users who are not experienced in the fields of machine learning (ML), deep learning (DL), or computer vision (CV) can still benefit from it. The other objective is to equip it with the most robust and accurate set of segmentation and detection tools so that the end product has a broad generalization, allowing it to perform well and accurately for various cell types right off the shelf.

PlantSeg uses a variant of 3D U-Net, called Residual 3D U-Net, for preprocessing and segmentation of multiple cell types [64]. PlantSeg performs best among Deep Learning algorithms for 3D Instance Segmentation and is very robust against image noise [22]. The segmentation module also includes the optional use of CellPose [53]. CellPose is a generalized segmentation algorithm trained on a wide range of cell types and is the first step toward increased optionality in TSeg. The Cell Tracking module consolidates the cell particles across the z-axis to materialize cells in 3D space and estimates centroids for each cell. The tracking module is also responsible for extracting the trajectories of cells based on the movements of centroids throughout consecutive video frames, which is eventually the input of the motion classifier module.

Most of the state-of-the-art pipelines are restricted to 2D space which is not a true representative of the actual motion of the organism. Many of them require knowledge and expertise in programming, or in machine learning and deep learning models and frameworks, thus limiting the demographic of users that can use them. All of them solely include a subset of the aforementioned modules (i.e. detection, segmentation, tracking, and classification) [53]. Many pipelines rely on the user to train their own model, hand-tailored for their specific application. This demands high levels of experience and skill in ML/DL and consequently undermines the possibility and feasibility of quickly utilizing an off-the-shelf pipeline and still getting good results.

To address these we present TSeg. It segments T. gondii cells in 3D microscopic images, tracks their trajectories, and classifies the motion patterns observed throughout the 3D frames. TSeg is comprised of four modules: pre-processing, segmentation, tracking, and classification. We developed TSeg as a plugin

for Napari [52] - an open-source fast and interactive image viewer for Python designed for browsing, annotating, and analyzing large multi-dimensional images. Having TSeg implemented as a part of Napari not only provides a user-friendly design but also gives more advanced users the possibility to attach and execute their custom code and even interact with the steps of the pipeline if needed. The preprocessing module is equipped with basic and extra filters and functionalities to aid in the preparation of the input data. TSeg gives its users the advantage of utilizing the functionalities that PlantSeg and CellPose provide. These functionalities can be chosen in the pre-processing, detection, and segmentation steps. This brings forth a huge variety of algorithms and pre-built models to select from, making TSeg not only a great fit for T. gindii, but also a variety of different cell types.

2.2 Background

The recent solutions in generalized and automated segmentation tools are focused on 2D cell images. Segmentation of cellular structures in 2D is important but not representative of realistic environments. Microbiological organisms are free to move on the z-axis and tracking without taking this factor into account cannot guarantee a full representation of the actual motility patterns. As an example, Fazli et al. [10] identified three distinct motility types for T. gondii with two-dimensional data, however, they also acknowledge and state that based established heuristics from previous works there are more than three motility phenotypes for T. gondii. The focus on 2D research is understandable due to several factors. 3D data is difficult to capture as tools for capturing 3D slices and the computational requirements for analyzing this data are not available in most research labs. Most segmentation tools are unable to track objects in 3D space as the assignment of related centroids is more difficult. The additional noise from capture and focus increases the probability of incorrect assignment. 3D data also has issues with overlapping features and increased computation required per frame of time.

Fazli et al. [10] studies the motility patterns of T. gondii and provides a computational pipeline for identifying motility phenotypes of T. gondii in an unsupervised, data-driven way. In that work Ca2+ is added to T. gondii cells inside a Fetal Bovine Serum. T. gondii cells react to Ca2+ and become motile and fluorescent. The images of motile T. gondii cells were captured using an LSM 710 confocal microscope. They use Python 3 and associated scientific computing libraries (NumPy, SciPy, scikit-learn, matplotlib) in their pipeline to track and cluster the trajectories of T. gondii. Based on this work Fazli et al. [11] work on another pipeline consisting of preprocessing, sparsification, cell detection, and cell tracking modules to track T. gondii in 3D video microscopy where each frame of the video consists of image slices taken 1 micro-meters of focal depth apart along the z-axis direction. In their latest work Fazli et al. [12] developed a lightweight and scalable pipeline using task distribution and parallelism. Their pipeline consists of multiple modules: reprocessing, sparsification, cell detection, cell tracking, trajectories extraction, parametrization of the trajectories, and clustering. They could classify three distinct motion patterns in T. gondii using the same data from their previous work.

While combining open source tools is not a novel architecture, little has been done to integrate 3D cell tracking tools. Fazeli et al. [9] motivated by the same interest in providing better tools to non-software

professionals created a 2D cell tracking pipeline. This pipeline combines Stardist [61] and TrackMate [55] for automated cell tracking. This pipeline begins with the user loading cell images and centroid approximations to the ZeroCostDL4Mic [2] platform. ZeroCostDL4Mic is a deep learning training tool for those with no coding expertise. Once the platform is trained and masks for the training set are made for hand-drawn annotations, the training set can be input to Stardist. Stardist performs automated object detection using Euclidean distance to probabilistically determine cell pixels versus background pixels. Lastly, Trackmate uses segmentation images to track labels between timeframes and display analytics.

This Stardist pipeline is similar in concept to TSeg. Both create an automated segmentation and tracking pipeline but TSeg is oriented to 3D data. Cells move in 3-dimensional space that is not represented in a flat plane. TSeg also does not require the manual training necessary for the other pipeline. Individuals with low technical expertise should not be expected to create masks for training or even understand the training of deep neural networks. Lastly, this pipeline does not account for imperfect datasets without the need for preprocessing. All implemented algorithms in TSeg account for microscopy images with some amount of noise.

Wen et al. [62] combines multiple existing new technologies including deep learning and presents 3DeeCellTracker. 3DeeCellTracker segments and tracks cells on 3D time-lapse images. Using a small subset of their dataset they train the deep learning architecture 3D U-Net for segmentation. For tracking, a combination of two strategies was used to increase accuracy: local cell region strategies, and spatial pattern strategy. Kapoor et al. [21] presents VollSeg that uses deep learning methods to segment, track, and analyze cells in 3D with irregular shape and intensity distribution. It is a Jupyter Notebook-based Python package and also has a UI in Napari. For tracking, a custom tracking code is developed based on Trackmate.

Many segmentation tools require some amount of knowledge in Machine or Deep Learning concepts. Training the neural network in creating masks is a common step for open-source segmentation tools. Automating this process makes the pipeline more accessible to microbiology researchers.

2.3 Method

2.3.1 Data

Our dataset consists of 11 videos of *T. gondii* cells under a microscope, obtained from different experiments with different numbers of cells. The videos are on average around 63 frames in length. Each frame has a stack of 41 image slices of size 500×502 pixels along the z-axis (z-slices). The z-slices are captured 1μm apart in optical focal length making them 402μm×401μm×40μm in volume. The slices were recorded in raw format as RGB TIF images but are converted to grayscale for our purpose. This data is captured using a PlanApo 20x objective (NA = 0.75) on a preheated Nikon Eclipse TE300 epifluorescence microscope. The image stacks were captured using an iXon 885 EMCCD camera (Andor Technology, Belfast, Ireland) cooled to -70°C and driven by NIS Elements software (Nikon Instruments, Melville, NY) as part of related research by Ward et al. [33]. The camera was set to frame transfer sensor mode, with a vertical pixel shift

speed of 1.0 µs, vertical clock voltage amplitude of +1, readout speed of 35MHz, conversion gain of 3.8×, EM gain setting of 3 and 2×2 binning, and the z-slices were imaged with an exposure time of 16ms.

2.3.2 Software

Napari Plugin

TSeg is developed as a plugin for Napari - a fast and interactive multi-dimensional image viewer for Python that allows volumetric viewing of 3D images [52]. Plugins enable developers to customize and extend the functionality of Napari. For every module of TSeg, we developed its corresponding widget in the GUI, plus a widget for file management. The widgets have self-explanatory interface elements with tooltips to guide the inexperienced user to traverse through the pipeline with ease. Layers in Napari are the basic viewable objects that can be shown in the Napari viewer. Seven different layer types are supported in Napari: Image, Labels, Points, Shapes, Surface, Tracks, and Vectors, each of which corresponds to a different data type, visualization, and interactivity [52]. After its execution, the viewable output of each widget gets added to the layers. This allows the user to evaluate and modify the parameters of the widget to get the best results before continuing to the next widget. Napari supports bidirectional communication between the viewer and the Python kernel and has a built-in console that allows users to control all the features of the viewer programmatically. This adds more flexibility and customizability to TSeg for the advanced user. The full code of TSeg is available on GitHub under the MIT open source license at `https://github.com/salirezav/tseg`. TSeg can be installed through Napari's plugins menu.

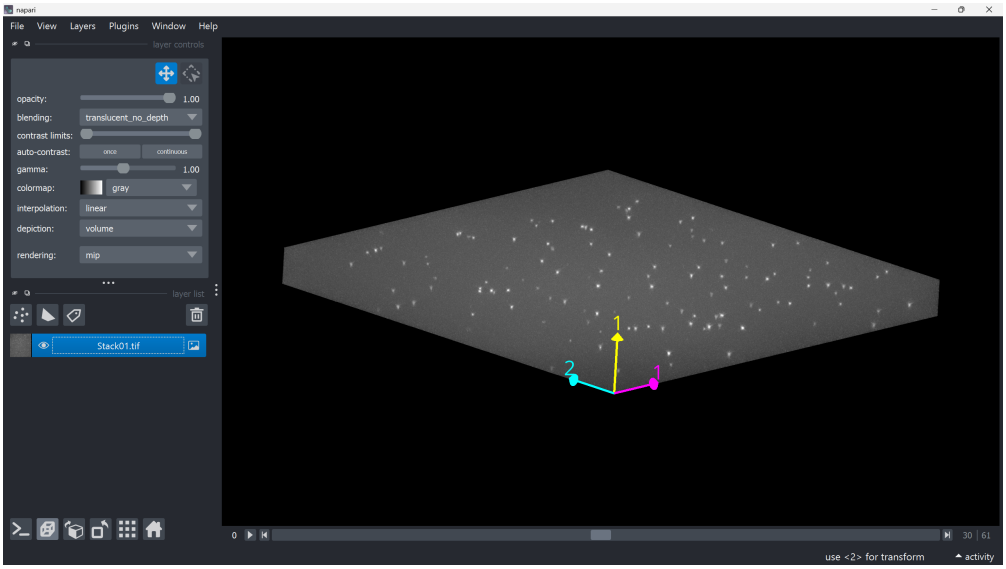

Figure 2.1: TSeg's Napari Plugin Interface

Computational Pipeline

Pre-Processing

Due to the fast imaging speed in data acquisition, the image slices will inherently have a vignetting artifact, meaning that the corners of the images will be slightly darker than the center of the image - Figure 2.2. To eliminate this artifact we added adaptive thresholding and logarithmic correction to the pre-processing module. Furthermore, another prevalent artifact on our dataset images was a Film-Grain noise (AKA salt and pepper noise). To remove or reduce such noise a simple gaussian blur filter and a sharpening filter are included.

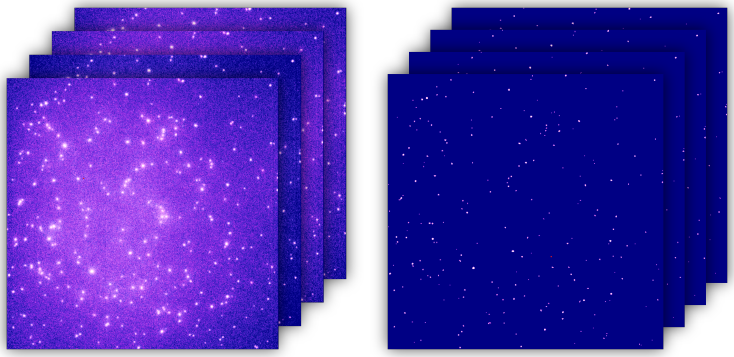

Figure 2.2: On the left, a sample frame of the 3D video of *T. gondii* cells. The image is captured using a PlanApo 20x objective (NA = 0.75) on a preheated Nikon Eclipse TE300 epifluorescence microscope. On the right, the same frame after denoising.

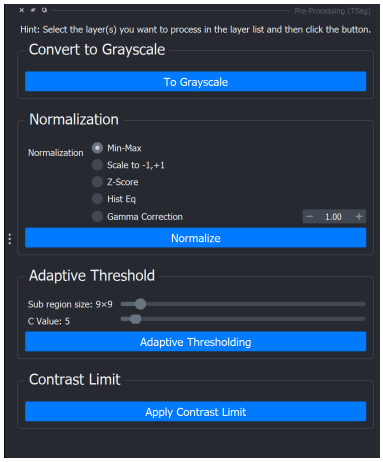

Figure 2.3: The pre-processing widget includes adaptive thresholding, normalization, and noise removal to enhance image quality.

Cell Detection and Segmentation

TSeg's Detection and Segmentation modules are in fact backed by PlantSeg and CellPose. The Detection

Module is built only based on PlantSeg's CNN Detection Module [64], and for the Segmentation Module, only one of the two tools can be selected to be executed as the segmentation tool in the pipeline. Naturally, each of the tools demands specific interface elements different from the others since each accepts different input values and various parameters. TSeg orchestrates this and makes sure the arguments and parameters are passed to the corresponding selected segmentation tool properly and the execution will be handled accordingly. The parameters include but are not limited to input data location, output directory, and desired segmentation algorithm - Figure 2.4. This allows the end-user complete control over the process and feedback from each step of the process. The preprocessed images and relevant parameters are sent to a modular segmentation controller script. As an effort to allow future development on TSeg, the segmentation controller script shows how the pipeline integrates two completely different segmentation packages.

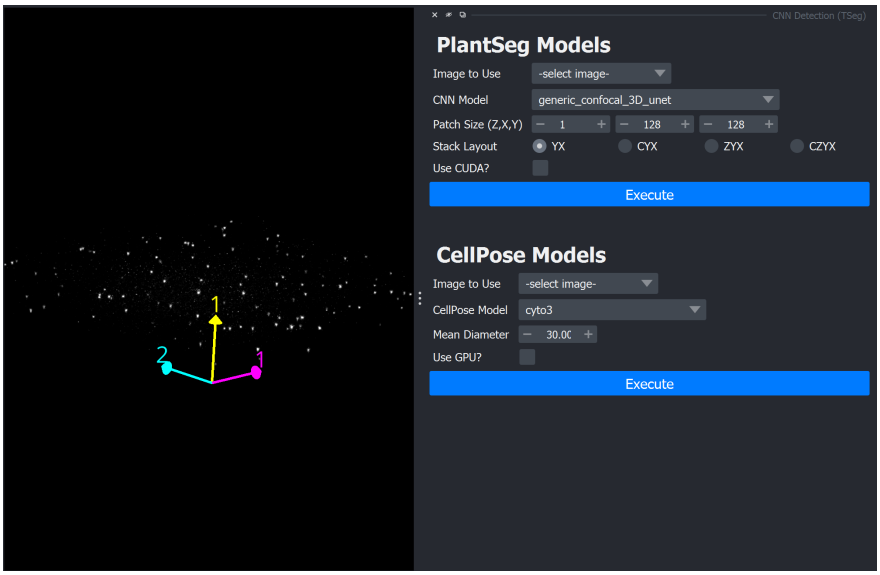

Figure 2.4: The CNN Detection widget integrates PlantSeg for tissue-specific 3D segmentation and CellPose for diverse cell types. These tools are implemented in the backend via their APIs, ensuring seamless operation.

Tracking

The tracking widget of TSeg employs connected component analysis and the Hungarian algorithm for accurate cell tracking across 3D time-lapse images, and, leverages autoregressive modeling to analyze cell trajectories, enabling these trajectories to be clustered in an unsupervised manner for a deeper understanding of motility - Figure 2.6. Features in each segmented image are found using the scipy label function. In order to reduce any leftover noise, any features under a minimum size are filtered out and considered leftover noise. After feature extraction, centroids are calculated using the center of mass function in scipy. The centroid of the 3D cell can be used as a representation of the entire body during tracking. The tracking algorithm goes through each captured time instance and connects centroids to the likely next movement of the cell. Tracking involves a series of measures in order to avoid incorrect assignments. An incorrect

assignment could lead to inaccurate result sets and unrealistic motility patterns. If the same number of features in each frame of time could be guaranteed from segmentation, minimum distance could assign features rather accurately. Since this is not a guarantee, the Hungarian algorithm must be used to associate a cost with the assignment of feature tracking. The Hungarian method is a combinatorial optimization algorithm that solves the assignment problem in polynomial time. The cost for the tracking algorithm determines which feature is the next iteration of the cell's tracking through the complete time series. The combination of distance between centroids for all previous points and the distance to the potential new centroid. If an optimal next centroid can't be found within an acceptable distance of the current point, the tracking for the cell is considered as complete. Likewise, if a feature is not assigned to a current centroid, this feature is considered a new object and is tracked as the algorithm progresses. The complete path for each feature is then stored for motility analysis.

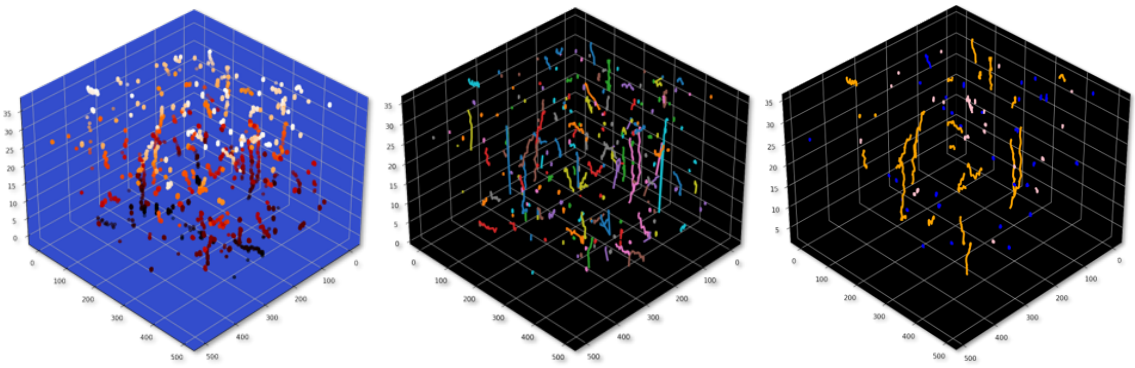

Figure 2.5: Left, 3D connected component labeling (CCL) is used to extract features from the segmented images. Middle, the centroids of the features are calculated using the center of mass function in scipy. Right, the tracking algorithm connects centroids across time instances to track the cells.

Motion Classification

To classify the motility pattern of *T. gondii* in 3D space in an unsupervised fashion we implement and use the method that Fazli et. al. introduced [12]. In that work, they used an autoregressive model (AR); a linear dynamical system that encodes a Markov-based transition prediction method. The reason is that although K-means is a favorable clustering algorithm, there are a few drawbacks to it and to the conventional methods that draw them impractical. Firstly, K-means assumes Euclidian distance, but AR motion parameters are geodesics that do not reside in a Euclidean space, and secondly, K-means assumes isotropic clusters, however, although AR motion parameters may exhibit isotropy in their space, without a proper distance metric, this issue cannot be clearly examined [12].

2.4 Evaluation

TSeg's performance in segmentation was evaluated over datasets introduced in [41]. The Cell Tracking Challenge (CTC) offers a diverse array of 2D and 3D time-lapse microscopy datasets, each capturing

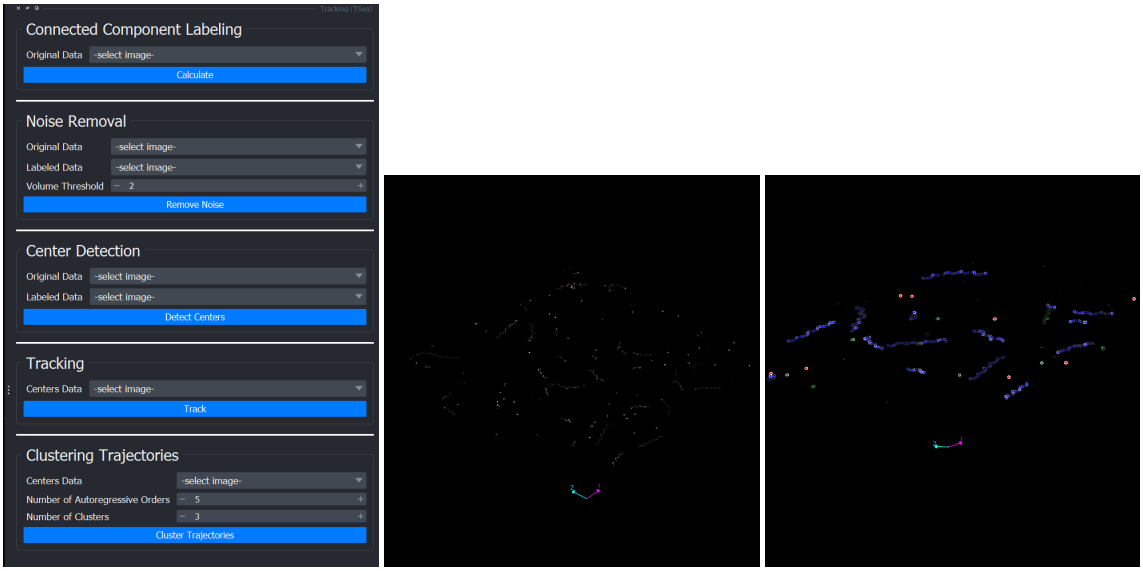

Figure 2.6: The tracking widget allows the user to set the parameters for the tracking and clustering algorithms and visualizes the results.

unique biological specimens under various imaging modalities. Table 2.1 contains is an overview of these datasets, detailing the organisms studied, imaging techniques employed, and acquisition specifics. CellPose has 26 and PlantSeg has 17 different pretrained models that can perform segmentation over 2D and 3D biomedical data. 10 samples from the 2D datasets, and one from the 3D datasets were randomly selected and processed with each of the 43 models using the API provided by PlantSeg and CellPose. Each dataset contains sequences of time-lapse video frames, therefore sample of a 2D dataset is comprised of a single 2D grayscale image, and each sample from the 3D datasets has a stack of 2D images recorded simultaneously across the z-axis to comprise one frame. The prediced masks of the models were evaluated against the provided ground-truth data using the Jaccard Index (JI) score and averaged across all samples of the same dataset. Tables 2.2 and 2.3 show the average JI scores of the models for each dataset.

2.4.1 Performance on 2D Datasets

The results show that segmentation performance varies significantly across datasets and models. Key observations include: **Consistently High Performance:** Models such as `cyto3`, `nuclei`, and `deepbacs_cp3` achieved high IoU scores (above 0.95) across datasets such as `BF-C2DL-HSC` and `BF-C2DL-MuSC`, which suggests that these models are well-suited for segmenting brightfield microscopy images of stem cells.

Challenges with Certain Datasets: The `DIC-C2DH-HeLa` dataset posed difficulties for most models, with average IoU scores significantly lower (around 0.36), indicating that differential interference

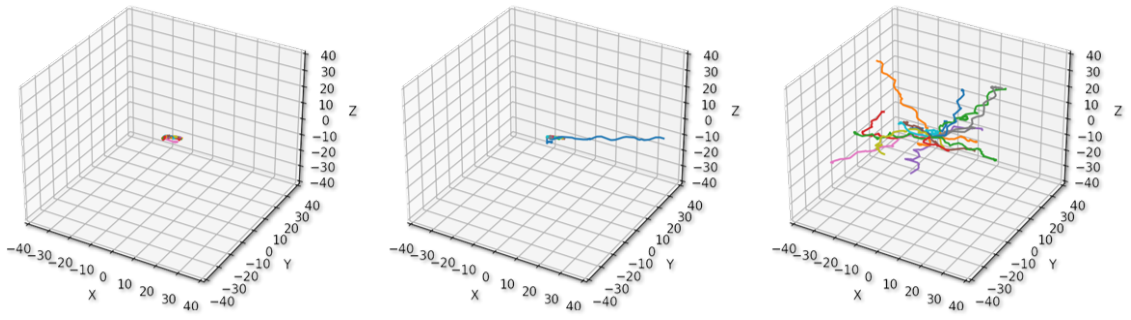

Figure 2.7: Clustering of *T. gondii* motility patterns in 3D space using an autoregressive model (AR) as introduced by Fazli et al. [12]. The AR model addresses the limitations of K-means by considering geodesic distances and non-isotropic clusters.

contrast (DIC) microscopy images present unique segmentation challenges.

Moderate Performance for Fluorescence-Based Data: Models generally performed better on fluorescence microscopy datasets such as `Fluo-C2DL-MSC`, but the performance varied across different fluorescent markers and imaging conditions.

Model-Specific Variability: While some models performed well across multiple datasets, others showed strong dataset-specific biases, highlighting the importance of dataset-matching when selecting segmentation models.

2.4.2 Performance on 3D Datasets

For the 3D datasets, the following trends emerged:

Superior Performance on Some Fluorescence Datasets: Models like `PlantSeg_3Dnuc_platinum` and `lightsheet_3D_unet_root_ds2x` demonstrated high performance (IoU ∼0.96–0.97) on `Fluo-C3DH-A549` and `Fluo-C3DH-A549-SIM`, suggesting strong adaptability to fluorescence-based 3D segmentation.

Lower Performance on More Complex 3D Structures: Datasets such as Fluo-N3DH-CE and Fluo-N3DH-CHO showed lower overall IoU scores, with some models failing to generalize well to these samples.

Sparse Data and Missing Values: Several models had missing scores for certain datasets, indicating either difficulties in processing specific images or incompatibility with the dataset structure.

2.5 Conclusion and Final Remarks

TSeg is an easy to use pipeline designed to study the motility patterns of *T. gondii* in 3D space. It is developed as a plugin for Napari and is equipped with a variety of deep learning based segmentation tools borrowed from PlantSeg and CellPose, making it a suitable off-the-shelf tool for applications incorporating images of cell types not limited to *T. gondii*. Future work on TSeg includes the expansion of implemented algorithms and tools in its preprocessing, segmentation, tracking, and clustering modules.

Table 2.1: Cell Tracking Challenge Datasets

Dataset Name	**Organism**	**Description**	**Imaging Modality**	**Dimension**
BF-C2DL-HSC	Mouse	Hematopoietic stem cells cultured in hydrogel microwells.	Brightfield Microscopy	2D
BF-C2DL-MuSC	Mouse	Muscle stem cells cultured in hydrogel microwells.	Brightfield Microscopy	2D
DIC-C2DH-HeLa	Human	HeLa cells cultured on a flat glass surface.	Differential Interference Contrast (DIC) Microscopy	2D
Fluo-C2DL-Huh7	Human	Huh7 cells expressing the fusion protein YFP-TIA-1.	Fluorescence Microscopy	2D
Fluo-C2DL-MSC	Rat	Mesenchymal stem cells cultured on a flat polyacrylamide substrate.	Fluorescence Microscopy	2D
Fluo-N2DH-GOWT1	Mouse	GFP-GOWT1 stem cells.	Fluorescence Microscopy	2D
Fluo-N2DL-HeLa	Human	HeLa cells stably expressing H2b-GFP.	Fluorescence Microscopy	2D
Fluo-C3DH-A549	Human	A549 lung cancer cells embedded in a Matrigel matrix.	Fluorescence Microscopy	3D
Fluo-C3DH-H157	Human	GFP-transfected H157 lung cancer cells embedded in a Matrigel matrix.	Fluorescence Microscopy	3D
Fluo-C3DL-MDA231	Human	MDA231 human breast carcinoma cells infected with a pMSCV vector including the GFP sequence, embedded in a collagen matrix.	Fluorescence Microscopy	3D
Fluo-N3DH-CE	C. elegans	Developing C. elegans embryo.	Fluorescence Microscopy	3D
Fluo-N3DH-CHO	Chinese Hamster	Chinese Hamster Ovarian (CHO) nuclei overexpressing GFP-PCNA.	Fluorescence Microscopy	3D
Fluo-N3DL-DRO	Drosophila melanogaster	Developing Drosophila melanogaster embryo.	Fluorescence Microscopy	3D
Fluo-N3DL-TRIC	Tribolium castaneum	Developing Tribolium castaneum embryo (3D cartographic projection).	Fluorescence Microscopy	3D
Fluo-N3DL-TRIF	Tribolium castaneum	Developing Tribolium castaneum embryo.	Fluorescence Microscopy	3D

Table 2.2: 2D Segmentation Model Performance Summary

dataset name	BF-C2DL-HSC	BF-C2DL-MuSC	DIC-C2DH-HeLa	Fluo-C2DL-Huh7	Fluo-C2DL-MSC	Fluo-N2DH-GOWT1	Fluo-N2DH-SIM+	Fluo-N2DL-HeLa	PhC-C2DH-U373	PhC-C2DL-PSC
cyto3	0.98	0.95	0.36	0.60	0.90	0.89	0.82	0.75	0.87	0.87
nuclei	0.99	0.99	0.36	0.60	0.89	0.86	0.80	0.75	0.87	0.90
cyto2_cp3	0.94	0.95	0.35	0.60	0.89	0.87	0.82	0.74	0.85	0.87
tissuenet_cp3	0.98	0.97	0.36	0.59	0.89	0.86	0.80	0.75	0.87	0.91
livecell_cp3	0.98	0.99	0.36	0.59	0.89	0.86	0.80	0.75	0.87	0.91
yeast_PhC_cp3	0.98	0.98	0.34	0.59	0.84	0.86	0.71	0.75	0.86	0.88
yeast_BF_cp3	0.99	0.99	0.36	0.60	0.89	0.86	0.79	0.73	0.87	0.91
bact_phase_cp3	0.97	0.97	0.35	0.59	0.89	0.86	0.79	0.73	0.86	0.89
bact_fluor_cp3	0.93	0.95	0.33	0.59	0.89	0.86	0.79	0.73	0.84	0.89
deepbacs_cp3	0.99	0.99	0.36	0.60	0.89	0.86	0.80	0.75	0.87	0.90
cyto2	0.95	0.96	0.36	0.60	0.90	0.88	0.82	0.73	0.86	0.86
cyto	0.97	0.97	0.35	0.59	0.89	0.87	0.80	0.72	0.84	0.86
CPx	0.97	0.98	0.34	0.59	0.89	0.88	0.80	0.73	0.86	0.90
neurips_grayscale_cyto2	0.99	0.98	0.35	0.60	0.89	0.88	0.89	0.67	0.87	0.91
CP	0.97	0.95	0.35	0.60	0.89	0.88	0.81	0.72	0.86	0.89
TN1	0.98	0.99	0.36	0.60	0.89	0.86	0.80	0.75	0.86	0.91
TN2	0.99	0.98	0.36	0.59	0.89	0.86	0.80	0.75	0.87	0.91
TN3	0.99	0.99	0.36	0.60	0.89	0.86	0.80	0.75	0.87	0.91
LC1	0.99	0.99	0.27	0.59	0.89	0.89	0.79	0.75	0.87	0.91
LC2	0.96	0.97	0.36	0.60	0.89	0.86	0.79	0.75	0.85	0.90
LC3	0.98	0.97	0.33	0.60	0.89	0.86	0.80	0.74	0.82	0.85
LC4	0.95	0.98	0.36	0.59	0.89	0.86	0.80	0.75	0.87	0.83
confocal_2D_unet_ovules_ds2x	0.99	0.99	0.36	0.59	0.89	0.86	0.80	0.75	0.87	0.91
lightsheet_2D_unet_root_ds1x	0.99	0.99	0.36	0.59	0.89	0.86	0.80	0.75	0.87	0.91
lightsheet_2D_unet_root_nuclei_ds1x	0.99	0.99	0.36	0.59	0.89	0.86	0.80	0.75	0.87	0.91
confocal_2D_unet_sa_meristem_cells	0.99	0.99	0.36	0.59	0.89	0.86	0.80	0.75	0.87	0.91

Table 2.3: 3D Segmentation Model Performance Summary

dataset_name	Fluo-C3DH-A549	Fluo-C3DH-A549-SIM	Fluo-C3DH-H157	Fluo-N3DH-CE	Fluo-N3DH-CHO	Fluo-N3DH-SIM+
cyto3	0.96	0.97	0.93	0.80	0.83	0.92
nuclei	0.96	0.97	0.93	0.80	0.84	0.93
cyto2_cp3	0.97	0.97	0.93	0.80	0.79	0.93
tissuenet_cp3	0.96	0.97	0.93	0.80	0.84	0.93
livecell_cp3	0.96	0.95	0.93	0.80	0.84	0.93
yeast_PhC_cp3	0.92	0.93	0.88	0.68	0.84	0.86
yeast_BF_cp3	0.95	0.97	0.92	0.80	0.83	0.88
bact_phase_cp3	0.96	0.97	0.93	0.79	0.83	0.93
bact_fluor_cp3	0.96	0.97	0.93	0.80	0.83	0.93
deepbacs_cp3	0.96	0.97	0.93	0.80	0.84	0.93
cyto2	0.96	0.98	0.93	0.80	0.84	0.94
cyto	0.96	0.97	0.93	0.80	0.82	0.93
CPx	0.98	0.99	0.93	0.80	0.83	0.93
neurips_grayscale_cyto2	0.97	0.98	0.93	0.73	0.82	0.92
CP	0.98	0.99	0.93	0.79	0.82	0.93
TN1	0.96	0.97	0.93	0.80	0.84	0.93
TN2	0.96	0.97	0.93	0.80	0.84	0.93
TN3	0.96	0.99	0.93	0.80	0.84	0.93
LC1	0.86	0.97	0.92	0.80	0.82	0.92
LC2	0.96	0.97	0.93	0.80	0.84	0.93
LC3	0.96	0.13	0.93	0.79	0.82	0.30
LC4	0.96	0.97	0.93	0.80	0.84	0.93
generic_confocal_3D_unet	0.96	0.97	0.88	0.78	0.84	—
generic_light_sheet_3D_unet	0.96	0.97	0.88	0.78	0.84	—
confocal_3D_unet_ovules_ds1x	0.96	0.97	0.88	0.78	0.84	—
confocal_3D_unet_ovules_ds2x	0.96	0.97	0.88	0.78	0.84	—
confocal_3D_unet_ovules_ds3x	0.96	0.97	0.88	—	0.84	—
lightsheet_3D_unet_root_ds1x	0.96	0.97	0.88	—	0.84	—
lightsheet_3D_unet_root_ds2x	0.96	0.97	0.88	—	0.84	—
lightsheet_3D_unet_root_ds3x	0.96	0.97	0.88	—	0.84	—
lightsheet_3D_unet_root_nuclei_ds1x	0.96	0.97	0.88	—	0.84	—
confocal_3D_unet_sa_meristem_cells	0.96	0.97	0.88	—	0.84	—
confocal_3D_unet_mouse_embryo_nuclei	0.96	0.97	0.88	—	0.84	—
PlantSeg_3Dnuc_platinum	0.96	0.97	0.88	—	0.84	—

Chapter 3

Training a Supervised Cilia Segmentation Model from Self-Supervision

3.1 Introduction

Cilia are hair-like membranes that extend out from the surface of the cells and are present on a variety of cell types such as lungs and brain ventricles and can be found in the majority of vertebrate cells. Categorized into motile and primary, motile cilia can help the cell to propel, move the flow of fluid, or fulfill sensory functions, while primary cilia act as signal receivers, translating extracellular signals into cellular responses [17]. Ciliopathies is the term commonly used to describe diseases caused by ciliary dysfunction. These disorders can result in serious issues such as blindness, neurodevelopmental defects, or obesity [14]. Motile cilia beat in a coordinated manner with a specific frequency and pattern [32]. Stationary, dyskinetic, or slow ciliary beating indicates ciliary defects. Ciliary beating is a fundamental biological process that is essential for the proper functioning of various organs, which makes understanding the ciliary phenotypes a crucial step towards understanding ciliopathies and the conditions stemming from it [69].

Identifying and categorizing the motion of cilia is an essential step towards understanding ciliopathies. However, this is generally an expert-intensive process. Studies have proposed methods that automate the ciliary motion assessment [70]. These methods rely on large amounts of labeled data that are annotated manually which is a costly, time-consuming, and error-prone task. Consequently, a significant bottleneck to automating cilia analysis is a lack of automated segmentation. Segmentation has remained a bottleneck of the pipeline due to the poor performance of even state-of-the-art models on some datasets. These datasets tend to exhibit significant spatial artifacts (light diffraction, out-of-focus cells, etc.) which confuse traditional image segmentation models [38].

Video segmentation techniques tend to be more robust to such noise, but still struggle due to the wild inconsistencies in cilia behavior: while healthy cilia have regular and predictable movements, unhealthy cilia display a wide range of motion, including a lack of motion altogether [23]. This lack of motion

especially confounds movement-based methods which otherwise have no way of discerning the cilia from other non-cilia parts of the video. Both image and video segmentation techniques tend to require expert-labeled ground truth segmentation masks. Image segmentation requires the masks in order to effectively train neural segmentation models to recognize cilia, rather than other spurious textures. Video segmentation, by contrast, requires these masks in order to properly recognize both healthy and diseased cilia as a single cilia category, especially when the cilia show no movement.

To address this challenge, we propose a two-stage image segmentation model designed to obviate the need for expert-drawn masks. We first build a corpus of segmentation masks based on optical flow (OF) thresholding over a subset of healthy training data with guaranteed motility. We then train a semi-supervised neural segmentation model to identify both motile and immotile data as a single segmentation category, using the flow-generated masks as "pseudolabels". These pseudolabels operate as "ground truth" for the model while acknowledging the intrinsic uncertainty of the labels. The fact that motile and immotile cilia tend to be visually similar in snapshot allows us to generalize the domain of the model from motile cilia to all cilia. Combining these stages results in a semi-supervised framework that does not rely on any expert-drawn ground-truth segmentation masks, paving the way for full automation of a general cilia analysis pipeline.

3.2 Background

Dysfunction in ciliary motion indicates diseases known as ciliopathies, which can disrupt the functionality of critical organs like the lungs and kidneys. Understanding ciliary motion is crucial for diagnosing and understanding these conditions. The development of diagnosis and treatment requires the measurement of different cell properties including size, shape, and motility [56].

Accurate analysis of ciliary motion is essential but challenging due to the limitations of manual analysis, which is /labor-intensive, subjective, and prone to error. [70] proposed a modular generative pipeline that automates ciliary motion analysis by segmenting, representing, and modeling the dynamic behavior of cilia, thereby reducing the need for expert intervention and improving diagnostic consistency. [45] developed a computational pipeline using dynamic texture analysis and machine learning to objectively and quantitatively assess ciliary motion, achieving over 90% classification accuracy in identifying abnormal ciliary motion associated with diseases like primary ciliary dyskinesia (PCD). Additionally, [69] explored advanced feature extraction techniques like Zero-phase PCA Sphering (ZCA) and Sparse Autoencoders (SAE) to enhance cilia segmentation accuracy. These methods address challenges posed by noisy, partially occluded, and out-of-phase imagery, ultimately improving the overall performance of ciliary motion analysis pipelines. Collectively, these approaches aim to enhance diagnostic accuracy and efficiency, making ciliary motion analysis more accessible and reliable, thereby improving patient outcomes through early and accurate detection of ciliopathies. However, these studies rely on manually labeled data. The segmentation masks and ground-truth annotations, which are essential for training the models and validating their performance, are generated by expert reviewers. This dependence on manually labeled data is a significant limitation making automated cilia segmentation the bottleneck to automating cilia analysis.

In the biomedical field, where labeled data is often scarce and costly to obtain, several solutions have been proposed to augment and utilize available data effectively. These include semi-supervised learning [66], [59], which utilizes both labeled and unlabeled data to enhance learning accuracy by leveraging the data's underlying distribution. Active learning [51] focuses on selectively querying the most informative data points for expert labeling, optimizing the training process by using the most valuable examples. Data augmentation techniques [3], [31], [67], [49], [66], [58], [30], [47], such as image transformations and synthetic data generation through Generative Adversarial Networks [13], [68], increase the diversity and volume of training data, enhancing model robustness and reducing overfitting. Transfer learning [66], [50], [46], [18] transfers knowledge from one task to another, minimizing the need for extensive labeled data in new tasks. Self-supervised learning [25], [29], [40] creates its labels by defining a pretext task, like predicting the position of a randomly cropped image patch, aiding in the learning of useful data representations. Additionally, few-shot, one-shot, and zero-shot learning techniques [34], [42] are designed to operate with minimal or no labeled examples, relying on generalization capabilities or metadata for making predictions about unseen classes.

A promising approach to overcome the dependency on manually labeled data is the use of unsupervised methods to generate ground truth masks. Unsupervised methods do not require prior knowledge of the data [24]. Using domain-specific cues unsupervised learning techniques can automatically discover patterns and structures in the data without the need for labeled examples, potentially simplifying the process of generating accurate segmentation masks for cilia. Inspired by advances in unsupervised methods for image segmentation, in this work, we firstly compute the motion vectors using optical flow of the ciliary regions and then apply autoregressive modelling to capture their temporal dynamics. Autoregressive modelling is advantageous since the labels are features themselves. By analyzing the OF vectors, we can identify the characteristic motion of cilia, which allows us to generate pseudolabels as ground truth segmentation masks. These pseudolabels are then used to train a robust semi-supervised neural network, enabling accurate and automated segmentation of both motile and immotile cilia.

3.3 Methodology

Dynamic textures, such as sea waves, smoke, and foliage, are sequences of images of moving scenes that exhibit certain stationarity properties in time [8]. Similarly, ciliary motion can be considered as dynamic textures for their orderly rhythmic beating. Taking advantage of this temporal regularity in ciliary motion, OF can be used to compute the flow vectors of each pixel of high-speed videos of cilia. In conjunction with OF, autoregressive (AR) parameterization of the OF property of the video yields a manifold that quantifies the characteristic motion in the cilia. The low dimension of this manifold contains the majority of variations within the data, which can then be used to segment the motile ciliary regions.

3.4 Optical Flow Properties

Taking advantage of this temporal regularity in ciliary motion, we use OF to capture the motion vectors of ciliary regions in high-speed videos. OF provides the horizontal u and vertical v components of the motion for each pixel. From these motion vectors, several components can be derived such as the magnitude, direction, divergence, and importantly, the curl (rotation). The curl, in this context, represents the rotational motion of the cilia, which is indicative of their rhythmic beating patterns. We extract flow vectors of the video recording of cilia, under the assumption that pixel intensity remains constant throughout the video.

$$I(x, y, t) = I(x + u\delta t, y + v\delta t, t + \delta t) \tag{3.4.1}$$

(3.4.1) Where $I_{x,y,t}$ is the pixel intensity at position x, y a time t. Here, u_t, v_t are small changes in the next frame taken after t time, and u, v, respectively, are the OF components that represent the displacement in pixel positions between consecutive frames in the horizontal and vertical directions at pixel location x, y.

3.5 Autoregressive Modeling

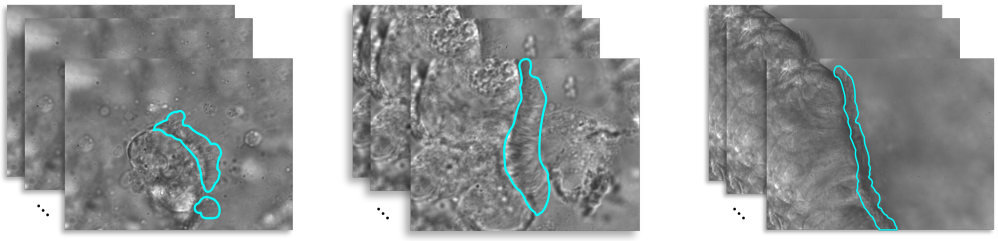

Figure 3.1: A sample of three videos in our cilia dataset with their manually annotated ground truth masks.

Figure 3.1 shows a sample of the OF component at a random time. From OF vectors, elemental components such as rotation are derived, which highlights the ciliary motion by capturing twisting and turning movements. To model the temporal evolution of these motion vectors, we employ an autoregressive (AR) model **doi:10.5244/C.21.76**. This model captures the dynamics of the flow vectors over time, allowing us to understand how the motion evolves frame by frame. The AR model helps in decomposing the motion into a low-dimensional subspace, which simplifies the complex ciliary motion into more manageable analyses.

$$y_t = C\vec{x_t} + \vec{u} \tag{3.5.1}$$

$$\vec{x_t} = A_1\vec{x}_{t-1} + A_2\vec{x}_{t-2} + ... + A_d\vec{x}_{t-d} + \vec{v_t} \tag{3.5.2}$$

In equation (3.5.1), y_t represents the appearance of cilia at time t influenced by noise u. Equation (3.5.2) represents the state x of the ciliary motion in a low-dimensional subspace defined by an orthogonal basis C at time t, plus a noise term v_t and how the state changes from t to $t + 1$.

Equation (3.5.2) is a decomposition of each frame of a ciliary motion video yt into a low-dimensional state vector x_t using an orthogonal basis C. This equation at position x_t is a function of the sum of d of its previous positions $x_{t-1}, x_{t-2}, x_{t-d}$ each multiplied by its corresponding coefficients $A = A_1, A_2, ..., A_d$. The noise terms u and v are used to represent the residual difference between the observed data and the solutions to the linear equations. The variance in the data is predominantly captured by a few dimensions of C, simplifying the complex motion into manageable analyses.

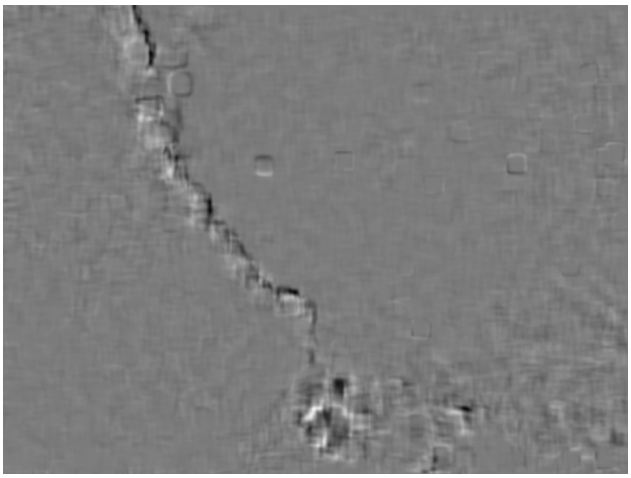

Figure 3.2: Representation of rotation (curl) component of OF at a random time.

Each order of the autoregressive model roughly aligns with different frequencies within the data, therefore, in our experiments, we chose d=5 as the order of our autoregressive model. This choice allows us to capture a broader temporal context, providing a more comprehensive understanding of the system's dynamics. We then created raw masks from this lower-dimensional subspace, and further enhanced them with adaptive thresholding to remove the remaining noise.

In 3.2 , the first-order AR parameter is showing the most variance in the video, which corresponds to the frequency of motion that cilia exhibit. The remaining orders have correspondence with other different frequencies in the data caused by, for instance, camera shaking. Evidently, simply thresholding the first-order AR parameter is adequate to produce an accurate mask, however, in order to get a more refined result we subtracted the second order from the first one, followed by a Min-Max normalization of pixel intensities and scaling to an 8-bit unsigned integer range. We used adaptive thresholding to extract the mask on all videos of our dataset. The generated masks exhibited under-segmentation in the ciliary region, and sparse over-segmentation in other regions of the image. To overcome this, we adapted a Gaussian blur filter followed by an Otsu thresholding to restore the under-segmentation and remove the sparse over-segmentation. Figure 3.4 illustrates the steps of the process.

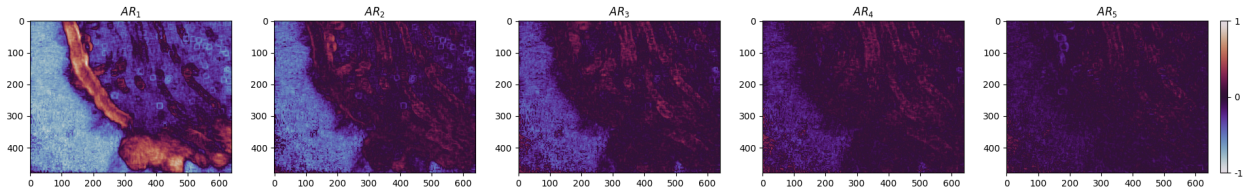

Figure 3.3: The pixel representation of the 5-order AR model of the OF component of a sample video. The x and y axes correspond to the width and height of the video.

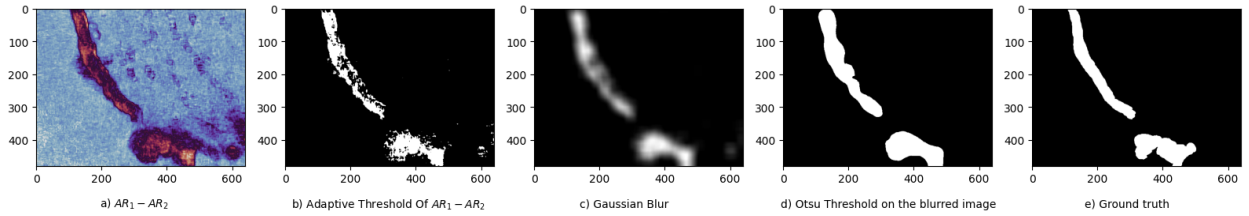

Figure 3.4: The process of computing the masks. **a)** Subtracting the second-order AR parameter from the first-order, followed by **b)** Adaptive thresholding, which suffers from under/over-segmentation. **c)** A Gaussian blur filter, followed by **d)** An Otsu thresholding eliminates the under/over-segmentation.

3.6 Training the model

Our dataset includes 512 videos, with 437 videos of dyskinetic cilia and 75 videos of healthy motile cilia, referred to as the control group. The control group is split into %85 and %15 for training and validation respectively. 108 videos in the dyskinetic group are manually annotated which are used in the testing step. Figure 3.1 shows annotated samples of our dataset.

In our study, we employed a Feature Pyramid Network (FPN) [27] architecture with a ResNet-34 encoder. The model was configured to handle grayscale images with a single input channel and produce binary segmentation masks. For the training input, one mask is generated per video using our methodology, and we use the first 250 frames from each video in the control group making a total of 18,750 input images. We utilized Binary Cross-Entropy Loss for training and the Adam optimizer with a learning rate of 10^{-3}. To evaluate the model's performance, we calculated the Dice score during training and validation. Data augmentation techniques, including resizing, random cropping, and rotation, were applied to enhance the model's generalization capability. The implementation was done using a library [19] based on PyTorch Lightning to facilitate efficient training and evaluation. Table 3.1 contains a summary of the model parameters and specifications.

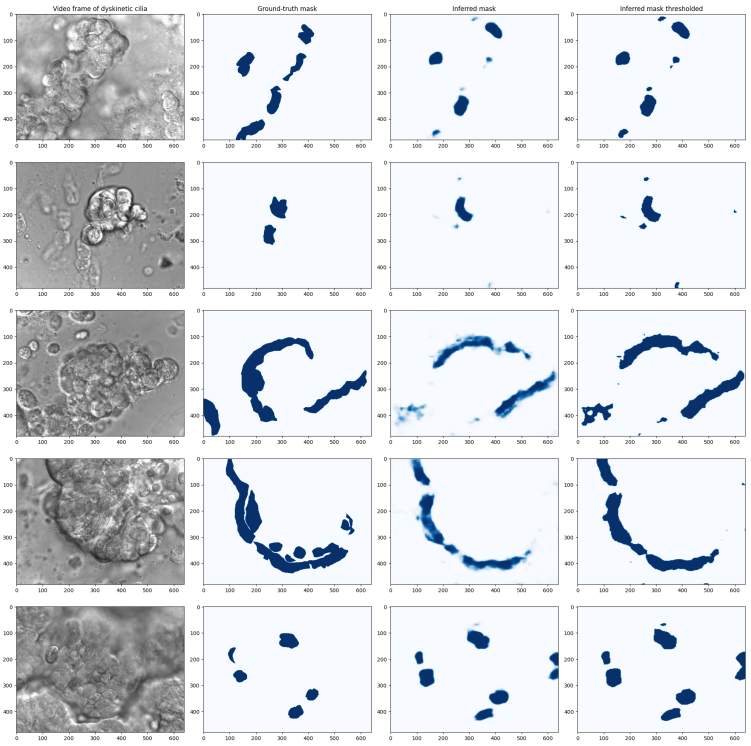

Figure 3.5: The model predictions on 5 dyskinetic cilia samples. The first column shows a frame of the video, the second column shows the manually labeled ground truth, the third column is the model's prediction, and the last column is a thresholded version of the prediction.

3.7 Results and Discussion

The model's performance metrics, including IoU, Dice score, sensitivity, and specificity, are summarized in @tbl:metrics. The validation phase achieved an IoU of 0.312 and a Dice score of 0.476, which indicates a moderate overlap between the predicted and ground truth masks. The high sensitivity (0.999) observed during validation suggests that the model is proficient in identifying ciliary regions, albeit with a specificity of 0.813, indicating some degree of false positives. In the testing phase, the IoU and Dice scores decreased to 0.230 and 0.374, respectively, reflecting the challenges posed by the dyskinetic cilia data, which were not included in the training or validation sets. Despite this, the model maintained a reasonable sensitivity of 0.631 and specificity of 0.787.

Figure 3.5 provides visual examples of the model's predictions on dyskinetic cilia samples, alongside the manually labeled ground truth and thresholded predictions. The dyskinetic samples were not used in the training or validation phases. These predictions were generated after only 20 epochs of training with a small training data. The visual comparison reveals that, while the model captures the general structure of ciliary regions, there are instances of under-segmentation and over-segmentation, which are

Table 3.1: Summary of model architecture, training setup, and dataset distribution

Aspect	**Details**
Architecture	FPN with ResNet-34 encoder
Input	Grayscale images with a single input channel
Number of Epochs	20
Batch Size	4
Training Samples	15,662
Validation Samples	2,763
Test Samples	108
Loss Function	Binary Cross-Entropy Loss
Optimizer	Adam optimizer with a learning rate of 10^{-3}
Evaluation Metric	Dice score during training and validation
Data Augmentation Techniques	Resizing, random cropping, and rotation
Implementation	Using a Python library with Neural Networks for Image Segmentation based on PyTorch $Iakubovskii$: 2019

more pronounced in the dyskinetic samples. This observation is consistent with the quantitative metrics, suggesting that further refinement of the pseudolabel generation process or model architecture could enhance segmentation accuracy.

Table 3.2: The performance of the model in validation and testing phases.

Phases	**IoU over dataset**	**Dice Score**	**Sensitivity**	**Specificity**
Validation	0.312	0.476	0.999	0.813
Testing	0.230	0.374	0.631	0.787

These results show the potential of our approach to reduce the reliance on manually labeled data for cilia segmentation. The use of this unsupervised learning framework allows the model to generalize from the motile cilia domain to the more variable dyskinetic cilia, although with some limitations in accuracy. Future work could focus on expanding the dataset and improving the process of generating pseudolabels to enhance the model's accuracy.

3.8 Conclusions and Final Remarks

In this study, we introduced a self-supervised framework for cilia segmentation that eliminates the need for expert-labeled ground truth segmentation masks. Our approach takes advantage of the inherent vi-

sual similarities between healthy and unhealthy cilia to generate pseudolabels from optical flow-based motion segmentation of motile cilia. These pseudolabels are then used as ground truth for training a semi-supervised neural network capable of identifying regions containing dyskinetic cilia. Our results indicate that the self-supervised framework is a promising step towards automated cilia analysis. The model's ability to generalize from motile to dyskinetic cilia demonstrates its potential applicability in clinical settings. Although there are areas for improvement, such as enhancing segmentation accuracy and expanding the dataset, the framework sets the foundation for more efficient and reliable cilia analysis pipelines.

Chapter 4

Minimally-Supervised Biomedical image Segmentation via Contrastive Learning

4.1 Introduction

Image segmentation is a fundamental process in many computer vision applications and is used to partition the image into separate regions. It is an essential part in various biomedical applications, including lesion and tumor detection and analysis, organ localization and identification, diagnosis and monitoring, and cell and tissue analysis. Similarly, image segmentation is a cornerstone of quantitative cell research, particularly for studying cellular dynamics like motility [56] and morphological changes. Given its critical role, it has been the focus of extensive research, with ongoing advancements aimed at improving accuracy, automation, and generalization across diverse imaging modalities.

Biomedical images come in a vast variety of formats, types, and modalities [37], [60], [71]. Similarly, due to the variety of biological structures, segmentation targets can vary from nuclei and cell membranes to organelles such as mitochondria, cilia, tumors, and lesions, as well as blood vessels, bone, and brain structures [57]. Deep learning (DL) has advanced the field of image segmentation, particularly with the success of convolutional neural networks (CNN) [63]. While CNNs revolutionized segmentation for their high accuracy, due to the large diversity in biomedical image modalities, formats, and structures as well as the scarcity of ground truth data, CNNs are tailored for specific tasks [39] in biomedical image segmentation and therefore suffer from overfitting and exhibit poor generalizability over unseen data. Furthermore, their specificity to tasks, high computational demands, and complex implementation limit their broader application.

Inspired by Large Language Models (LLMs), Foundation Models such as the Segment Anything Model (SAM) [28] demonstrate excellent zero-shot segmentation performance across a large variety of general images. Studies that build upon SAM [26] have shown promising zero-shot learning capabilities and can segment objects in biomedical images regardless of their modality. However, when applied

to biomedical data without fine-tuning, SAM often struggles to match the accuracy of domain-specific models like U-Net. Its zero-shot performance varies significantly across medical datasets and tasks, highlighting the need for fine-tuning to adapt it effectively for biomedical image segmentation. Furthermore, although SAM excels at segmenting objects with well-defined, envelope or convex geometries, it struggles with biological structures that exhibit diffuse or punctate patterns such as cilia, which are even difficult to generate hand-drawn labels for.

Unsupervised methods, on the other hand, are used in scenarios where domain-specific cues suffice for crafting an algorithm for segmentation and when obtaining ground truth data is costly [57]. However, since they are domain-specific, unsupervised methods also exhibit poor generalizability. Self-supervised learning (SSL) is also a promising direction in unsupervised segmentation. Contrastive learning (CL) is a successful variant of SSL and refers to a type of learning where the goal is to learn representations by contrasting positive pairs (similar or related data points) against negative pairs (dissimilar or unrelated data points). This approach is widely used in self-supervised learning where labels are not available. Contrastive coding (CC), often seen as a subset or a specific implementation of CL, refers more specifically to the encoding process where contrastive loss functions are used to train models to produce these discriminative embeddings.

Contrastive learning provides an alternative approach to segmentation by leveraging similarities and differences in the data rather than relying on explicit labels. For addressing all the aforementioned issues, we turned to contrastive coding to teach the network to recognize objects of the same texture and configuration. By learning representations that cluster visually similar structures together while separating dissimilar ones, contrastive learning enables segmentation with minimal user interaction. This makes it particularly suitable for biomedical image analysis, where labeled data is scarce, and manual annotations are costly and time-consuming. The code to our method is available at `https://github.com/quinngroup/contrastive-coding`

The rest of this paper is structured as follows. Section 4.2 describes our proposed contrastive learning approach for biomedical image segmentation, detailing the network architecture and training procedure, and presents the experimental setup and datasets used to evaluate our method. Section 4.3.5 discusses the segmentation performance and compares it with existing approaches. Finally, Section 4.4 concludes the paper with a summary of findings and potential directions for future research.

4.2 Background

Image segmentation is a crucial topic in computer vision and in particular deep learning. In image segmentation, an input image is broken down into its mutually exclusive semantic constituents such as the independent objects and the background. To address the first aforementioned shortcoming, Hyunseob et al. proposed a model called MDNet [43]. MDNet, or Multi Domain Network, is a supervised tracking method that learns domain-independent representations from pre-training. In supervised learning, a set including different objects which are semantically similar together, such as "pedestrian", "ball", "car", or "flower," are used for training. The most important drawback of MDNet, and any supervised segmen-

tation framework like Mask- and Cascade-RCNN [1], [16] SSD [35], is that it relies on large amounts of labeled data for training from various objects, while after training the model with such a big dataset, there is still no guarantee that it can detect any other objects. There always exists sets of objects which are not used for training, and as a consequence, the network may not detect those types of objects properly.

Some unsupervised segmentation methods were proposed recently [7], [20], [36], [65], the most popular of which is W-Net [65]. However, since W-Net has to reconstruct the entire image again from the segmentation map, background and other objects of no interest have to be present in the segmentation mask, increasing the burden on the network to perfectly segment them, while they are of no interest. Indeed, we first started our experimentation to try to extend the work in W-Net, but we found it relied heavily on the final Conditional Random Field (CRF) module to fix the background creeping, and we could not distill the objects of interest alone with the segmentation mask. Also, in [7], an unsupervised segmentation method was proposed for separating the background from foreground using deep learning, and again, distilling the object of interest alone is still an issue in this work, since sometimes the object of interest is visually closer to the background than to the foreground, as in some of our data that we present later.

For addressing all the aforementioned issues we turned to contrastive coding in order to teach the network to recognize objects of the same texture and con guration. In contrastive coding, the goal is to represent instances (images/videos/patches) with vectors, and have instances that are similar attract and instances that are dissimilar repel each other. This is typically done with dot product or cosine similarity on the learnt vectors. There has been a lot of recent work in unsuper vised contrastive learning [6], [15], [44], such as SimCLR[4], [5], where patches from the same image are made to attract each other, while patches from different images are made to repel each other. To aid with better object recognition, the patches are transformed with the usual image augmentation techniques like color jittering, blurring, flipping, and rotation.

4.3 Methodology

4.3.1 Network architecture

The network is designed to take in a patch of dimensions $i \times k \times k$ and output a vector of size d to represent this patch in the dot product operations. We achieve this with a network constructed as follows: Three MBConv layers each outputting 32 channels, followed by a max pooling layer that downsizes the image by half, then another 3 MBConv layers each outputting 64 channels, followed by a global max pooling layer downsizing the image to $d \times 1 \times 1$ followed by a fully connected layer that outputs another d-dimensional vector with a final activation function of $tanh$. Figure 4.1(a) illustrates the network architecture and the application of the network to the current and next video frames. An MBConv layer is adapted from EfficientNet[54]. Figure 4.1(b) shows the internals of an MBConv layer.

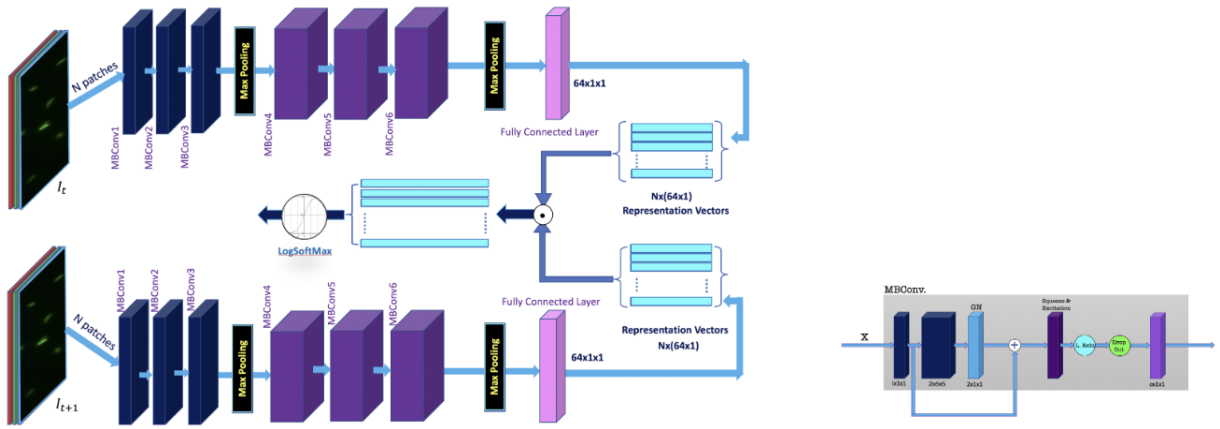

Figure 4.1: (a) The architecture of our contrastive network applied to two consecutive frames. (b) The internals of an MBConv layer.

4.3.2 Contrastive training

Let $x_1, x_2, \ldots, x_N$ be patches of size $i \times k \times k$ pixels from an input image I_t. We aim to represent each patch with a vector representation of size d. The vector representation of a patch x_i is obtained using a convolutional neural network, whose final output layer produces a d-dimensional vector, i.e., $v_i = f_\theta(x_i)$, where θ represents the parameters of the neural network f.

The goal of contrastive learning is to bring similar vectors closer together while pushing dissimilar ones farther apart. To achieve this, we need to sample vectors that should be similar and others that should be dissimilar. We use the observation that our videos are Nyquist sampled, i.e., the sampling rate in our videos is high relative to the frequency of the recorded motion. This implies that consecutive frames differ only slightly in content. Therefore, a patch x_i from the same location in two consecutive frames I_t and I_{t+1} will most likely be similar, and this forms the basis for sampling positive examples.

For negative examples, however, we sample random patches from both the current frame and the next frame. Even though these random patches might contain objects visually similar to the current patch, we assume that the corresponding patch from the next frame will be the most similar to the current patch and should thus be coupled positively above any other pairing. We set a ratio of $m : 1$ for negative to positive samples to contrast with the vectors from the current frame.

4.3.3 Datasets

To evaluate the performance of our proposed segmentation method, we utilize a diverse set of biomedical video datasets. By incorporating datasets with a wide range of cell shapes, sizes, and motility patterns,

Table 4.1: Cell Tracking Challenge (CTC) 2D Datasets

Dataset Name	**Modality**	**Cell Type**
BF-C2DL-HSC	Brightfield (BF)	Mouse hematopoietic stem cells
BF-C2DL-MuSC	Brightfield (BF)	Mouse muscle stem cells
DIC-C2DH-HeLa	DIC	HeLa cells on a flat glass
Fluo-C2DL-Huh7	Fluorescence (Fluo)	Human hepatocarcinoma-derived cells
Fluo-C2DL-MSC	Fluorescence (Fluo)	Rat mesenchymal stem cells
Fluo-N2DH-GOWT1	Fluorescence (Fluo)	GFP-GOWT1 mouse stem cells
Fluo-N2DL-HeLa	Fluorescence (Fluo)	HeLa cells expressing H2b-GFP
PhC-C2DH-U373	Phase Contrast (PhC)	Glioblastoma-astrocytoma U373 cells
PhC-C2DL-PSC	Phase Contrast (PhC)	Pancreatic stem cells
Fluo-N2DH-SIM+	Fluorescence (Fluo)	Simulated nuclei of HL60 cells

we aim to assess the generalizability of our method across different biological structures and imaging conditions. By applying our method to these datasets, we also aim to evaluate its ability to handle complex, diffuse, or punctate patterns.

Cell Tracking Challenge Datasets

We use all available 2D datasets from the Cell Tracking Challenge (CTC) [41]. These datasets include various cell types and imaging modalities, such as fluorescence and phase-contrast microscopy images. They cover a range of biological structures and provide a diverse testbed for evaluating the performance of segmentation methods across different imaging conditions.

4.3.4 Training process

Each iteration of the training, we construct the matrix $R^{n\times d}$ which is the set of patches of an image I_t after passing them through the representation network where column i represents $v_i = f_\theta(x_i)$. To represent the similarity with all the negative and positive samples, we construct the matrix $M^{n\times(m+1)}$ where each column is the dot product between the matrix R with a matrix $Q^{n\times d}$ of random patches sampled randomly from the $2N$ available patches at hand from the current I_t and next I_{t+1} frames, except for the last column, the column of positive patches, where the matrix Q is set to be the vectors of the next patches of the matrix R from the next frame I_{t+1}. We also transform the next-frame positive patches by flipping them horizontally and vertically each with probability 0.5. This is so that the network learns to associate the same texture in different positions and configurations.

4.3.5 Similarity and loss

We choose the cosine similarity between vectors as our similarity metric. The vector output of the convolutional network is, therefore, projected onto the $L2$ unit sphere (i.e., normalized), before being used with dot products. For practical numerical stability, though, we use logSoftmax with negative log-likelihood instead of softmax and cross-entropy.

4.4 Results and Discussion

We evaluate our method on a subset of 2D datasets from the CTC. The CTC offers a diverse array of 2D and 3D time-lapse microscopy datasets, each capturing unique biological specimens under various imaging modalities. Table 4.1 contains an overview of these datasets, detailing the organisms studied, imaging techniques employed, and acquisition specifics.

For each dataset, or part of dataset, we leave out 20% of the data as a testing portion, and of the remaining 80%, we take 70% of it for training, and 30% for validation. We use the loss on the validation to choose the best model, and report the dice coefficient using the best trained model on the testing portion. In each iteration we sample 1024 patches within the input image, and construct the matrix with the number of negative samples $m = 9$, and the size of representation vector $d = 64$. As noted before, the contrastive loss only minimizes a lower bound on the error, so the training error of the negative log likelihood loss never goes down to 0. We train to 50 epochs for each part of the dataset and use the Adam optimizer as well with the same $10e^{-}3$ learning rate. To generate masks we take user's input in the form of at least one point indicating the coordinates of the object of interest. These coordinates represent the center of the patch whose representation vector will be the anchor to compare against. We sweep the entire image with patches of size 15×15 and stride of 1, generating a representation vector per each pixel in the image, and report the dot product of these vectors and the anchor vector. We use reflective padding instead of zero padding. Finally, the user can select a suitable threshold to binarize the raw mask into the final mask.

Table 4.2 shows a mix of strong and moderate results for dice coefficients and precision. The BF-C2DL-HSC and BF-C2DL-MuSC datasets exhibit the weakest performance, with Dice scores of 0.341 and 0.261, respectively. This poor performance is due to the brightfield imaging modality, which introduces significant intensity variations and makes boundary segmentation challenging. Additionally, artifacts and shadows within the hydrogel environment likely contribute to false positives and inconsistent mask predictions. The DIC-C2DH-HeLa dataset achieves a moderate Dice score of 0.711, showing a reasonable ability to capture cell structures. However, the fine-grained details of the HeLa cells in differential interference contrast (DIC) imaging pose difficulties in maintaining sharp boundary delineation, leading to a loss in segmentation accuracy.

For the Fluo-C2DL-MSC dataset, the Dice coefficient of 0.591 indicates moderate segmentation quality. The elongated morphology of mesenchymal stem cells complicates boundary definitions, leading to thresholding artifacts. The PhC-C2DH-U373 dataset, with a Dice coefficient of 0.510, shows lower per-

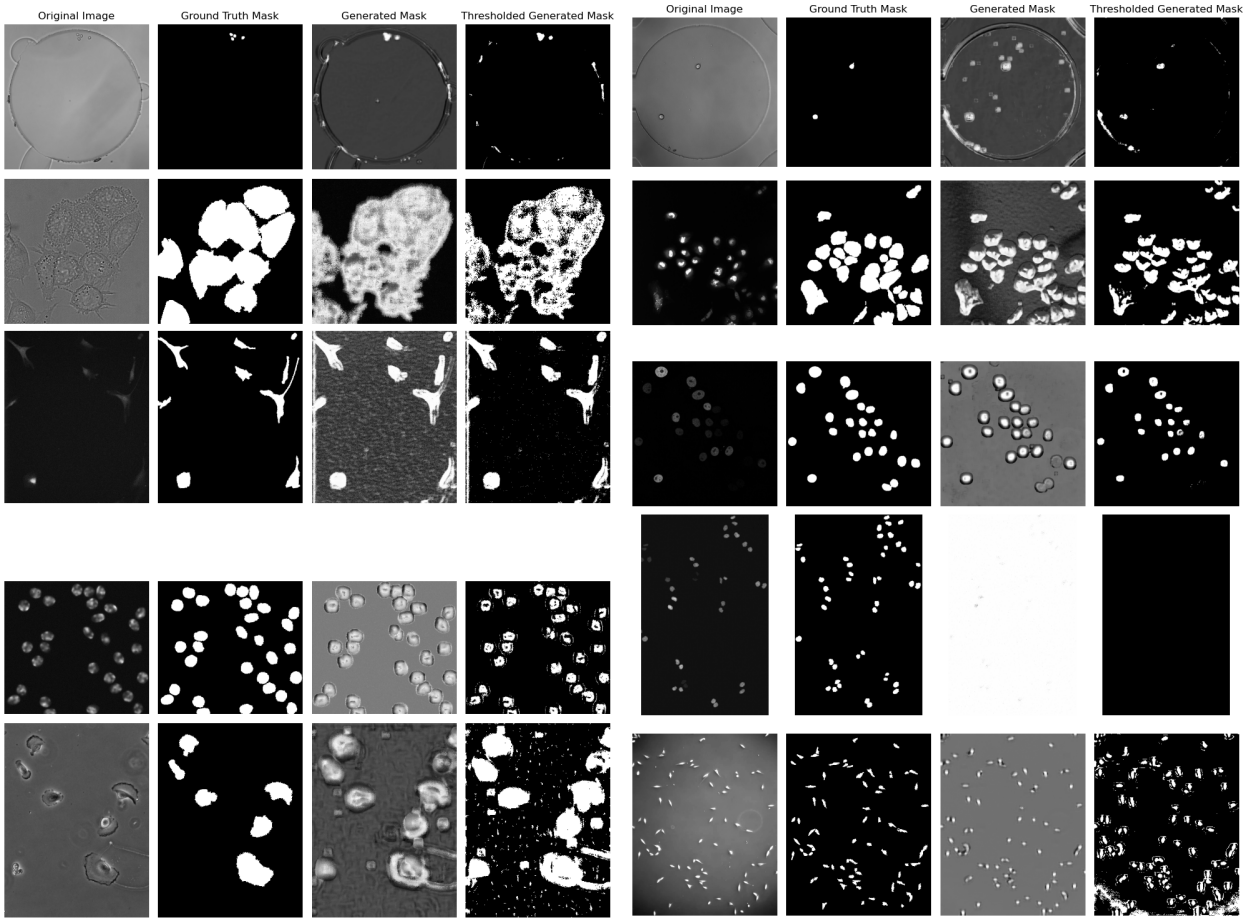

Figure 4.2: Visual comparison of segmentation performance across different datasets. The variation in performance across datasets indicates the challenges caused by different imaging modalities and cell types.

Table 4.2: Dice coefficients and intersection-over-union (IoU) scores for the CTC datasets.

Dataset Name	**IoU**	**Dice**
Fluo-N2DH-GOWT1	0.815	0.8971
Fluo-N2DL-HeLa	0.487	0.655
PhC-C2DL-PSC	0.736	0.847
Fluo-C2DL-Huh7	0.617	0.762
Fluo-N2DH-SIM+	0.786	0.7404
BF-C2DL-HSC	0.206	0.341
BF-C2DL-MuSC	0.15	0.261
DIC-C2DH-HeLa	0.551	0.711
Fluo-C2DL-MSC	0.419	0.591
PhC-C2DH-U373	0.342	0.51

formance due to halo effects in phase contrast imaging, which interfere with precise boundary extraction and introduce noise.

Overall, the results suggest that datasets with clear and well-defined fluorescence-stained boundaries (such as GOWT1) tend to perform best, while datasets relying on phase contrast, brightfield, or DIC imaging suffer from boundary inconsistencies and intensity variations that complicate segmentation.

4.5 Conclusion

We have introduced a method for object segmentation using contrastive coding, requiring minimal user input to select the object of interest. By enforcing similarity between temporally adjacent patches and differentiating dissimilar ones, the model learns embeddings that enable segmentation without the need for labeled datasets. Our approach generates visually plausible masks and demonstrates good results in some datasets, although it achieves only moderate performance in others. We discuss the reasons for these varying results and emphasize the novelty of our method. Additionally, we provide a GUI tool to assist users in marking the object of interest and setting the appropriate threshold for the entire video.

While our results are encouraging, there are several avenues for future work. First, refining the segmentation boundaries through post-processing or boundary-focused contrastive objectives could help address residual errors in challenging datasets. Second, extending this framework to three-dimensional or volumetric time-series data would further increase its applicability to advanced imaging techniques. In conclusion, our contrastive learning-based approach offers a scalable and practical alternative to fully supervised or foundation-model-driven segmentation pipelines, enabling segmentation of diverse biomedical structures with zero annotation.

Chapter 5

Conclusion

This is where your Conclusion will go

Appendix A

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Bibliography

[1] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6154–6162.

[2] L. von Chamier *et al.*, "Democratising deep learning for microscopy with zerocostdl4mic," *Nature communications*, vol. 12, no. 1, pp. 1–18, 2021. DOI: 10.1038/s41467-021-22518-0.

[3] C. Chen *et al.*, "Improving the generalizability of convolutional neural network-based segmentation on cmr images," *Frontiers in Cardiovascular Medicine*, vol. 7, 2020, ISSN: 2297-055X. DOI: https://doi.org/10.3389/fcvm.2020.00105. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fcvm.2020.00105.

[4] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*, PmLR, 2020, pp. 1597–1607.

[5] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. E. Hinton, "Big self-supervised models are strong semi-supervised learners," *Advances in neural information processing systems*, vol. 33, pp. 22 243–22 255, 2020.

[6] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020.

[7] I. Croitoru, S.-V. Bogolin, and M. Leordeanu, "Unsupervised learning of foreground object segmentation," *International Journal of Computer Vision*, vol. 127, pp. 1279–1302, 2019.

[8] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, "Dynamic textures," *International journal of computer vision*, vol. 51, pp. 91–109, 2003. DOI: https://doi.org/10.1023/A:1021669406132.

[9] E. Fazeli *et al.*, "Automated cell tracking using stardist and trackmate," *F1000Research*, vol. 9, 2020. DOI: 10.12688/f1000research.27019.1.

[10] M. S. Fazli, S. A. Velia, S. N. Moreno, and S. Quinn, "Unsupervised discovery of toxoplasma gondii motility phenotypes," in *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, IEEE, 2018, pp. 981–984. DOI: 10.1109/isbi.2018.8363735.

[11] M. S. Fazli, S. A. Vella, S. N. Moreno, G. E. Ward, and S. P. Quinn, "Toward simple & scalable 3d cell tracking," in *2018 IEEE International Conference on Big Data (Big Data)*, IEEE, 2018, pp. 3217–3225. DOI: 10.1109/BigData.2018.8622403.

[12] M. S. Fazli *et al.*, "Lightweight and scalable particle tracking and motion clustering of 3d cell trajectories," in *2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, IEEE, 2019, pp. 412–421. DOI: 10.1109/dsaa.2019.00056.

[13] I. Goodfellow *et al.*, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014. DOI: 10.48550/arXiv.1406.2661.

[14] J. N. Hansen, S. Rassmann, B. Stüven, N. Jurisch-Yaksi, and D. Wachten, "CiliaQ: A simple, open-source software for automated quantification of ciliary morphology and fluorescence in 2d, 3d, and 4D images," *The European Physical Journal E*, vol. 44, no. 2, p. 18, Mar. 2021. DOI: https://doi.org/10.1140/epje/s10189-021-00031-y.

[15] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9729–9738.

[16] K. He, G. Gkioxari, P. Dollár, and R. Girshick, *Mask r-cnn*, 2018. arXiv: 1703.06870 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1703.06870.

[17] S. Hoyer-Fender, "Primary and motile cilia: Their ultrastructure and ciliogenesis," in *Cilia and Nervous System Development and Function*, K. L. Tucker and T. Caspary, Eds. Dordrecht: Springer Netherlands, 2013, pp. 1–53, ISBN: 978-94-007-5808-7. DOI: 10.1007/978-94-007-5808-7_1. [Online]. Available: https://doi.org/10.1007/978-94-007-5808-7_1.

[18] M. L. Hutchinson, E. Antono, B. M. Gibbons, S. Paradiso, J. Ling, and B. Meredig, *Overcoming data scarcity with transfer learning*, 2017. DOI: https://doi.org/10.48550/arXiv.1711.05099. arXiv: 1711.05099 [cs.LG].

[19] P. Iakubovskii, *Segmentation models pytorch*, https://github.com/qubvel/segmentation_models.pytorch, 2019.

[20] X. Ji, J. F. Henriques, and A. Vedaldi, "Invariant information clustering for unsupervised image classification and segmentation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9865–9874.

[21] V. Kapoor and C. Carabaña, "Cell tracking in 3d using deep learning segmentations," in *Python in Science Conference*, 2021, pp. 154–161. DOI: 10.25080/majora-1b6fd038-014.

[22] A. Kar, M. Petit, Y. Refahi, G. Cerutti, C. Godin, and J. Traas, "Assessment of deep learning algorithms for 3d instance segmentation of confocal image datasets," *bioRxiv*, 2021. DOI: 10.1101/2021.06.09.447748. eprint: https://www.biorxiv.org/content/early/2021/06/10/2021.06.09.447748.full.pdf. [Online]. Available: https://www.biorxiv.org/content/early/2021/06/10/2021.06.09.447748.

[23] C. Kempeneers and M. A. Chilvers, "To beat, or not to beat, that is question! the spectrum of ciliopathies," *Pediatric Pulmonology*, vol. 53, no. 8, pp. 1122–1129, 2018. DOI: https://doi.org/10.1002/ppul.24078. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ppul.24078. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ppul.24078.

[24] T. Khatibi, N. Rezaei, L. Ataei Fashtami, and M. Totonchi, "Proposing a novel unsupervised stack ensemble of deep and conventional image segmentation (sedcis) method for localizing vitiligo lesions in skin images," *Skin Research and Technology*, vol. 27, no. 2, pp. 126–137, 2021. DOI: http://dx.doi.org/10.1111/srt.12920.

[25] D. Kim, D. Cho, and I. S. Kweon, "Self-supervised video representation learning with space-time cubic puzzles," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, 2019, pp. 8545–8552. DOI: https://doi.org/10.48550/arXiv.1811.09795.

[26] S. Kim *et al.*, *Medivista: Medical video segmentation via temporal fusion sam adaptation for echocardiography*, 2024. arXiv: 2309.13539 [eess.IV]. [Online]. Available: https://arxiv.org/abs/2309.13539.

[27] A. Kirillov, K. He, R. Girshick, and P. Dollár, "A unified architecture for instance and semantic segmentation," in *Computer Vision and Pattern Recognition Conference*, CVPR, 2017. DOI: https://doi.org/10.48550/arXiv.2112.04603.

[28] A. Kirillov *et al.*, *Segment anything*, 2023. arXiv: 2304.02643 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2304.02643.

[29] A. Kolesnikov, X. Zhai, and L. Beyer, "Revisiting self-supervised visual representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 1920–1929. DOI: https://doi.org/10.48550/arXiv.1901.09005.

[30] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012. DOI: 10.1145/3065386.

[31] J. Krois *et al.*, "Generalizability of deep learning models for dental image analysis," *Scientific Reports*, vol. 11, no. 1, p. 6102, Mar. 2021, ISSN: 2045-2322. DOI: http://dx.doi.org/10.1038/s41598-021-85454-5. [Online]. Available: 10.1038/s41598-021-85454-5.

[32] W. Lee, P. Jayathilake, Z. Tan, D. Le, H. Lee, and B. Khoo, "Muco-ciliary transport: Effect of mucus viscosity, cilia beat frequency and cilia density," *Computers & Fluids*, vol. 49, no. 1, pp. 214–221, 2011. DOI: https://doi.org/10.1016/j.compfluid.2011.05.016.

[33] J. Leung, M. Rould, C. Konradt, C. Hunter, and G. Ward, "Disruption of tgphil1 alters specific parameters of toxoplasma gondii motility measured in a quantitative, three-dimensional live motility assay," *PloS one*, vol. 9, e85763, Jan. 2014. DOI: 10.1371/journal.pone.0085763.

[34] F.-F. Li, R. Fergus, P. Perona, *et al.*, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 28, no. 4, pp. 594–611, 2006. DOI: http://dx.doi.org/10.1109/TPAMI.2006.79.

[35] W. Liu *et al.*, "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, Springer, 2016, pp. 21–37.

[36] X. Liu, Q. Xu, J. Ma, H. Jin, and Y. Zhang, "Mslrr: A unified multiscale low-rank representation for image segmentation," *IEEE transactions on image processing*, vol. 23, no. 5, pp. 2159–2167, 2014.

[37] Y. Liu, S. J. Wagner, and T. Peng, "Multi-modality microscopy image style augmentation for nuclei segmentation," *Journal of Imaging*, vol. 8, no. 3, p. 71, 2022.

[38] C. Lu, M. Marx, M. Zahid, C. W. Lo, C. Chennubhotla, and S. P. Quinn, "Stacked neural networks for end-to-end ciliary motion analysis," *arXiv preprint arXiv:1803.07534*, 2018. DOI: https://doi.org/10.48550/arXiv.1803.07534.

[39] J. Ma and B. Wang, "Towards foundation models of biological image segmentation," *Nature Methods*, vol. 20, no. 7, pp. 953–955, 2023.

[40] A. Mahendran, J. Thewlis, and A. Vedaldi, "Cross pixel optical-flow similarity for self-supervised learning," in *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part V 14*, Springer, 2019, pp. 99–116. DOI: https://doi.org/10.48550/arXiv.1807.05636.

[41] M. Maška *et al.*, "The cell tracking challenge: 10 years of objective benchmarking," *Nature Methods*, vol. 20, no. 7, pp. 1010–1020, 2023.

[42] E. G. Miller, N. E. Matsakis, and P. A. Viola, "Learning from one example through shared densities on transforms," in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, IEEE, vol. 1, 2000, pp. 464–471. DOI: https://doi.org/10.1109/CVPR.2000.855856.

[43] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4293–4302.

[44] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[45] S. P. Quinn, M. J. Zahid, J. R. Durkin, R. J. Francis, C. W. Lo, and S. C. Chennubhotla, "Automated identification of abnormal respiratory ciliary motion in nasal biopsies," *Science translational medicine*, vol. 7, no. 299, 299ra124–299ra124, 2015. DOI: http://dx.doi.org/10.1126/scitranslmed.aaa1233.

[46] M. Raghu, C. Zhang, J. Kleinberg, and S. Bengio, "Transfusion: Understanding transfer learning for medical imaging," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc., 2019. DOI: https://doi.org/10.48550/arXiv.1902.07208. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/file/eb1e78328c46506b46a4ac4a1e378b91-Paper.pdf.

[47] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, Springer, 2015, pp. 234–241. DOI: 10.48550/arXiv.1505.04597.

[48] G. Saadatnia and M. Golkar, "A review on human toxoplasmosis," *Scandinavian journal of infectious diseases*, vol. 44, no. 11, pp. 805–814, 2012. DOI: 10.3109/00365548.2012.693197.

[49] 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," *Scientific Reports*, vol. 9, no. 1, p. 16 884, Nov. 2019, ISSN: 2045-2322. DOI: https://doi.org/10.1016/j.imu.2021.100779.

[50] T. H. Sanford *et al.*, "Data augmentation and transfer learning to improve generalizability of an automated prostate segmentation model," en, *AJR Am J Roentgenol*, vol. 215, no. 6, pp. 1403–1410, Oct. 2020. DOI: http://dx.doi.org/10.2214/AJR.19.22347.

[51] B. Settles, "Active learning literature survey," 2009.

[52] N. Sofroniew *et al.*, *napari: a multi-dimensional image viewer for Python*, version v0.4.16, If you use this software, please cite it using these metadata., May 2022. DOI: 10.5281/zenodo.6598542. [Online]. Available: https://doi.org/10.5281/zenodo.6598542.

[53] C. Stringer, T. Wang, M. Michaelos, and M. Pachitariu, "Cellpose: A generalist algorithm for cellular segmentation," *Nature methods*, vol. 18, no. 1, pp. 100–106, 2021. DOI: 10.1101/2020.02.02.931238.

[54] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*, PMLR, 2019, pp. 6105–6114.

[55] J.-Y. Tinevez *et al.*, "Trackmate: An open and extensible platform for single-particle tracking," *Methods*, vol. 115, pp. 80–90, 2017, Image Processing for Biologists, ISSN: 1046-2023. DOI: https://doi.org/10.1016/j.ymeth.2016.09.016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1046202316303346.

[56] S. A. Vaezi, G. Orlando, M. S. Fazli, G. E. Ward, S. N. Moreno, and S. Quinn, "A novel pipeline for cell instance segmentation, tracking and motility classification of toxoplasma gondii in 3d space.," in *SciPy*, 2022, pp. 60–63. DOI: https://doi.org/10.25080/majora-212e5952-009.

[57] S. A. Vaezi and S. Quinn, "Training a supervised cilia segmentation model from self-supervision," *Proceedings of the 23nd*, 2024.

[58] D. A. Van Dyk and X.-L. Meng, "The art of data augmentation," *Journal of Computational and Graphical Statistics*, vol. 10, no. 1, pp. 1–50, 2001. DOI: http://dx.doi.org/10.1198/10618600152418584.

[59] J. E. Van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine learning*, vol. 109, no. 2, pp. 373–440, 2020. DOI: http://dx.doi.org/10.1007/s10994-019-05855-6.

[60] T. Vicar *et al.*, "Cell segmentation methods for label-free contrast microscopy: Review and comprehensive comparison," *BMC bioinformatics*, vol. 20, pp. 1–25, 2019.

[61] M. Weigert, U. Schmidt, R. Haase, K. Sugawara, and G. Myers, "Star-convex polyhedra for 3d object detection and segmentation in microscopy," in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE, Mar. 2020. DOI: 10.1109/wacv45572.2020.9093435. [Online]. Available: https://doi.org/10.1109%2Fwacv45572.2020.9093435.

[62] C. Wen *et al.*, "3DeeCellTracker, a deep learning-based pipeline for segmenting and tracking cells in 3D time lapse images," *Elife*, vol. 10, Mar. 2021. DOI: 10.7554/eLife.59187. [Online]. Available: https://doi.org/10.7554/eLife.59187.

[63] W. Weng and X. Zhu, "Inet: Convolutional networks for biomedical image segmentation," *Ieee Access*, vol. 9, pp. 16 591–16 603, 2021.

[64] A. Wolny *et al.*, "Accurate and versatile 3d segmentation of plant tissues at cellular resolution," *eLife*, vol. 9, C. S. Hardtke, D. C. Bergmann, D. C. Bergmann, and M. Graeff, Eds., e57613, Jul. 2020, ISSN: 2050-084X. DOI: 10.7554/eLife.57613. [Online]. Available: https://doi.org/10.7554/eLife.57613.

[65] X. Xia and B. Kulis, "W-net: A deep model for fully unsupervised image segmentation," *arXiv preprint arXiv:1711.08506*, 2017.

[66] A. Yakimovich, A. Beaugnon, Y. Huang, and E. Ozkirimli, "Labels in a haystack: Approaches beyond supervised learning in biomedical applications," *Patterns*, vol. 2, no. 12, p. 100 383, 2021, ISSN: 2666-3899. DOI: https://doi.org/10.1016/j.patter.2021.100383. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2666389921002506.

[67] W. Yan *et al.*, "Mri manufacturer shift and adaptation: Increasing the generalizability of deep learning segmentation for mr images acquired with different scanners," *Radiology: Artificial Intelligence*, vol. 2, no. 4, e190195, 2020, PMID: 33937833. DOI: 10.1148/ryai.2020190195. eprint: 10.1148/ryai.2020190195. [Online]. Available: 10.1148/ryai.2020190195.

[68] X. Yi, E. Walia, and P. Babyn, "Generative adversarial network in medical imaging: A review," *Medical image analysis*, vol. 58, p. 101 552, 2019. DOI: https://doi.org/10.48550/arXiv.1809.07294.

[69] M. Zain, E. Miller, S. Quinn, and C. Lo, "Low level feature extraction for cilia segmentation," in *Proceedings of the Python in Science Conference*, 2022. DOI: https://doi.org/10.25080/majora-212e5952-026.

[70] M. Zain *et al.*, "Towards an unsupervised spatiotemporal representation of cilia video using a modular generative pipeline," in *Proceedings of the Python in Science Conference*, 2020. DOI: http://dx.doi.org/10.25080/Majora-342d178e-017.

[71] T. Zhou, S. Ruan, and S. Canu, "A review: Deep learning for medical image segmentation using multi-modality fusion," *Array*, vol. 3, p. 100 004, 2019.