Georg-August-Universität Göttingen
Fakultät für Mathematik und Informatik
Institute of Computer Science

Bachelor's Thesis

Localizing Cells in Phase-Contrast Microscopy Images using Sparse and Noisy Center-Point Annotations

Submitted on April 1, 2024 by

Benjamin Eckhardt
benjamin.eckhardt@stud.uni-goettingen.de
University of Göttingen,
University of Tartu

Reviewed by

Dr. Dmytro Fishman
dmytro.fishman@ut.ee
University of Tartu

Dr. Constantin Pape
constantin.pape@informatik.uni-goettingen.de
University of Göttingen

Abstract

Counting living cells is an important step in many biological research workflows. Our collaborators at the Wellcome Sanger Institute in Cambridge study vital genes in humans via saturation genome editing screening. For this, they need to repeatedly count surviving cells a great number of times. An automated high-throughput method to accurately count living cells would save a lot of time and resources. In this thesis, we developed several deep learning and classical computer vision algorithms to detect and count cells in phase-contrast microscopy images. The acquisition of ground truth annotations for machine learning can be a bottleneck, because it requires expert knowledge and a lot of time. To reduce annotation effort, we focus on counting cells only using point annotations, which are relatively cheap to acquire. Most of the methods we employ suffer from poor image quality and annotation sparsity. However, we develop a promising prototype to localize and count cells by regressing density maps. Through developing methods to automatically count living cells in microscopy images, we contribute to valuable research on the human genome. Our code, data and experimental results are available at github.com/beijn/bachelor-thesis.

Contents

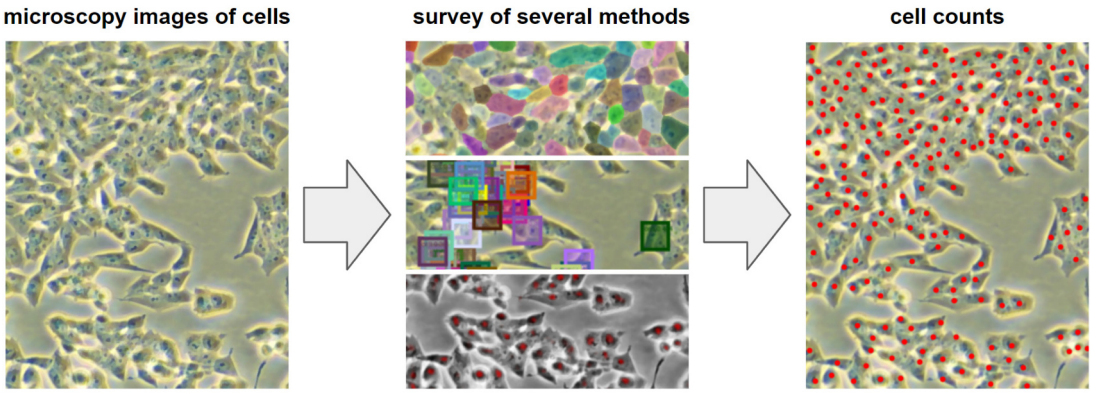

Figure 1: Visual representation of the work performed in this thesis. Phase-contrast microscopy images were acquired by our research partners at Sanger (left panel). We implemented, applied and evaluated several deep learning approaches for detecting cells present in these images (middle panel), such as zero-shot instance segmentation (middle top), object detection (middle center), and density map regression (middle bottom). Those methods were trained using sparse and at times noisy point annotations (right panel).

1 Introduction

Computer vision and in particular object detection using deep learning has been the key for automated high-throughput data analysis in many applications. This thesis applies object detection in, at a first glance, a very unusual area — genome research. Specifically, our collaborators at the Wellcome Sanger Institute in Cambridge aim to map vital genes in humans. To this end, they continuously introduce targeted mutations in the genome of a human cell culture and count the surviving cells' proportion. The great number of counts that has to be performed demands for an automated, high-throughput workflow with low recurring costs. Our collaborators rely on the Countess 3 FL cell counting device by ThermoFisher in an expensive and time-consuming semi-manual process with unacceptable inaccuracies. By developing multiple approaches to count cells using only microscopy images, we minimize material and time expenses and contribute to a completely automated and resource efficient high-throughput workflow for the mapping of vital genes.

Classical image analysis approaches often suffer from susceptibility to small fluctuations in data. Machine learning and in particular deep learning algorithms can learn robust labeling functions, but often require large amounts of high-quality annotations. Producing annotations of sufficient quality takes a lot of work and may therefore be a major bottleneck. Therefore, in this thesis we focus on predicting cell counts in microscopy images using only point annotations at cell centers, which are very cheap to obtain with a single click per instance. The images were obtained with the EVOS physical phase-contrast microscope (ThermoFisher) with digital controls enabling precisely reproducible imaging settings. However, the images are of such low resolution, that even a trained expert could not discern cells in some parts of the images with high confluency. Therefore, while being cheap to obtain, the point labels are sparse and noisy. We partly address the noisiness, but leave the sparsity for future work.

We apply several pre-existing zero-shot methods like Cellpose, Stardist and MicroSAM and show that they fail to provide the desired generalization on our dataset. We fine-tune the well-known object detection model Mask R-CNN on a synthetic task derived from the point labels and report little promising results as well. Lastly, we develop our own model based on regressing density maps derived as a superposition of Gaussian distributions around the point annotations, of which the sum predicts the cell count. This method learns to localize cells well, but the regressed counts need improvement. Briefly we report on other approaches, such as clustering, classical computer-vision filters and Fourier space related transformations with the aim to pronounce cell structures in the images. As a baseline, we develop a cell count estimate based on foreground-background segmentation and an assumed constant cell density in the foreground.

2 Related Work

Counting objects, and in particular counting cells, has already been given a lot of attention [1–25]. Approaches may be categorized as counting by detection and counting by regression [2].

Counting by Detection. Object Detection algorithms aim to localize individual objects within an image. Thereafter, counting of objects becomes trivial. Nowadays, there has been substantial progress in object detection due to availability of new deep learning architectures, big datasets, and greater computational power [3–7]. Some notable approaches include R-CNN [3], Fast R-CNN [4], YOLO [5], RetinaNet [6], and Mask R-CNN [7]. R-CNN relies on Selective Search for region proposals [3]. Fast R-CNN uses an end-to-end training method for improved performance [4]. YOLO streamlines detection using a single CNN for classification and localization [5], while RetinaNet improves upon YOLO with focal loss for better localization performance [6]. Lastly, Mask R-CNN extends the R-CNN framework with segmentation capabilities, enabling instance segmentation and complex object recognition [7].

However, object detection requires ground truth in the form of bounding boxes or instance masks, which are difficult to acquire and were not available to us. We test Mask R-CNN by He et al. [7] with synthetic ground truth derived from point annotations.

Instance Segmentation. Instance Segmentation algorithms localize objects by outlining their precise shape within an image. Similarly to Object Detection, the object count is trivially given by the number of masks. Several instance segmentation methods have been proposed to alleviate the need for ground truth annotations for new image domains. We study the works on zero-shot cellular segmentation on Stardist by Schmidt et al. [8] and Cellpose by Stringer et al. [9] and Pachitariu et al. [10].

By training on very large and diverse segmentation corpora, Kirillov et al.s' Segment Anything (SAM) [11] achieves zero-shot instance detection and segmentation in a great variety of domains. Many have studied SAM's performance in medical imaging [12–15] and in particular cellular segmentation [16–19], such as we do. Archit et al. with Segment Anything for Microscopy (MicroSAM) [19] fine tune SAM for cellular segmentation and develop a model-in-the loop framework for efficient interactive fine-tuning. A different approach to instance segmentation is to predict object boundaries in an image [25]. Instead of one instance mask per instance, the ground truth is a single map with all instance borders marked. Cells can be identified with the enclosures.

Counting by Regression. The seminal work by Lempitsky et al. [2] introduces the idea of counting objects via maxima in an object density map. Such a density map can however also be used to regress counts directly. The great advantage of this method is that it requires only minimal ground truth annotations in the form of one localizing point per cell. Furthermore, due to the construction of the density map as a superposition of Gaussian distributions around the annotated points, the method is especially robust against noisy annotations. Xie et al. [20] apply fully convolutional neural networks (CNN) [26,27] to predict the cell center density map. We implement their ideas with our *CellNet*, but in contrast to Xie et al. [20] we use the U-Net architecture [25] with a ResNet [28] backbone.

Not being aware of the previous work on counting by regression [2,20–22] we developed our own similar approach starting from CenterNet by Zhou et al. [24]. Compared to CenterNet, we superimpose the Gaussians not by taking the point-wise maximum, but by summing them. Also, we assume a fixed object size. Therefore, the sum of our density maps corresponds to the expected number of cells [2,20]. The density map of Zhou et al. is of lower resolution than the image, therefore they need to regress an additional x-y-offset per detected object to locate it precisely. Our density map is of the same resolution as the input, and the location of cells is therefore equal to the local maxima.

3 Background

Object detection has been the key to many areas of computer vision, from obstacle recognition in self-driving cars to cancer detection in medical imaging. This thesis focuses on applying object recognition in, at a first glance, a very unusual area — genome research. More specifically, our collaborators are looking for vital genes in humans. To this end, they continuously introduce mutations in target genes and count surviving cells. Normally, this is done via a pseudo-manual process of operating a dedicated cell counting device. To speed up their workflows, we develop efficient models that can automatically recognize and count cells from phase-contrast microscopy images.

3.1 Practical Application in Human Genome Studies

Estimating the number of living cells is the central step in many biological workflows. Determining the proportion of surviving cells after a treatment can help to determine the safety of a drug or the effectivity of a cancer treatment. In our case, our collaborators at the Wellcome Sanger Institute aim to map all vital genes in the human genome through Saturation Genome Editing (SGE) screening. SGE screening is to introduce all possible single nucleotide variants (SNVs) and clinical variants into a gene of interest and to analyze their effects on cell survival. An SNV is a gene that differs only in a single base pair from the original gene. A clinically relevant variant is a variant of a gene that has been found to be linked with clinical phenotype or a disease, and may be different by a greater number of base-pairs. The project aim is to create variant effect maps for all approximately 20,000 protein encoding human genes. These account for about 1% of the whole human DNA, which contains approximately 3.2 billion base pairs [29]. Therefore, a very large number of screens has to be performed, and each screen can involve multiple cell counting processes. Developing resource and time efficient automations is crucial for the success of this research.

3.2 Expected Improvements upon the Previous Method

Collaborators count cells using a Countess 3 FL Automated Cell Counter (ThermoFisher)[1] and EVE disposable cell counting chambers (NanoEntek)[2] as per the manufacturer's instructions. Their workflow involves a human expert spending several minutes for one sample to prepare it and apply it to the device. The preparation would involve costly single-use reagents consumables. With the great number of counts to perform, these expenses stack up significantly.

The cells need to be dyed with Trypan Blue, which makes them unusable for further studies. The volume of cells that is required for the Countess 3 FL is comparatively small. Yet, it is still desirable for many biological workflows to develop non-invasive methods for counting cells, so that fewer cells have to be grown and sustained. The manual pipetting of small volumes (10µl) of cell suspension and dye is time-consuming and introduces additional inaccuracy. Counting cells using dye-less imaging methods, such as phase-contrast microscopy, is non-invasive. Preparing cell cultures for phase-contrast microscopy can be done much quicker and with no material waste due to fully reusable equipment. Furthermore, it can be integrated easier into a fully automated workflow.

Our collaborators report unacceptable inconsistency of the counts yield by the Countess 3 FL, such that two counts of one and the same sample regularly differ by >10%. In such a case, our collaborators prepare a new sample and recount. The inconsistency of the method therefore leads to repetition of the whole preparation and counting process, which multiplies the overall expenses.

[1] https://www.thermofisher.com/tw/zt/home/life-science/cell-analysis/cell-analysis-instruments/automated-cell-counters/models/countess-3-fl.html [thermofisher.com, retrieved 23.03.2024]

[2] https://www.nanoentek.com/theme/nanont2_en/shop/02/product01_view.php?it_id=1547539043 [nanoentek.com, retrieved 23.03.2024]

Using computer vision, we hope to improve counting accuracy and consistency. A microscope can easily be slid over a large sample surface to gradually increase its effective field of view until a desired robustness against local fluctuations in the cell distribution is reached. Furthermore, we reduce human intervention, such as dye application, as a source of error. Additionally, our microscopy image based computer vision approach reduces the dependence on dedicated hardware, by relying only on common flexible lab-ware such as a microscopes and computers. The software can continuously be improved with new algorithms and new data, without a change to the physical laboratory setup.

3.3 Phase Contrast Microscopy

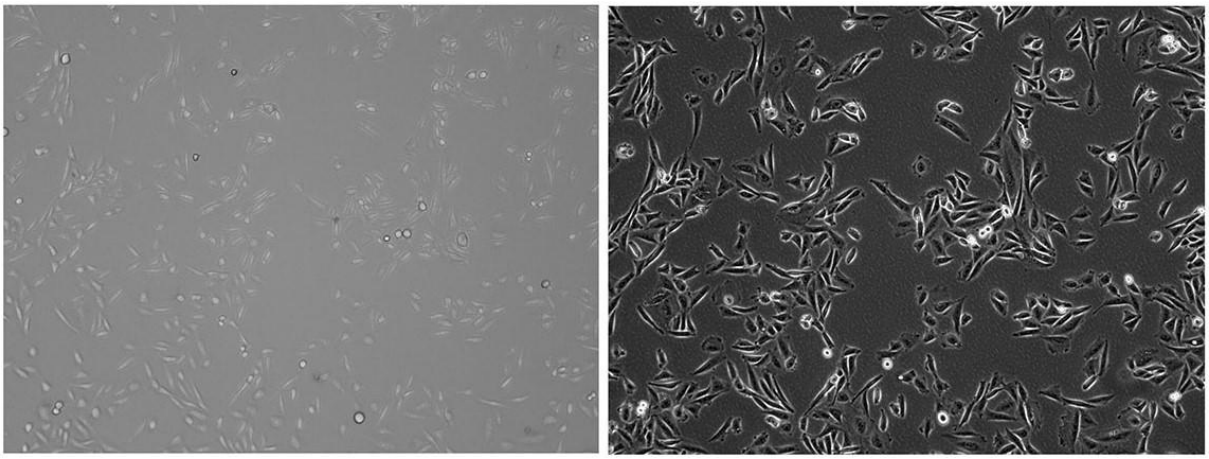

Figure 2: Comparison of bright-field and phase-contrast microscopy images at same magnification. Phase-contrast microscopy offers greater contrast, at cell boundaries, without using dyes.[3]

In comparison to bright-field microscopy, phase-contrast microscopy offers a higher contrast for low-opacity specimen, such as the cells we study (see Figure 2). When light passes through a medium, a part of it gets absorbed or reflected, such that thicker and more opaque objects cast a darker shadow. Bright-field microscopy captures this shadow, such that cells and their membranes appear darker than their surroundings. But individual cells offer very low visual contrast, because they are very small and transparent. Dyes can enhance the contrast greatly. Fluorescence microscopy uses dies and enables imaging of specific cell components, to which the dyes bind, with very high contrast. However, the application of dyes is undesirable in many biological workflows, as it usually makes the samples unusable for further experiments and adds cost.

As light passes through a medium, it refracts according to the material's thickness and refractive index, leading to a small phase shift in the light waves which passed through the material compared to those which passed through a different material. Phase-contrast imaging captures this phase shift, such that very small differences in cell structures can lead to much greater visual contrast than in bright-field microscopy without using dyes.

Many methods have been proposed to compute phase-contrast images digitally from bright-field images, such as reconstructing the phase difference between two or more bright-field with different focal planes [30], or from the chromatic aberration in a single image [31]. The phase-contrast in our images is created physically by interfering light before it reaches the camera sensor [32].

Our images have been obtained using the physical phase-contrast microscope EVOS (ThermoFisher)[4]. See Figure 3, upper left, for an example section. The microscope has digital controls, so that, after vi-

[3]Image sourced from ThermoFisher, https://www.thermofisher.com/tw/zt/home/life-science/cell-analysis/cellular-imaging/cell-imaging-systems/evos-objectives/selection-guide-evos-objectives.html, Figure 4, retrieved 25.04.2024.

[4]ThermoFisher, https://www.thermofisher.com/tw/zt/home/life-science/cell-analysis/cellular-imaging/evos-cell-imaging-systems/models.html, retrieved 24.03.2024

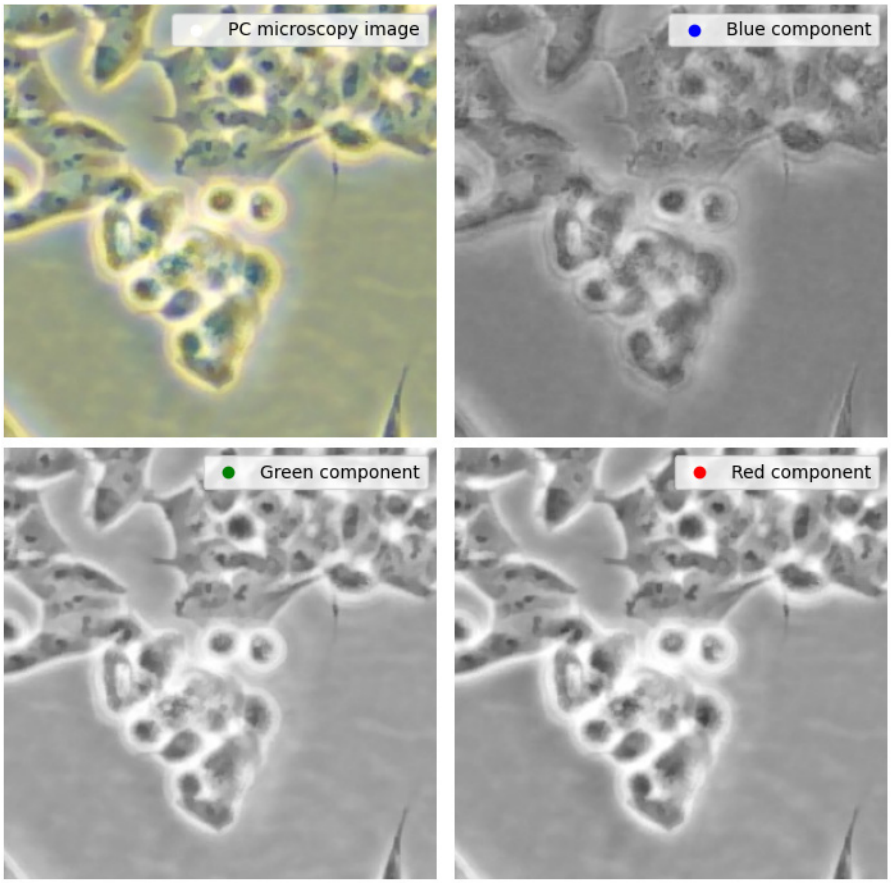

Figure 3: Part of one of our phase-contrast microscopy images. All three imaging channels combined as one RGB image (upper left). The blue channel is very much out of focus (upper right). The green channel (lower left) has the best focus, while the red channel (lower right) is only slightly worse. The images are very blurry and exhibit great phase-contrast imaging artifacts such as halos.

sually optimal settings have been identified by our collaborators, all images can be obtained with the exact same conditions. The microscope records light in three color channels. We observed, that each color channel has a different focal plane. As illustrated in Figure 3, the green color channel has the best focus, red is slightly out of focus, and blue is very much out of focus.

Processing the images was very challenging due to the low resolution of the microscope and the prevalence of halo artifacts, such that many cells could not be annotated, and segmentation models failed to consistently detect boundaries. The collaborating lab was also equipped with a bright-field microscope with higher resolution, but analog controls. We chose against it, in favor of the reproducible imaging settings and greater visual contrast of the EVOS microscope. However, if our work were to be expanded for morphological analysis, a microscope with greater resolution would be necessary to enable confident delineation of individual cells.

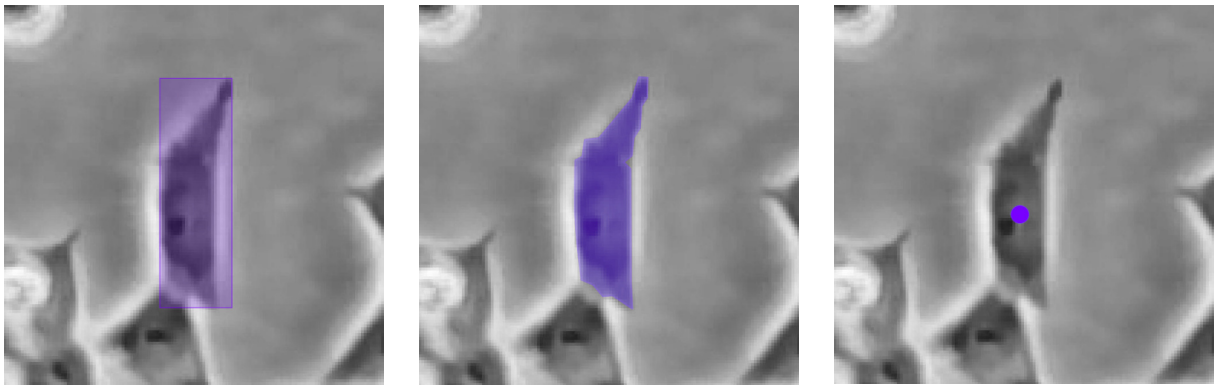

Figure 4: Different computer vision tasks. Object detection (left) typically requires two clicks to determine the minimal fully enclosing bounding box of an object. Instance segmentation (middle) labels each pixel as part of an instance or not. It can be done with brushes or polygons and requires many highly precise clicks. Keypoint detection (right) requires only one click per point of interest. All annotations typically require an additional class label — in our case there is only one class "Cell".

3.4 Deep Learning Computer Vision

Classification, a most important and basic task in computer vision (CV) and in particular deep learning (DL), assigns one class label from a fixed set of classes to a whole image. Classification can be used with a sliding window on a small image patch and several count classes like 1-4, 5-20 cells to count or locate cells [21]. In this thesis, however, we only formulate the problem of localizing cells as Object Detection, Instance Segmentation and Keypoint Detection, as illustrated in Figure 4.

While these are all different possible formulations for localizing cells, they greatly differ in the type and amount of information they need as training targets. From the cheapest (keypoint detection) to the most expensive labels (instance segmentation) the amount of work and domain insight necessary for their creation differs greatly. Therefore, it is crucial to choose the right formulation given the application's demands and annotation availabilities.

3.4.1 Object Detection

The goal in Object Detection, as seen in Figure 4, left, is to predict for every *object* in an image it's position, and commonly also its class, and spatial extension. The maximum number of detectable objects is often limited by the architecture of a model, namely by the dimensions of the feature volumes from which the predictions are drawn.

To detect cells, not only their location, but also their size along the x- and y-axes must be known.

As an evaluation metric, commonly, Average Precision (AP) is used. AP is given by the average of the precision over the recall, given by different confidence thresholds. Mean Average Precision (mAP) generalizes to multi-class problems by taking the mean of the AP over the class.

3.4.2 Instance Segmentation

The semantic segmentation task, as seen in Figure 4, middle, is to attribute each pixel in the target the class of the object it belongs to — the whole prediction is called a class map. Semantic segmentation in its naive form cannot be applied to recognition of individual cells in our case, because the cells are mostly tightly packed in large clusters. A semantic segmentation algorithm would merge all these into just one connected object, thereby the resulting count would be misleading. An approach is to predict an additional class for cell borders. Enclosures of cell border that are filled with cells can be identified as individual cells. We study such an approach in Section 5.3.

A different approach is instance segmentation, which predicts a separate shape for each object, called instance masks. Class maps in semantic segmentation can be expressed as a stack of density maps for each class, whereas the final output can be obtained through an argmax-projection. In instance segmentation, such a density map is predicted for each instance. For evaluation of instance segmentations, usually metrics like mean average precision on the basis of the intersection of predicted and ground truth masks are used. However, since we only have point labels as ground truth, we evaluate segmentation quality only by our visual impression.

3.4.3 Keypoint Detection and Density Map Regression

Another approach to detect objects and their properties is through predicting a single reference point per object, typically the center, and regressing any number of properties via dedicated prediction-heads from the corresponding features. The object centers can be found as the local maxima in a density map. However, for the purpose of cell detection in this thesis, no further object properties need to be regressed, and it is sufficient to regress to a single density map.

The target density map is obtained by convolving a delta function per annotated cell center with a Gaussian distribution. The standard deviation of the Gaussians can be adapted to each object's size. Lacking cell size annotations, we apply the same standard deviation to all. Therefore, our target density map can be computed efficiently with a single convolution operation. As the density map is a superposition of Gaussian distributions, its sum equals the expected number of objects. Therefore, cell counts can be obtained by summing all probabilities without the need of any discretizing post-processing [20]. Individual cells can be identified with local maxima in the heatmap.

Labeling only cell centers via dotting is a natural detection strategy for humans, and minimizes the critical annotation effort, while providing just enough information to localize individual cells [2]. Using Gaussian distributions as training targets, instead of discrete points, is a natural representation of label noise and leads to robustness against positioning errors [2].

Whereas we derived our approach conceptually starting out from CenterNet [24], it became very similar to the approach by Xie et al. [20], from whom we adopted the idea to count cells by summing the densities.

3.5 Data Annotation

Many frameworks for data labeling have been implemented, especially in the computer vision domain. We use Label Studio[5]. Label Studio allows an individual with little technical knowledge to annotate images for a diversity of computer vision tasks, with relative ease. This software was our sole ground truth collection method. While it has been used to provide us data for any of the aforementioned three CV tasks, we only used the single class, relation-less, key-point labeling facilities for the results presented in this work. Our collaborators report an expert annotation effort for point labels at cell centers using Label Studio of roughly 1 hour per image with about 1800 cells.

[5] https://labelstud.io/

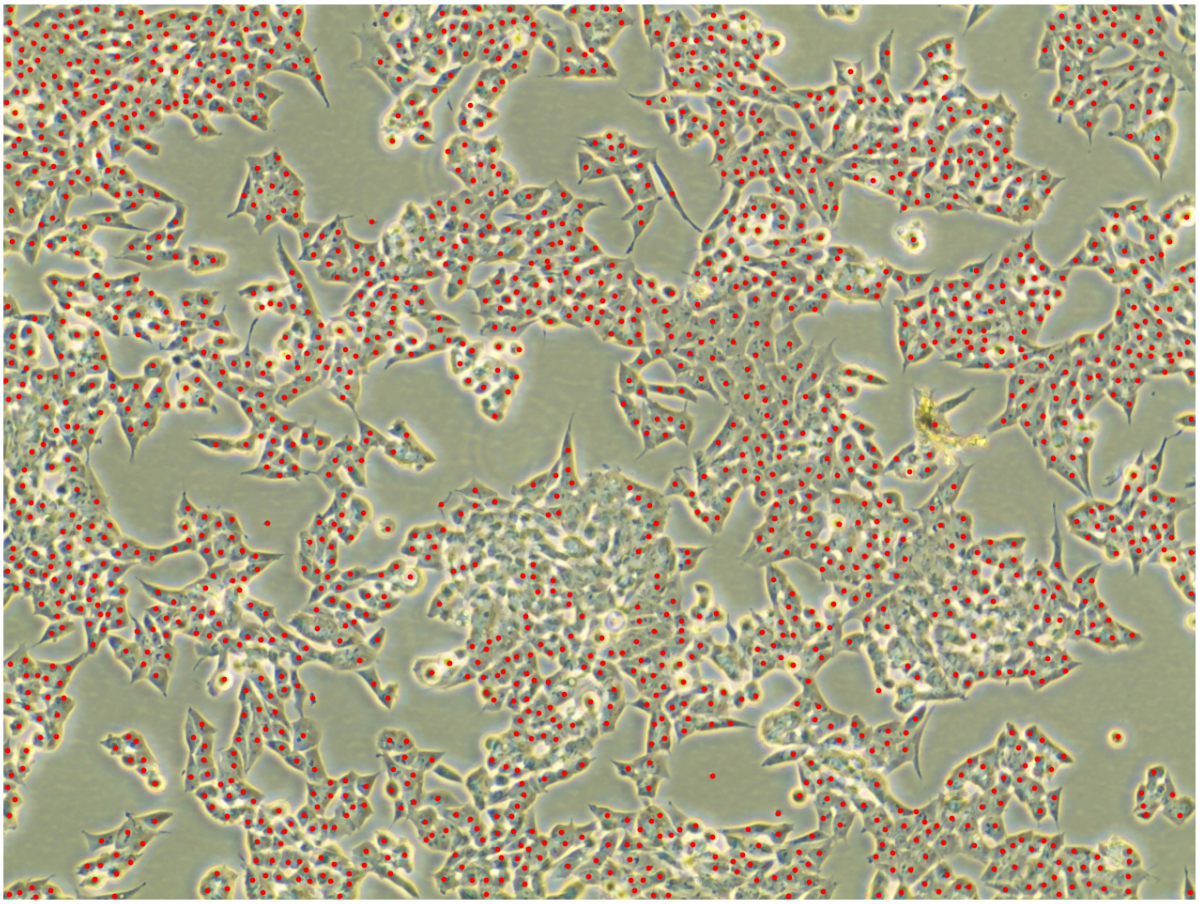

Figure 5: Annotated image from our dataset. Physical phase-contrast 3-channel image obtained at 20x magnification. The center-point annotations are displayed as red dots.

4 Dataset Description

3 images with annotations as displayed in Figure 5 have been collected. The images are physical phase-contrast 3-channel images with 1536×2048 resolution, which have been obtained with the EVOS (ThermoFisher) microscope at 20x magnification. The image properties have been described in Section 3.3. Each image was annotated with 1766, 1198 and 980 points respectively. All experiments making use of annotations used only the first annotated image. Using the complete available data remains as a future improvement. Only for CellNet (Section 5.4) and U-Net (Section 5.3) we use the two other annotated images for evaluation.

Our collaborators also provided a low number of $0.6 - 1\%$ point annotations for objects other than cells, such as debris and dead cells. These annotations were supposed to serve as negative examples for the models, but due to miscommunication with our partners were used along with other cell annotations. More data with different imaging and annotation settings has been collected or exploratory purposes, on which we do not elaborate here.

5 Methods

We apply several deep learning approaches to detect, segment and count objects in images. The training of object detection and instance segmentation models relies on ground truth in the form of bounding boxes and instance masks. Having only point annotations, we generate synthetic ground truth for object detection and semantic segmentation of cells. Zero-shot models have been proposed for generalization of predictions to new data domains without ground truth. We test the performance of four zero-shot instance segmentation models, three of which have been pretrained specifically for cellular segmentation. Using the annotated points, we study different prompting schemes for SAM. Ultimately, we develop an approach to regress cell counts using object density maps.

5.1 Object Detection with Mask R-CNN

As an object detection model we chose Mask R-CNN [7] in the implementation by *torchvision*[6], because it was part of the *TorchVision Object Detection Finetuning Tutorial*[7] from which we started. As it became apparent that object detection approaches are not well suited for dealing with point annotations, we refrained from testing other object detection models, such as from the YOLO-family. To use Mask R-CNN, we adopted the point annotations so that they adhere to the model requirements. We created synthetic box annotations as squares of uniform 25 pixel radius around each annotation point, and synthetic instance masks covering the same area. We derive the model architecture and training settings from the *Torchvision Object Detection Finetuning Tutorial*[7]. Thus, ResNet-50 [28] was used as a backbone pretrained on COCO [33] with Stochastic Gradient Descent (SGD) as an optimizer. Because the great number of objects exceeded the working memory, we split the image in 12 static tiles, as seen in Figure 9. We use no data augmentation and only color-channel-wide normalization as preprocessing. This Mask R-CNN implementation would occasionally yield invalid empty masks, which would break other code. As a crude solution to this problem we trained the model only on the masks, which were fully contained within each image tile.

5.2 Zero-Shot Instance Segmentation

Due to the lack of instance segmentation ground truth, we evaluate the zero-shot performance of Cellpose [9,10], Stardist [8], and MicroSAM [19], which have been pretrained specifically for cellular segmentation. We study SAM [11] with several prompting schemes, using synthetic multiple-point- and box-prompts, derived from the point annotations. Lacking ground truth, we do not evaluate the models quantitatively according to common segmentation metrics, but remain qualitative.

5.2.1 Cellpose

We use the official Python implementation of Cellpose [9], a generalist cellular segmentation algorithm and framework[8]. First we simply run inference of the pretrained Cellpose model, which ships with the package, according to the provided API. The Cellpose framework includes a size model to predict the average cell diameter. We use this size model's diameter prediction as a value for the respective required parameter of the segmentation model. We additionally test slightly and greatly smaller and bigger values derived from this. Lastly, we aim to use the fine-tuning facilities of Cellpose 2 [10] to adapt the model to our dataset and improve the predictions. As we do not have real ground truth segmentation masks, we used segmentations generated by running a watershed with the annotated center points as sources together with a foreground-background mask, obtained as described in Section 9.1.

[6] We used stable release version 0.15 available at https://pytorch.org/vision/0.15/.

[7] https://pytorch.org/tutorials/intermediate/torchvision_tutorial.html

[8] We used version 2.0.5 of the official Python implementation available at https://anaconda.org/conda-forge/cellpose.

5.2.2 SAM and MicroSAM

We use the work on MicroSAM by Archit et al. [19] for their API and for access to the original pretrained SAM models [11] and the models specifically fine-tuned on microscopy image datasets.[9] The SAM architecture involves multiple components, one of which is an image encoder, the backbone, a Vision Transformer [34]. The backbone encodes an image to a compact representation with high information content, which allows further computations to be carried out much more efficiently. Therefore, the majority of the computational load in SAM's architecture is located in the backbone, and choosing the appropriate size to balance computational load with prediction quality can be crucial. We evaluate all available backbones, differing in size and in the case of MicroSAM's backbones pretraining image modality, to determine the best fit for further experiments. We survey different ways to make use of SAM's flexible promptability to find the best way to generate high quality segmentation masks using only the available center point annotations.

Automatic Instance Segmentation. The automatic instance segmentation capability of SAM as implemented by Kirillov et al. is simply wrapping the point prompt mechanism using individual single point prompts on a regular grid. Only promising instance mask candidates are selected [11].

Single Point Prompts. Point prompts allow any number of positive and negative points to hint an instance. Given the available center point annotations, we prompt the model with each of them independently as a single positive point to generate a segmentation mask for each annotated cell.

Multiple Point Prompts. We observed general confusion of the SAM models regarding the scale of the objects which the point prompts pointed at. Therefore, we aimed to improve the results through hinting at the cells' sizes by including additional 4 negative points arranged in an X-shape around the positive annotation point in a constant radius according to an average cell size estimate.

Box Prompts. We make use of SAM's box prompt mechanism to generate segmentation masks, by converting each annotated center-point to a square of constant size, according to an average cell size estimate. This conversion is identical to the one employed for the artificial Mask R-CNN ground truth boxes as described in Section 5.1. We prompt the model with each box independently to generate a segmentation mask for each annotated cell.

5.2.3 Stardist

We evaluate Stardist [8], a segmentation framework based on regressing star convex shaped polygons for each cell.[10] We simply apply a pretrained 2D model available with the official implementation using the official inference API. For the model we chose *2D_versatile_he*, as it has been pretrained on brightfield images, which are more similar to our phase-contrast images than the alternative fluorescence pretraining images.

5.3 Semantic Boundary Segmentation with U-Net

Instance segmentation can also be approached by predicting not each instance shape separately, but by predicting object outlines combined into one target map [25]. We compute the groud truth segmentation boundaries using a watershed algorithm with the point annotations as the sources, as shown in Figure 6. The given bounds are thickened with a maximum filter, resulting in 7 pixel wide boundaries, to increase the training signal and improve robustness against holes in the predicted boundaries. We do not use the image intensity based watershed as described in Section 9.1.3 as the ground truth, because is yielded worse model performance. Instead, we flood only the foreground-background mask (Section 9.1.1) itself.

[9] We used version 0.3.0 of the official Python implementation available at https://computational-cell-analytics.github.io/micro-sam/micro_sam.html.

[10] We used version 0.8.5 of the official Python implementation available at https://anaconda.org/conda-forge/stardist.

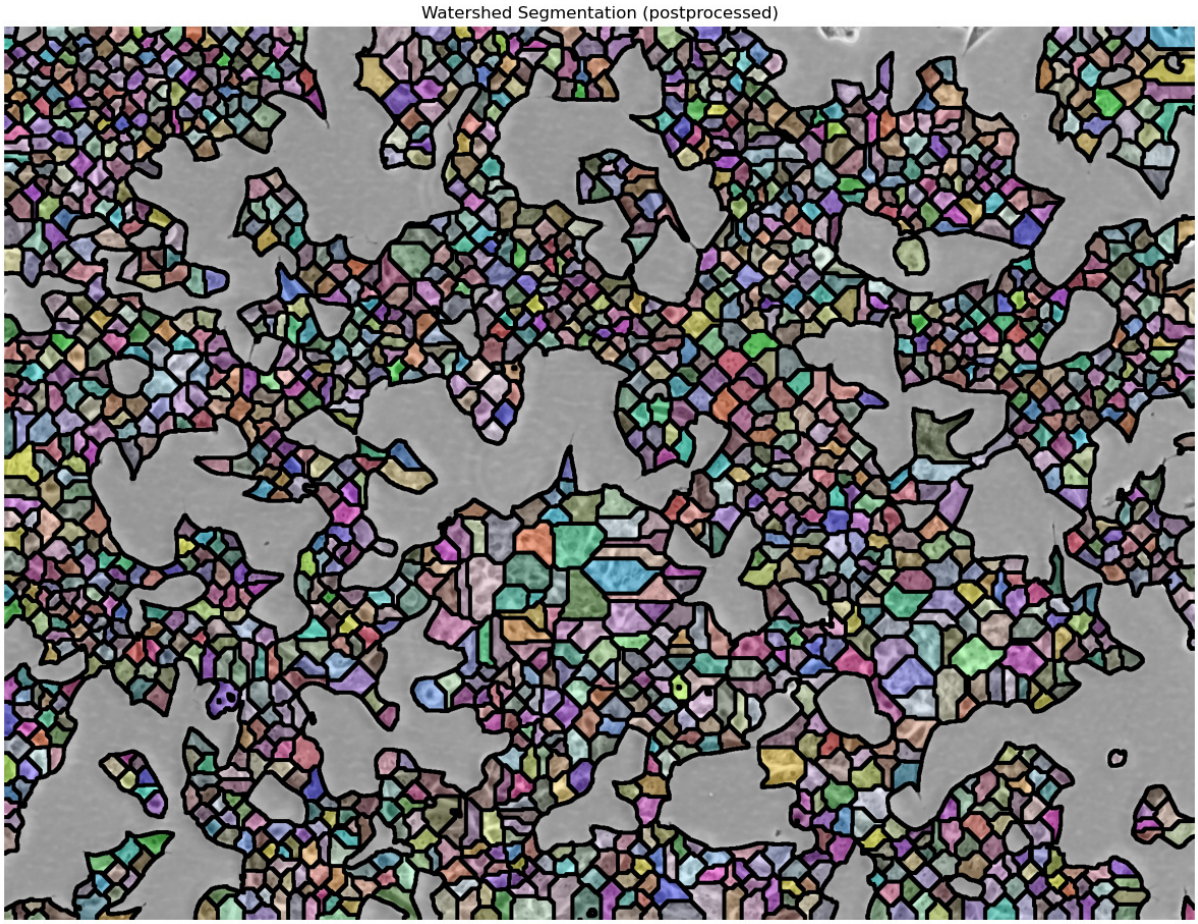

Figure 6: Ground truth boundaries used to train U-Net. The boundaries (black) have been computed with a watershed algorithm using only a foreground-background mask (gray).

We train a U-Net[11] [25] with ResNet-34 [28] backbone with a depth of 5 stages, each halving spatial resolution while doubling feature depth, starting with 16, with sigmoid as the last layer activation function. For the model's encoder we use pretrained weights from ImageNet [35]. It has been shown that this can speed up training convergence even on completely unseen domains, because the lower convolution layers can learn features that are generally useful, like to detect edges of certain orientation [3,36–38]. The training objective is to minimize the Mean Squared Error (MSE) loss between predicted and ground truth boundary maps. We train on one of the annotated images and randomly sample 128x128 pixel tiles from it. We normalize the image to $[0, 1]$ for preprocessing and apply no data augmentation.

The prediction has one channel. A high value indicates that a pixel is probably in a border region, a low value indicates, that a pixel is probably inside a cell or part of the background. We thresholded the boundary prediction and detected objects as the connected components separated by the boundaries. We did not implement a separation for enclosures which correspond to cell instances from enclosures which are filled with background for simplicity and time reasons.

[11]We use the U-Net implementation in version 0.3.3 of the Python package `segmentation-models-pytorch` available at https://smp.readthedocs.io/en/v0.3.3/.

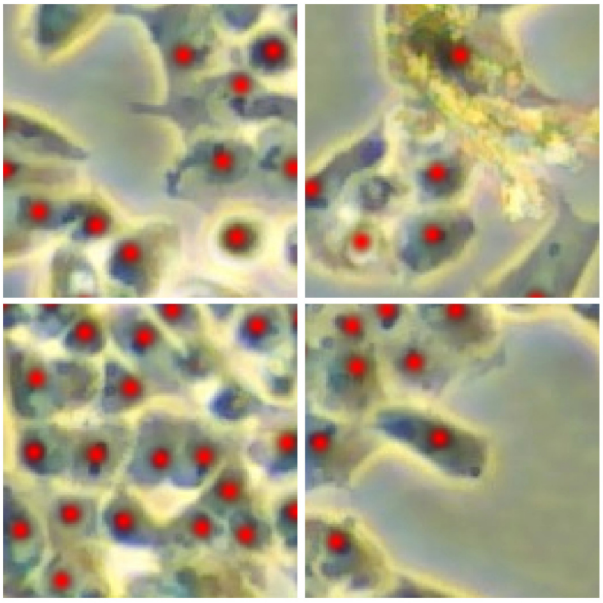

Figure 7: Ground truth density map used to train CellNet. Overlayed in red onto the image data are the Gaussian distributions with standard deviation of 3.5 pixels that correspond to the point annotations.

5.4 Regression based Counting with CellNet

We develop *CellNet* a fully convolutional neural network (CNN) [26,27] to predict an object density map as described in Section 3.4.3, similar to the work by Xie et al. [20]. The local maxima in the density map correspond to the locations of the objects, in our case cells. The number of objects in an image patch is determined by the sum of its density map [2].

The ground truth density map is constructed as the sum of Gaussian probability distributions centered around each point annotation with constant standard deviation [2,20], as shown in Figure 7. We compute the target density map starting from an array initialized with 0s. Every pixel corresponding to a point annotation is set to 1. The array is convolved with a Gaussian kernel with a standard deviation of 3.5 pixels. We chose the standard deviation as high as possible, without visually merging the distributions of the two pairwise closest annotations.

Similarly to Section 5.3 we train a U-Net[11] [25] with ResNet-34 [28] backbone with a depth of 5 stages, each halving spatial resolution while doubling feature depth, starting with 16, with sigmoid as the last layer activation function. However, we initialize the encoder weights randomly, as we observed worse convergence with pretrained weights. The training objective is to minimize the Mean Squared Error (MSE) loss between predicted and ground truth density maps. We train on one of the annotated images and randomly sample 128x128 pixel tiles from it. We normalize the image to $[0, 1]$ for preprocessing and apply no data augmentation. We sum the predicted density maps to predict cell counts.

6 Experiments and Results

To count cells in microscopy images, we employed several deep learning computer vision methods. Object detection and instance segmentation models allow for detection based counting, by trivially counting the number of mask or box predictions. Lacking appropriate ground truth annotations, we generated synthetic labels from point annotations. Training these models was challenging, and zero-shot performance was insufficient with our data. We trained a semantic segmentation model on synthetic cell boundaries, in which the number of enclosures determined the cell count. For regression based counting, we reused the same architecture to predict a density map based on the point annotations, the sum of which yielded a probabilistic cell count.

The computational environment for neural network training was the University of Tartu High Performance Computing cluster [39], equipped with Nvidia Tesla V100 and A100 GPUs with varying amounts of vRAM of which only one was used per experiment at a time. We kept no record of the specific node each experiment was run on, despite the great differences in different hardware components may have affected training. We developed a workflow based on Jupyter, Rsync and Git, to keep track of experiments and results in a reproducible way.

6.1 Mask R-CNN

Figure 8 shows Mask R-CNN's training progress, with training loss in orange and learning rate in blue. The model converges well, however the loss still remains relatively high. Mask R-CNN was trained with an SGD momentum of 0.9, a weight decay of 0.0005 and step-wise reducing learning rate of 0.005, decaying every 3 epochs with γ of 0.1 and with a linear warm-up of 5 epochs, for 10 epochs.

Figure 9 shows the cell boundaries predicted by the trained Mask R-CNN. Here one can see that all predicted boxes and masks are squares, just as the synthetic ground truth is. In some image regions, no cells are detected, while in other regions much more cells are predicted than there should be. To reduce the memory load, we separated the image in 12 independent images before training. A better approach would be to randomly sample tiles during training to avoid systematic errors at the tile boundaries.

Due to missing real ground truth, the quality of the bounding box and segmentation mask predictions could not be quantified. Due to technical issues, the model always predicted exactly 100 objects per tile, such that the ratio of predicted to ground truth objects was determined by the tile-size. Therefore, it did not make sense to quantify the model's detection rate. We moved on to other approaches, which seemed more promising due to not requiring synthetic ground-truth.

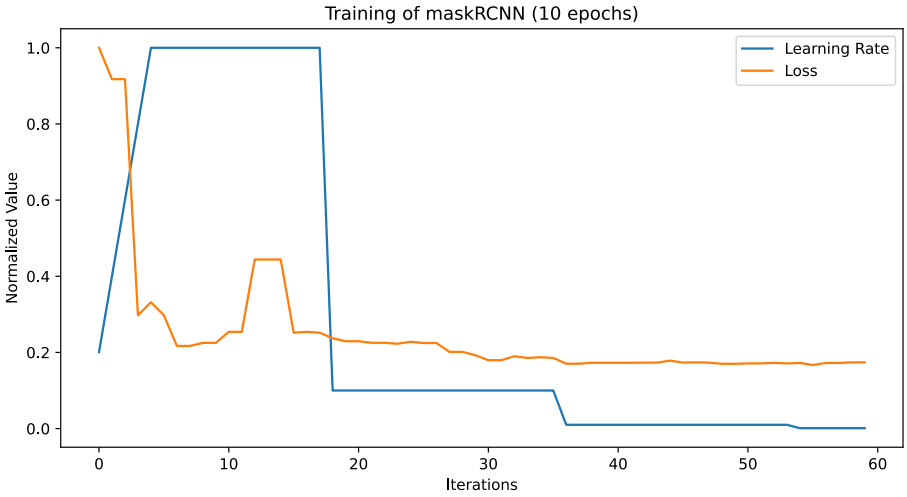

Figure 8: Mask R-CNN training graph. The learning rate (blue) was decreased in steps. The training loss (orange) decreased until stagnation at a relatively high value.

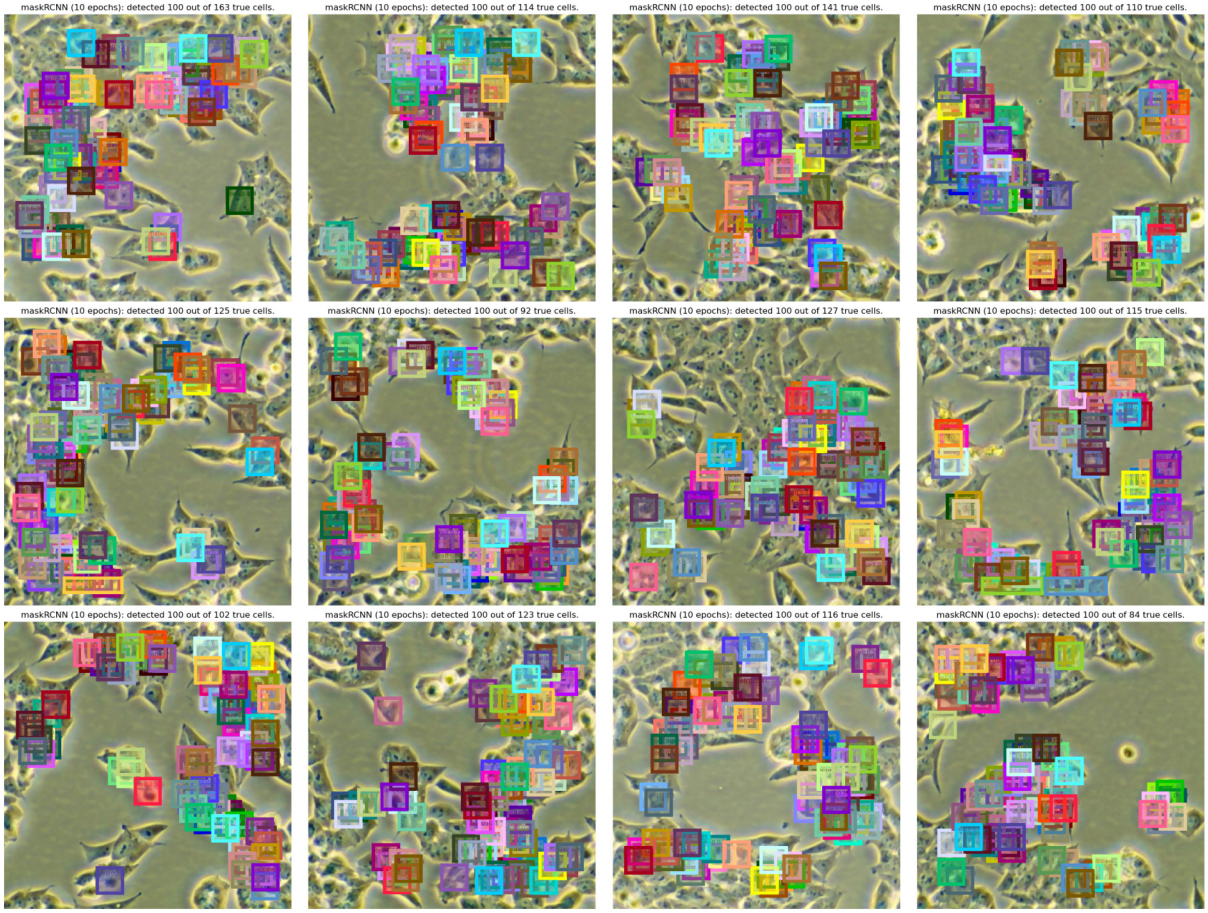

Figure 9: Mask R-CNN predictions on the tiled image. Each cell center is modeled with a constant sized bounding box and an equivalent instance mask. Only 100 objects which lie fully within each tile are predicted. The brightness of the tiles differ, because each tile is normalized independently. Many redundant predictions cluster at certain locations, while predictions are missing completely in other regions. All model predictions have a confidence of $37 - 39\%$*. Due to technical issues, we could only train on the subset of objects which were located fully within each tile.*

6.2 Zero-Shot Instance Segmentation

All applied zero-shot instance segmentation methods, no matter if pretrained especially for cellular segmentation or not, fall short to provide good results. Some methods like Cellpose and SAM with box-prompts generate a subset of good masks. This suggests that with a few actual ground truth masks, these models could be fine-tuned to provide options on our data.

6.2.1 Cellpose

The average cell diameter was predicted by the pretrained Cellpose size model as between 38.5 and 39.3 pixels for all 3 annotated images. This matches with a human sight estimate. The pretrained Cellpose segmentation model's predictions are displayed in Figure 10. It fails to detect a great proportion of cells in the image. A few detections are false positives of background holes between cells. Some clusters of neighboring cells get predicted as a single instance. For the instances that have been identified correctly, however, the quality of the masks is generally very high. The masks mostly follow the visible outlines well and have smooth edges. The predictions are very sparse, but of high quality with very little artifacts. The model has detected a total number of 2971 cells on three images using the automated diameter estimate, which have been annotated with a total number of 5671 cells. Cellpose

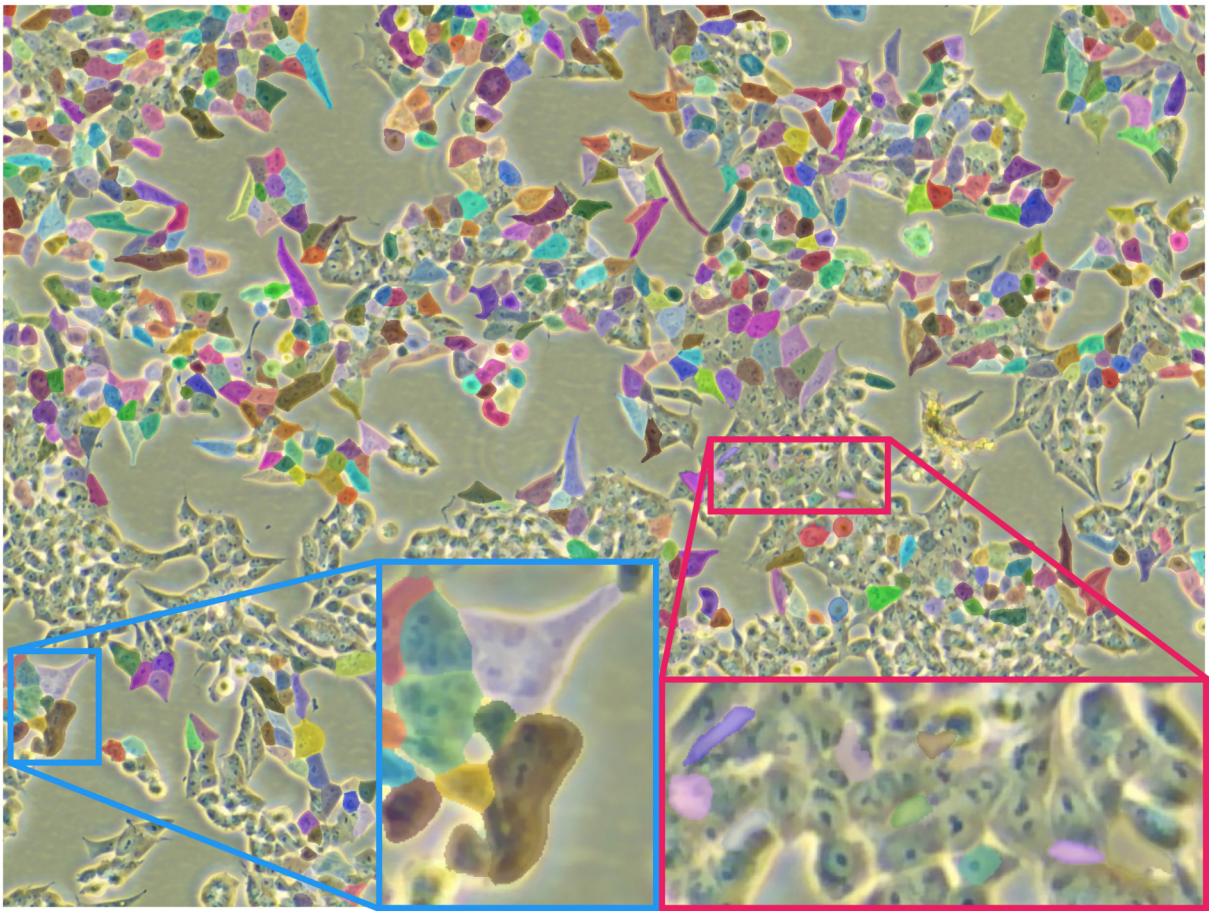

Figure 10: Predictions by pretrained Cellpose. Many cells are accurately segmented by the pretrained Cellpose model. But most cells are missed completely. Some neighboring cells get merged and are segmented as one (blue). Whereas some cell-shaped holes within the cell material get falsely segmented as cells (red).

thus exhibits a very low recall of $< 52\%$ on the dataset, even not accounting for the significant ground truth sparsity and false positive detections.

We prompted the segmentation model with different diameters besides the one automatically predicted by the size model, with ±2px and $\cdot\, 3^{\pm 1}$ relative to that diameter. However, we report no improvement in the results. The algorithm is not very robust to small changes in the diameters — a change from 39px to 40px or 36px drastically reduces the amount of segmented cells. Very small diameters lead to increased false positive detections of impurities in the sample and of background enclosures (even large ones). Larger diameters tend to merge increasingly more cells, with segmentation boundaries adhering less and less to the boundaries visible in the image.

We failed to fine-tune Cellpose on our custom generated fake ground truth segmentations for technical reasons. Despite the simplicity of the training API, the training just froze at the beginning. Because of these difficulties, we decided to abandon Cellpose. However, fine-tuning Cellpose remains as a viable possibility for future work, which we discuss in Section 7.

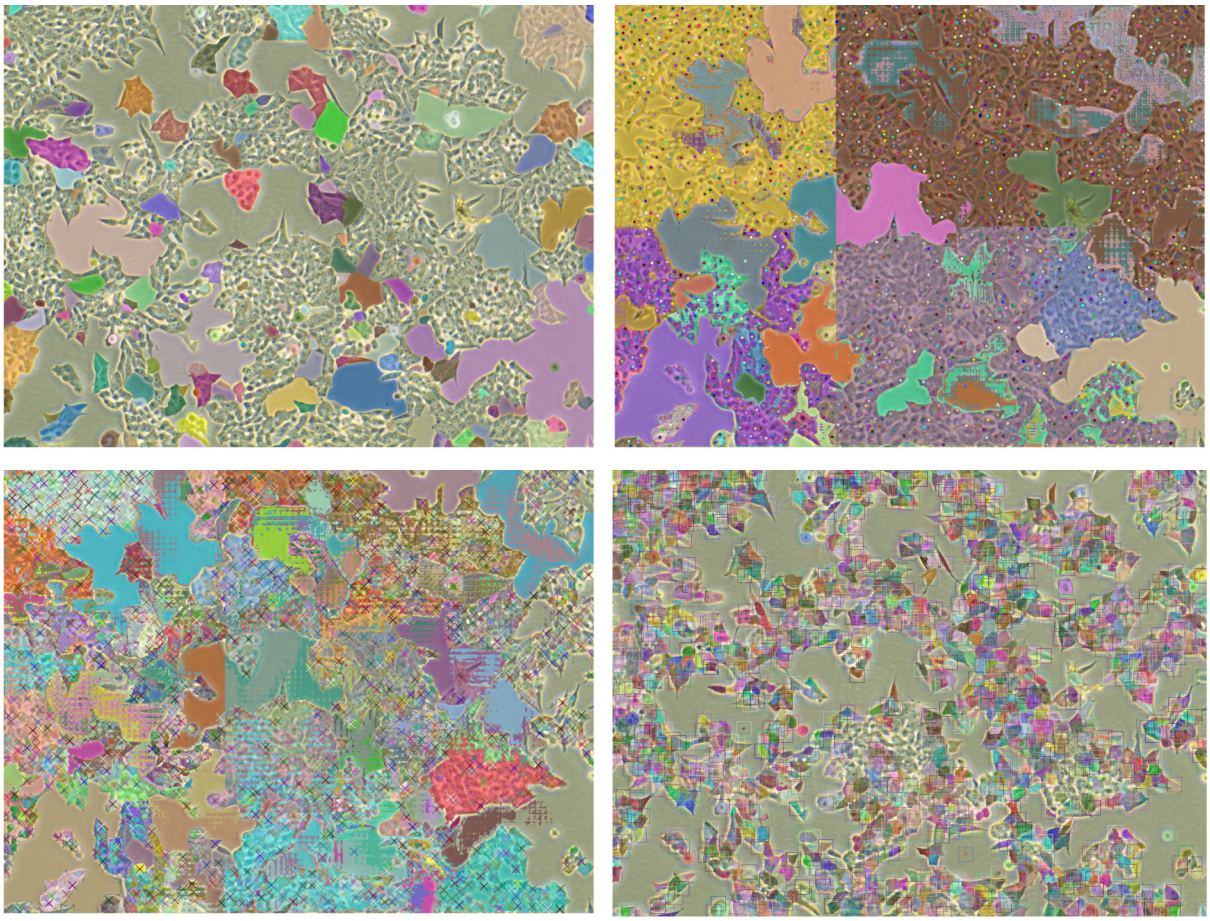

Figure 11: Results from different methods to prompt SAM (using the ViT-H backbone [34]). Automatic instance segmentation based on a regular point grid (upper left); independent point prompts at each annotated center point (upper right); independent multiple point prompts including 4 negative points in X-configuration with 12 pixel radius (lower left); and independent box prompts with 12 pixel radius (lower right).

6.2.2 SAM and MicroSAM

Automatic Instance Segmentation. MicroSAM's [19] implementation makes available multiple pretrained backbones, which can be easily swapped out. Firstly, we aim to compare the general usefulness of all the available backbones, by simply running the point-grid based automatic instance segmentation on a few images. The tested backbones include three differently sized ViT [34] backbones pretrained in the original Segment Anything work [11], as well as three MicroSAM [19] backbones of different sizes pretrained on light microscopy and electron microscopy images. The results are displayed in Figure 24. We observe that the MicroSAM backbones which have been fine-tuned especially for cellular segmentation in microscopy images have the poorest performance by predicting not even a single instance.[12] We therefore conclude that MicroSAM's [19] fine-tuning unlearns a part of the generalization capabilities, which have been one of Segment Anything's major achievements. Such that the fine-tuned models' performances drops more quickly outside its training domain than that of the very general model.

All original SAM backbones consistently fail to detected individual cells. The models can identify some clusters of cells and some sufficiently small background regions. These zero-shot predictions are useless to localize individual cells in our setting. However, in most cases, the boundaries between cell material

[12] This is likely not due to a usage error on our side, because we use exactly the same code for all models, as MicroSAM's API makes it easy to swap only a single parameter naming the backbone.

and background are not crossed by a mask, and the masks for the number of segmented background patches are very accurate. We conclude that SAM is useful as a general shape segmentor and investigate its use with different prompting systems below. We continue with the SAM ViT-H backbone.

Single Point Prompts. As displayed in Figure 11, upper right, prompting the model with each annotation point individually leads to huge and artifact-ridden instance segmentations. We assume this is due to the model missing information about the scale of the targeted objects. It therefore detects the most prominent shapes, which are defined by the boundaries between cell clusters and background. Furthermore, we observe heavy stripe and grid shaped artifacts, especially with the big shapes. The clearly visible vertical line suggests an issue with the tiling logic of MicroSAM's implementation. As the masks are very big, they overlap and occlude each other to a major extent. Our method of visualization which can only color-code one mask at each pixel in the image is therefore unsuited to capture the majority of the predicted shapes. This results in further visual artifacts, which give a worse than actual impression of the model's performance. The boundaries of cell material and background being seemingly populated by a lot of very thin masks is such an effect, because multiple masks with slightly different extension disperse there. Some smaller shapes may be visually occluded by a big one.

Multiple Point Prompts. As we think missing scale information is the major factor impacting the model's performance, we aim to use an additional 4 equally spaced negative points for each positive point as a size hint. The negative points were placed in an x-shape in a 12-pixel radius. We observe generally smaller predicted object shapes, but still nothing close to segmentation of individual cells. Despite smaller object sizes, the artifacts remain and even increase in severity.

Box Prompts. The last available prompting mechanism for our setting is box prompts. We assume the same radius of 12 pixels, such that the corners of the boxes coincide with the positions of the negative points in the previous experiment. The predictions are much better and resemble the actual cell shapes in many cases. This is despite having the same intended information content as the multiple point prompts. However, mostly more isolated cells are segmented more accurately, while cells in dense clusters often get assigned strange shapes that partly overlap with multiple visible cells. This is partly because the phase-contrast artifacts and the low resolution make it very difficult to draw precise boundaries between close cells. To a great extent, the mistakes are induced by the center points being not always in the geometrical center of a cell, and many cells falling out of the 12×12 pixel square. We conclude that this method, while able to produce several adequate shapes, cannot be used for the purpose of individual cell segmentation without ground truth box annotations and instance masks.

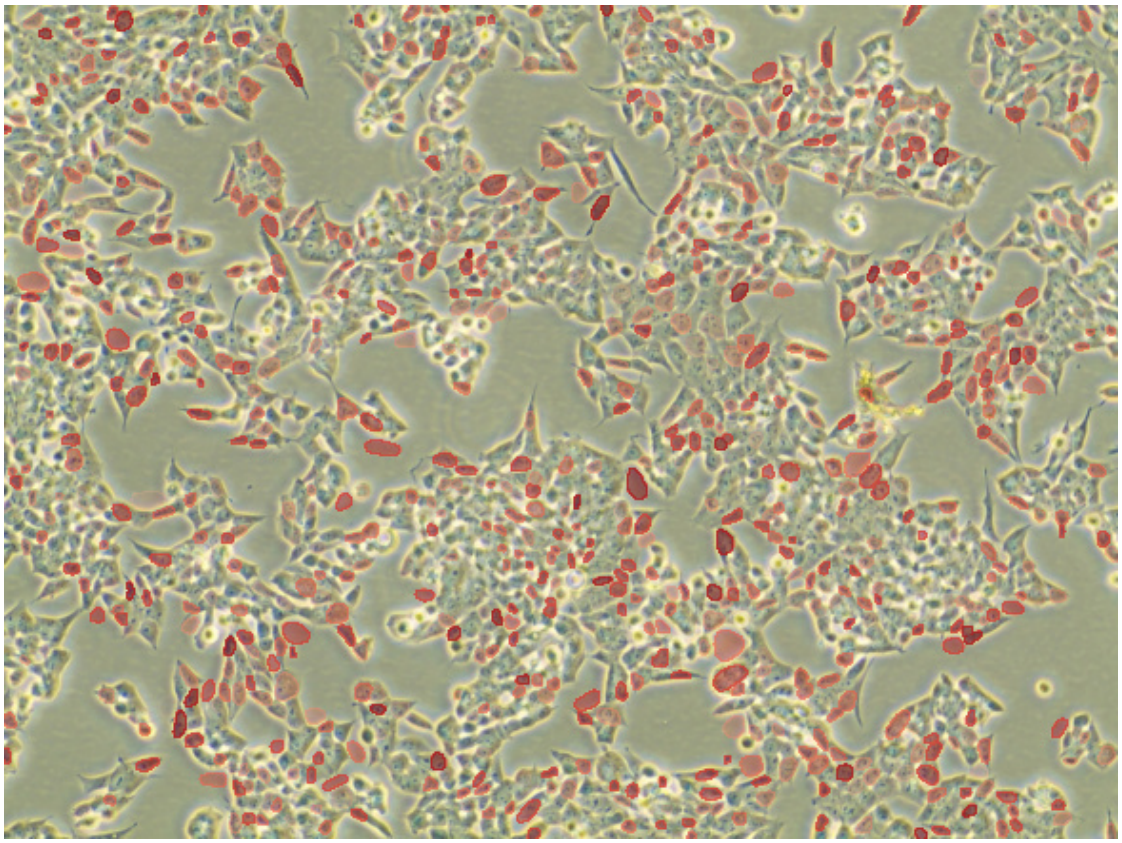

Figure 12: 0-shot instance segmentation predictions of Stardist [8].

6.2.3 Stardist

The predictions of the pretrained *2D_versatile_he* model on one of our images are displayed in Figure 12. Only a low number of cells are correctly detected. Most cells are missed completely. Many convex enclosures of background are wrongly predicted as cells. Overall, we find that Stardist predicts very pleasing cell shapes for the few correctly detected cells. This is probably to a great extent due to the inductive bias of regressing only star convex shaped polygons, which prevents most of the artifacts and broken shapes present in pixel based segmentation approaches. We assume the overall bad performance is to a great extent to the very different characteristics of our phase-contrast images to the pretraining bright-field images. Due to the poor quality of the predictions and the lack of segmentation ground truth, we do not evaluate Stardist's performance quantitatively, but move on to other approaches. However, with appropriate ground truth, retraining the model remains as a viable option.

6.3 U-Net

We trained the U-Net with a learning rate of $2 \cdot 10^{-3}$, stepwise reduced every 60 epochs by the factor 2, for 500 epochs. The loss didn't decrease significantly after 50 epochs. We train on one image and evaluate counts on the two other annotated images. Figure 13 displays the boundary predictions by U-Net. We count cells by thresholding the boundary prediction at 0.5 and counting the 8-connected components. We count only components which are bigger than 100 pixels, to suppress noisy small object predictions. The annotated and predicted cell counts for the training image and both test images are listed in Table 1.

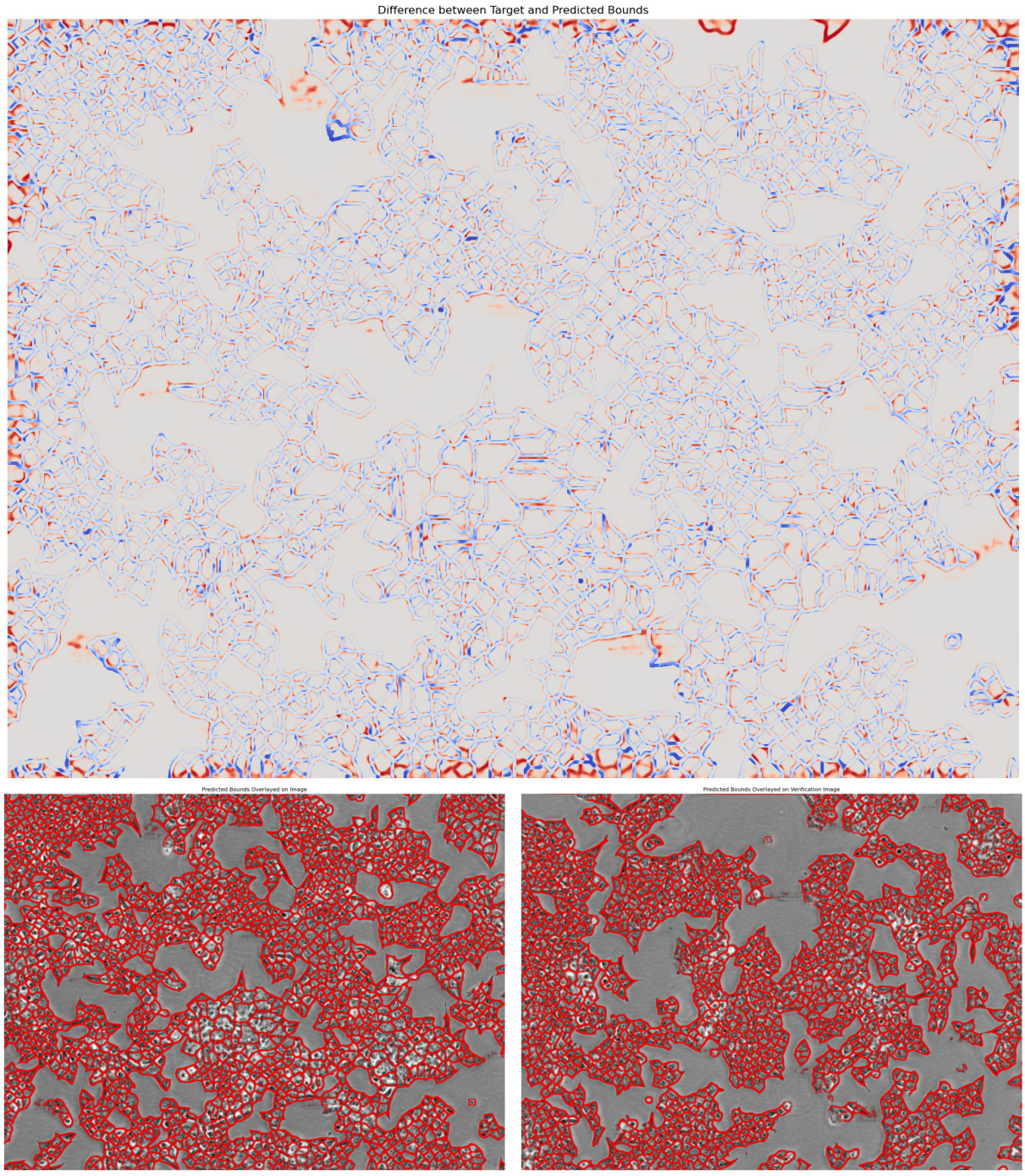

Figure 13: Cell boundary predictions by U-Net. The upper panel shows the difference between ground truth and prediction for the training image. In blue regions the model predicts fewer boundaries than there are in the ground truth, in red regions it predicts more than there is in the training data. The lower left shows the predicted boundaries on the training image. The lower right shows the prediction on one test image.

	train image	test image 1	test image 2
annotated points	1766	1198	890
detected objects	1762	1527	1533

Table 1: Cell counts as produced by U-Net.

The model predicts 27% — 72% more objects than there are annotated points. A part of that deviation is due to wrongly counting connected components of background as cells. But the major conclusion

to draw from these results is, that the model insufficiently generalizes to unseen images. One path to improve this is more data or data augmentation.

The visualization of the predictions in Figure 13 show that the model overfits to the wrong ground truth in the under-annotated regions. However, along the border of the training image, the model seems to generalize better and predict cell boundaries, which are missing in the ground truth due to incomplete annotations. This is especially visible in the difference map (Figure 13, top) in the bottom center and top right. This indicates that the model learns a meaningful relationship between images and boundaries and can generalize to some degree. The effect appears only at the image borders, because their relevance during training is lower due to the way we sample tiles. This also indicates that a future direction of improving robustness against sparse annotations would be to not sample training tiles from systematically annotation free regions.

In contrast to the original work on instance segmentation by boundary detection by Ronneberger et al. [25] it is relevant for our application to discern cell interior from background. We leave the separation of background and cell enclosures by predicting an additional semantic pixel class for background for future work.

6.4 CellNet

We trained the U-Net again with a learning rate of $2 \cdot 10^{-3}$, stepwise reduced every 60 epochs by the factor 2, for 500 epochs. Here, as well, the loss did not decrease significantly after 50 epochs. We trained on one image and evaluated counts on the two other annotated images. Figure 14 displays a sample of the ground truth density used to train the model, as well as the model's predictions on the training image and an unseen image. In the difference map (upper left) we can see that the model learned to localize the majority of the annotated cells well (dark circles with no or little blue). Most false positive densities (red) occur in close vicinity of the annotated cells. This suggests that the cell center annotations might be inconsistent, but the model tried to generalize.

	train image	test image 1	test image 2
annotated points	1766	1198	890
regressed counts	2328	2179	2083
ratio regressed to annotated	132%	182%	234%

Table 2: Cell counts as predicted by CellNet.

The peaks in the predicted density were broader and less intense than in the ground truth. The regressed counts, as listed in Table 2, differed up to 50% between different training runs. Both suggest, that due to annotation incompleteness the relationship between data and annotation is ambiguous.

We also trained the model with the same settings, but added additional annotation points in the under-annotated regions. With these, we observe much better training convergence and more contrast in the predicted density map. Because the additional points were not expert-annotations, we don't include them in the results. However, we note, that countering annotation sparsity seems to be a major path for improving the CellNet method.

In order to avoid the correction of the predicted density map by the normalization factor of the ground truth density map, we tried to not normalize the ground truth. However, the model failed to converge with both sigmoid and identity as a last layer activation function. We leave the localization of cells within the density maps for future work. Possible approaches include finding local maxima and blob detection.

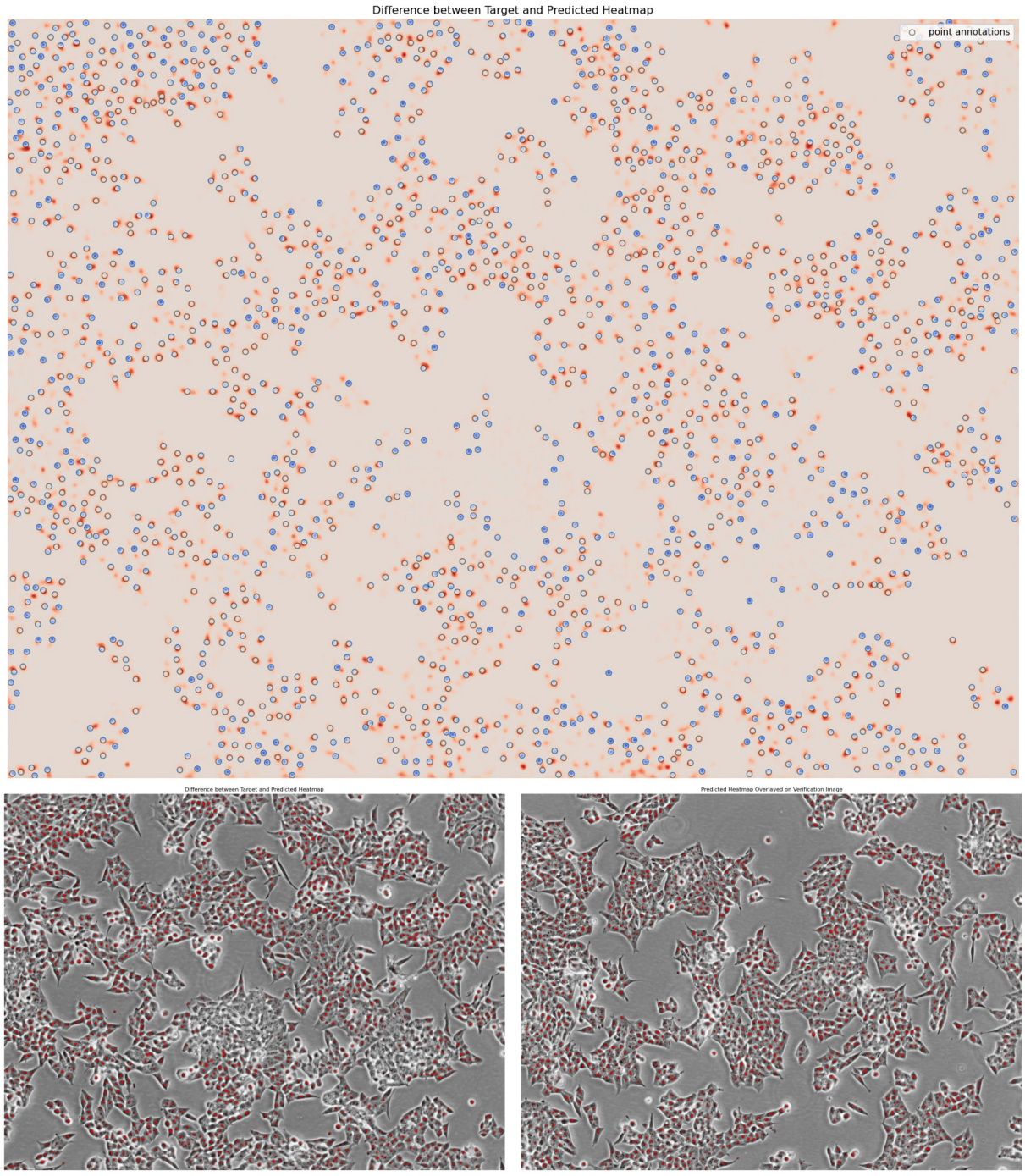

Figure 14: Cell detection by density map regression. The top panel shows the difference between target and prediction for the training image. In the blue regions, the model predicted less density than there is in the ground truth, and in red more. The markers indicate the annotated points. The bottom panels show the predicted density maps on the training image (left) and one test image (right).

7 Discussion

In this thesis, we aimed at detecting and counting cells in microscopy images using point annotations. We applied several deep learning methods and classical approaches to detect, segment, count, and highlight cells. We show promising results for many methods. Cellpose [9] predicted mostly very accurate instance segmentations, but had a very low detection rate. U-Net learned to replicate synthetic cell boundaries, but predicts too many cells. With CellNet we implemented an approach on density map regression. CellNet succeeded in localizing cells, however, the regression counts are too high.

Instance Segmentation. Lacking adequate ground truth to train or fine-tune segmentation models with, we tested the zero-shot performance of Cellpose, MicroSAM, SAM, and Stardist. However, the masks were of very low quality in the case of MicroSAM, SAM and Stardist, and too few in the case of Cellpose. Previous work has shown that fine-tuning capable zero-shot models can lead to drastically improved predictions even with a few training examples [40–42]. Training U-Net to predict synthetic segmentation boundaries resulted in overfitting on flawed ground truth and many interrupted boundaries on unseen data. Fine-tuning segmentation models such as Cellpose and SAM with better real or synthetic ground truth masks remains a viable future approach.

Density Map Regression. Regression of density maps with CellNet, as inspired by [20,2], could reasonably locate cells via local maxima in the density maps. However, the maxima are of inconsistent intensity and shape, and the regressed cells counts were too high. As a future direction it remains to prove, whether the data quality is the reason for this, or whether the method itself needs improvement. As a future experiment, one could search for an optimal standard deviation of the Gaussian density maps, for instance to accurately reflect label positioning noise. Because the majority of the density map is background, different loss functions such as Focal Loss [6] should be experimented with. To specifically improve the regressed counts, we propose to add a count regression loss to minimize the difference between the sums of the target and the predicted density maps on regions of varying scales. [22] propose deep supervision to regress densities at different feature resolutions.

Better Training Data or Better Approaches Dealing with Bad Training Data. All deep learning approaches suffered from the incompleteness of the point annotations. The models can not help but follow the data to overfit into the systematically under-annotated regions, leading to a weaker relationship between images and cells. Furthermore, the data volume is very small, such that predictions on new images are even less confident and consistent. Using more of the available images for training instead of evaluation will be a simple way to improve this to a limited extent. Major future directions of improvements for all the DL approaches surveyed in this work are the collection of more, and more complete data, as wells as methods to deal with sparse annotations.

Data Augmentation. Both the CellNet and U-Net models are relatively small with a ResNet-34 encoder, however they already overfit to the low amount of training data we give them. Data augmentation remains as a low-hanging fruit for future experiments to increase model robustness and reduce overfitting [43–46]. It remains to be seen, how much this increases the effective data volume and reduces the need for additional annotations.

Sparse Labels. Our collaborators reported, that it is impossible, even for a trained expert, to annotate cells in highly congested regions. Therefore, approaches to deal with label sparsity are inevitable for future experiments. A simple improvement would be, not to sample training tiles from under-annotated regions. To this end, an additional negative mask could be collected, to designate non-annotable regions. More sophisticated approaches have been developed to deal with data where negative labels are unavailable or not reliable, called *Positive Unlabeled* (PU) [47–51]. However, these approaches often rely on more complicated mechanisms such as handling predicted pseudo-labels and additional information such as class prior distributions. Other approaches target the training loss to enforce consistent predictions or feature representations between labeled and unlabeled instances [52,53]. Others have

improved tolerance for missing annotations by down-weighting negative examples in the loss [54]. PU or loss based approaches would also increase the robustness against missing cell annotations in less congested regions, thereby reducing the annotation strain further.

Inclusion of Negative Annotations. Our dataset contained a few $0.6 - 1\%$ negative point annotations for dead cells and debris, which we wrongly used as positive cell annotations. Future work should ignore or make appropriate use out of them. Of all tested methods, only CellNet and U-Net performed well enough, that the low number of wrong examples would make a visible difference. For U-Net's training, some of these negative examples lead to very small objects, which have been removed in the preprocessing. However, the bigger a wrong example would be, the more confidently the model would learn to reproduce it wrongly (Figure 13). In the predictions of CellNet (Figure 14), we can see, that the model does not learn to predict very small debris as cells, indicating that it generalizes a meaningful relationship between images and cell locations.

8 Conclusion

In this thesis we studied several deep learning and classical computer vision approaches to segment, detect, highlight and count cells in phase-contrast microscopy images using only sparse center-point annotations. We found that the zero-shot instance segmentation models SAM, Cellpose, MicroSAM, and Stardist could not provide the desired generalization on our images. Fine-tuning these remains as a viable approach for future work. We trained U-Net on boundary detection with synthetic masks. The model learned to segment cells on unseen images. However, cells are overly divided and artifacts are present, which is due to the low synthetic ground truth quality and the image blurriness. It can be improved with real labels or better synthetic label generation. Our approach on density map regression learned to localize cells. The regressed counts suffer from inconsistent labels and low training signal. We discussed future directions to deal with label sparsity, and techniques for improving the training signal. Both boundary detection and density map regression can be improved with more annotations and data augmentation.

Besides deep learning methods, we explored several other approaches to count cells or to extract information from the images which could be used in neural network training. A cell area based count estimation can serve as a baseline if averaged over great image areas, but is less robust due to variable cell densities. We explored watershed and clustering algorithms for the generation of synthetic segmentation ground-truth. Both reviled cell structures, but were strongly impacted by image blurriness and annotation sparsity. Future work needs to improve these algorithms before they can be successfully employed for neural network training. Manipulating images in Fourier space yielded promising pronunciation of cell structures. To reduce the tediousness of manual feature engineering, we developed an interactive application for free-form Fourier filter design. While very interesting and certainly useful for some applications, we found, that pure machine learning approaches are favorable if ground-truth is available.

Most our approaches to cell counting suffered from low imaging quality and sparse ground truth annotations. After the proposed improvements to the quality and amount of annotations and on the methodic sides are studied, it can be evaluated whether the imaging quality of the microscope is sufficient. If not, a different imaging setting will be necessary to meet the application's cell count accuracy demands. An automatic method to accurately count cells using only sparse, cheap to obtain center-point annotations will be of great use in research on the human genome and beyond.

Acknowledgements

I am grateful for Dmytro Fishman, for being an invested supervisor, caring for my personal wellbeing and academic success during and beyond this thesis. I thank Stuart Fawke, our numinous collaborator from the Welcome Sanger Institute, for this opportunity to make a contribution to meaningful research, and for kindly spending his time and energy to provide us data and explanations; in what felt like a prosperous collaboration transcending occasional misunderstandings. I thank Constantin Pape for being a strong source of ideas and guidance, whenever his time allowed, and I asked early enough. I thank the University of Tartu Institute of Computer Science for providing me access to research-accelerating hardware. I especially thank Anne-Kathrin Schultz for being a very awesome and invested study guide, making possible my country-crossing and hard to follow academic endeavours.

Bibliography

[1] Moen E, Bannon D, Kudo T, Graf W, Covert M, Van Valen D. Deep learning for cellular image analysis. Nature Methods 2019;16:1233–46

[2] Lempitsky V, Zisserman A. Learning to count objects in images. Advances in Neural Information Processing Systems 2010;23

[3] Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In:. Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, p. 580–7

[4] Girshick R. Fast r-cnn. In:. Proceedings of the IEEE international conference on computer vision, 2015, p. 1440–8

[5] Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. In:. Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, p. 779–88

[6] Lin T-Y, Goyal P, Girshick R, He K, Dollár P. Focal loss for dense object detection. In:. Proceedings of the IEEE international conference on computer vision, 2017, p. 2980–8

[7] He K, Gkioxari G, Dollár P, Girshick R. Mask R-CNN. In:. Proceedings of the IEEE international conference on computer vision, 2017, p. 2961–9

[8] Schmidt U, Weigert M, Broaddus C, Myers G. Cell Detection with Star-Convex Polygons. In:. Medical Image Computing and Computer Assisted Intervention - MICCAI 2018 - 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part II, 2018, p. 265–73. https://doi.org/10.1007/978-3-030-00934-2_30

[9] Stringer C, Wang T, Michaelos M, Pachitariu M. Cellpose: A Generalist Algorithm for Cellular Segmentation. Nature Methods 2021;18:100–6

[10] Pachitariu M, Stringer C. Cellpose 2.0: How to Train Your Own Model. Nature Methods 2022;19:1634–41

[11] Kirillov A, Mintun E, Ravi N, Mao H, Rolland C, Gustafson L, et al. Segment Anything. Arxiv Preprint Arxiv:230402643 2023

[12] Hu C, Li X. When sam meets medical images: An investigation of segment anything model (sam) on multi-phase liver tumor segmentation. Arxiv Preprint Arxiv:230408506 2023

[13] He S, Bao R, Li J, Grant P E, Ou Y. Accuracy of segment-anything model (sam) in medical image segmentation tasks. Arxiv Preprint Arxiv:230409324 2023

[14] Roy S, Wald T, Koehler G, Rokuss M, Disch N, Holzschuh J, et al. Zero-shot medical image segmentation capabilities of the Segment Anything Model. arXiv 2023. Arxiv Preprint Arxiv:230405396 n.d.

[15] Mazurowski M A, Dong H, Gu H, Yang J, Konz N, Zhang Y. Segment anything model for medical image analysis: an experimental study. Medical Image Analysis 2023;89:102918–9

[16] Deng R, Cui C, Liu Q, Yao T, Remedios L W, Bao S, et al. Segment anything model (sam) for digital pathology: Assess zero-shot segmentation on whole slide imaging. Arxiv Preprint Arxiv:230404155 2023

[17] Ma J, He Y, Li F, Han L, You C, Wang B. Segment anything in medical images. Nature Communications 2024;15:654–5

[18] Huang Y, Yang X, Liu L, Zhou H, Chang A, Zhou X, et al. Segment anything model for medical images?. Medical Image Analysis 2024;92:103061–2

[19] Archit A, Nair S, Khalid N, Hilt P, Rajashekar V, Freitag M, et al. Segment Anything for Microscopy. Biorxiv 2023:2023–8

[20] Xie W, Noble J A, Zisserman A. Microscopy cell counting and detection with fully convolutional regression networks. Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization 2018;6:283–92

[21] Lu E, Xie W, Zisserman A. Class-agnostic counting. In:. Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14, 2019, p. 669–84

[22] He S, Minn K T, Solnica-Krezel L, Anastasio M A, Li H. Deeply-supervised density regression for automatic cell counting in microscopy images. Medical Image Analysis 2021;68:101892–3

[23] Raza S E A, AbdulJabbar K, Jamal-Hanjani M, Veeriah S, Le Quesne J, Swanton C, et al. Deconvolving convolutional neural network for cell detection. In:. 2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019), 2019, p. 891–4

[24] Zhou X, Wang D, Krähenbühl P. Objects as points. Arxiv Preprint Arxiv:190407850 2019

[25] Ronneberger O, Fischer P, Brox T. 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, 2015, p. 234–41

[26] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proceedings of the IEEE 1998;86:2278–324

[27] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems 2012;25

[28] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In:. Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, p. 770–8

[29] Ensembl. Human assembly and gene annotation 2022

[30] Bostan E, Froustey E, Rappaz B, Shaffer E, Sage D, Unser M. Phase retrieval by using transport-of-intensity equation and differential interference contrast microscopy. In:. 2014 IEEE International Conference on Image Processing (ICIP), 2014, p. 3939–43

[31] Waller L, Kou S S, Sheppard C J, Barbastathis G. Phase from chromatic aberrations. Optics Express 2010;18:22817–25

[32] Zernike F. How I discovered phase contrast. Science 1955;121:345–9

[33] Lin T-Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, et al. Microsoft COCO: Common Objects in Context. In:. Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13, 2014, p. 740–55

[34] Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, et al. An image is worth 16x16 words: Transformers for image recognition at scale. Arxiv Preprint Arxiv:201011929 2020

[35] Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L. Imagenet: A large-scale hierarchical image database. In:. 2009 IEEE conference on computer vision and pattern recognition, 2009, p. 248–55

[36] Matthew Zeiler D, Rob F. Visualizing and understanding convolutional neural networks. In:, 2014

[37] He K, Girshick R, Dollár P. Rethinking imagenet pre-training. In:. Proceedings of the IEEE/CVF international conference on computer vision, 2019, p. 4918–27

[38] Xie Y, Richmond D. Pre-training on grayscale imagenet improves medical image classification. In:. Proceedings of the European conference on computer vision (ECCV) workshops, 2018, p. 0–1

[39] University of Tartu. UT Rocket 2018. https://doi.org/10.23673/PH6N-0144

[40] Hu X, Xu X, Shi Y. How to efficiently adapt large segmentation model (sam) to medical images. Arxiv Preprint Arxiv:230613731 2023

[41] Xie W, Willems N, Patil S, Li Y, Kumar M. SAM Fewshot Finetuning for Anatomical Segmentation in Medical Images. In:. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2024, p. 3253–61

[42] Paranjape J N, Nair N G, Sikder S, Vedula S S, Patel V M. Adaptivesam: Towards efficient tuning of sam for surgical scene segmentation. Arxiv Preprint Arxiv:230803726 2023

[43] Zhang H, Cisse M, Dauphin Y N, Lopez-Paz D. mixup: Beyond empirical risk minimization. Arxiv Preprint Arxiv:171009412 2017

[44] DeVries T, Taylor G W. Improved regularization of convolutional neural networks with cutout. Arxiv Preprint Arxiv:170804552 2017

[45] Cubuk E D, Zoph B, Mane D, Vasudevan V, Le Q V. Autoaugment: Learning augmentation policies from data. Arxiv Preprint Arxiv:180509501 2018

[46] Yun S, Han D, Oh S J, Chun S, Choe J, Yoo Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In:. Proceedings of the IEEE/CVF international conference on computer vision, 2019, p. 6023–32

[47] Wolny A, Yu Q, Pape C, Kreshuk A. Sparse object-level supervision for instance segmentation with pixel embeddings. In:. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, p. 4402–11

[48] Bekker J, Davis J. Learning from positive and unlabeled data: A survey. Machine Learning 2020;109:719–60

[49] Bepler T, Morin A, Rapp M, Brasch J, Shapiro L, Noble A J, et al. Positive-unlabeled convolutional neural networks for particle picking in cryo-electron micrographs. Nature Methods 2019;16:1153–60

[50] Lejeune L, Sznitman R. A positive/unlabeled approach for the segmentation of medical sequences using point-wise supervision. Medical Image Analysis 2021;73:102185–6

[51] Liu B, Dai Y, Li X, Lee W S, Yu P S. Building text classifiers using positive and unlabeled examples. In:. Third IEEE international conference on data mining, 2003, p. 179–86

[52] He K, Fan H, Wu Y, Xie S, Girshick R. Momentum contrast for unsupervised visual representation learning. In:. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, p. 9729–38

[53] Tarvainen A, Valpola H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. Advances in Neural Information Processing Systems 2017;30

[54] Kaliuzhnyi D, Fishman D, Papkov M. Reducing the Effect of Incomplete Annotations in Object Detection for Histopathology 2023

[55] Müller M. Fundamentals of music processing: Using Python and Jupyter notebooks. vol. 2. Springer; 2021

[56] Korzeniowski F, Widmer G. Feature learning for chord recognition: The deep chroma extractor. Arxiv Preprint Arxiv:161205065 2016

[57] Harris F J. On the use of windows for harmonic analysis with the discrete Fourier transform. Proceedings of the IEEE 1978;66:51–83

9 Appendix

9.1 Other Explorations

Lacking ground truth to train or fine-tune instance segmentation models, we studied watershed and clustering algorithms to derive instance masks based on pixel intensity and distance to the annotated points. To improve upon the blurriness of the images, we studied several approaches on how to make cell boundaries more visible. For this, we employed a wide range of classical computer vision filters, and studied the use of transformations in Fourier space. As a baseline estimate, we compute counts from the image area covered by cell material and background and analyze their robustness.

9.1.1 Area and Density based Count Estimation

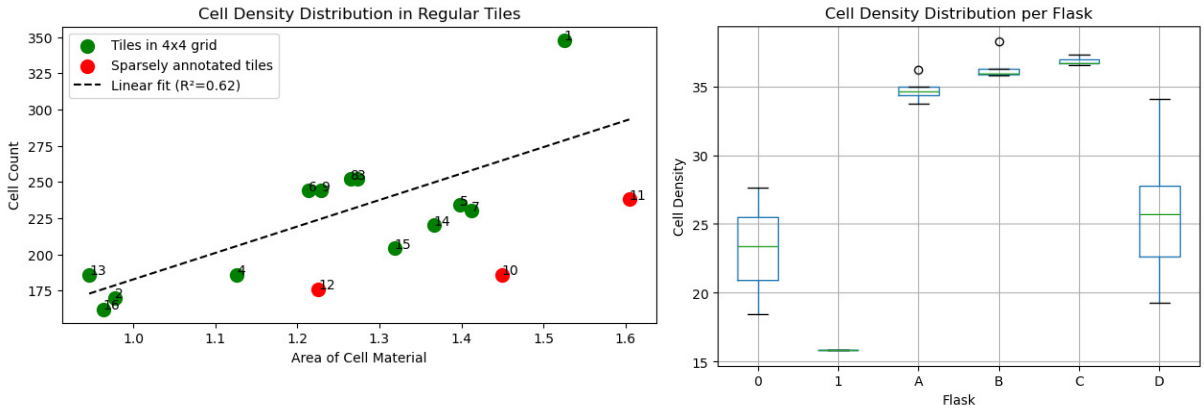

Figure 15: Left: Linear regression of computed cell area (x-axis) to annotated cell count (y-axis). One annotated image has been separated into 16 equally shaped tiles to study the uneven distribution of cell densities within a sample. A linear regression (line) has been performed on the well annotated tiles (green). In red are the tiles with large under-annotated regions. The numeral labels indicate the tile number from top left to bottom right in the image. | Right: Distribution of cell densities in different flasks. Each flask was imaged at several magnifications.

A rough baseline estimate for the number of cells on an image may be estimated by measuring the area covered by cell material, multiplying it with an assumed constant density of cells in the cell material.

The area is the sum of a binary foreground-background mask. The mask has been contributed by our coworker and is computed by thresholding a fake phase reconstruction generated by treating the blue and green channels as two separate bright-field images with different focal planes, according to [30]. This worked, because the different color channels of the phase-contrast images had different focal planes (see Section 3.3). In Figure 17, the resulting foreground is displayed as the colored area.

Cells are of different sizes and form clusters of varying density. If measured over greater area, these fluctuations may average-out. We study the robustness of area and density based count estimation by studying the uneven distribution of cell densities within a sample. To this end, in Figure 15, left, we separated one annotated image into 16 equally shaped tiles, plotted annotated cell count against computed cell area, and fitted a linear regression to the well annotated tiles.

The standard deviation of the densities in each tile relative to the mean density is 12%. This is well above the 10% regular difference which our collaborators wish to improve upon (Section 3.2). These results would improve with a greater field of view, as the density fluctuations would average-out.

As a future work, one could extrapolate the relationship between the size of the field of view and the density variance to determine what image area should be covered at least to fulfill the application's

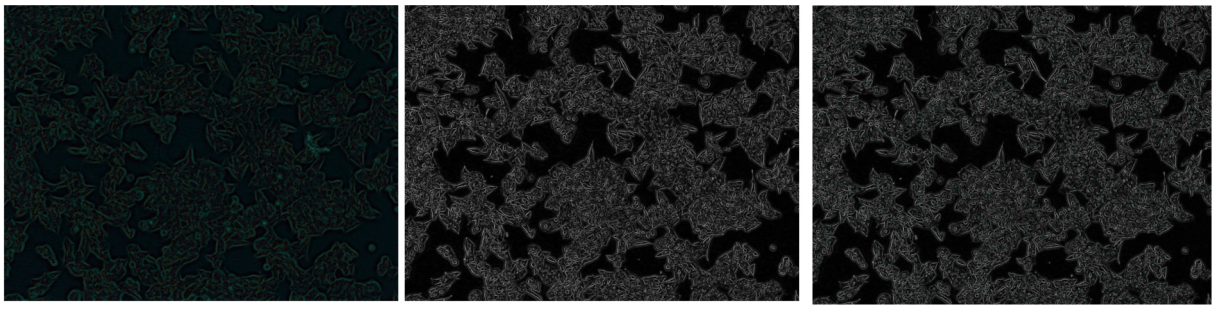

Figure 16: Cell boundary detection with image derivatives. To the left is the RGB Sobel-filtered image. The middle shows only the Sobel-filtered green channel. To the right is the first order derivative of the green image channel. The cyan dots indicate the annotated cell positions.

accuracy demands (Section 3.2). A possible approach could be to lower the value range of a density map by increasing the size of a blurring kernel.

However, our results on images from samples of other experiments in the series (flasks) indicate, that the cell density inside the cell material would fluctuate too much across different samples. Figure 15, left, plots the distribution of cell densities in multiple images taken at different magnifications from six different flasks. The densities have been computed using the same method for determining the area, but the flask-wide count determined by the Countess 3 FL (see Section 3.2). A possible explanation for great big variance within a flask is, that due to the different magnifications, our method of segmenting background via phase reconstruction is less reliable. A part of the great variance between flasks is, that flasks A, B, and C contain unusually densely-packed cells with almost no background area, due to a mistake in the lab. Their density is higher, because tightly-packed cells are overlapping more.

We conclude that this approach can serve as a simple to obtain baseline for cell counts, however it is not very reliable, as the density of cells varied with different screens.

9.1.2 Edge Detection with Image Derivatives

Cell boundaries are visible, because around them often the pixel intensities change drastically. Edge detection filters can thereby be implemented to pronounce the contours of individual cells. Figure 16 shows three different image derivative edge detection filters. All of them can nicely highlight the difference between background and cell material. Especially, the contours of cells on the edge of cell clusters are highlighted. However, boundaries of neighboring cells are indiscernible from other structures from inside the cells.

9.1.3 Synthetic Segmentation Ground Truth using Watershed

Because the performance of the surveyed zero-shot instance segmentation models is too low, we considered fine-tuning them to our data. However, due to the lack of ground truth, we try to derive synthetic ground truth instance masks only using the point annotations, by using a watershed algorithm.

Figure 17 shows the instance segmentation yielded by applying a watershed algorithm[13] to the green channel of an image, with the cell center annotations as sources, and a foreground-background mask. The mask is the cell-background segmentation described in Section 9.1.1. We can see that only a few cells are segmented well. Often multiple cells, despite having a watershed source each, get merged into one big object. Cells that do not have been annotated get merged with a neighboring cell. The quality of the segmentations depends highly on the mask. Some small enclosures of background are wrongly determined by the mask as foreground, and are then merged with a neighboring cell.

[13] We applied `skimage.segmentation.watershed` from version 0.22.0 of scikit-image.

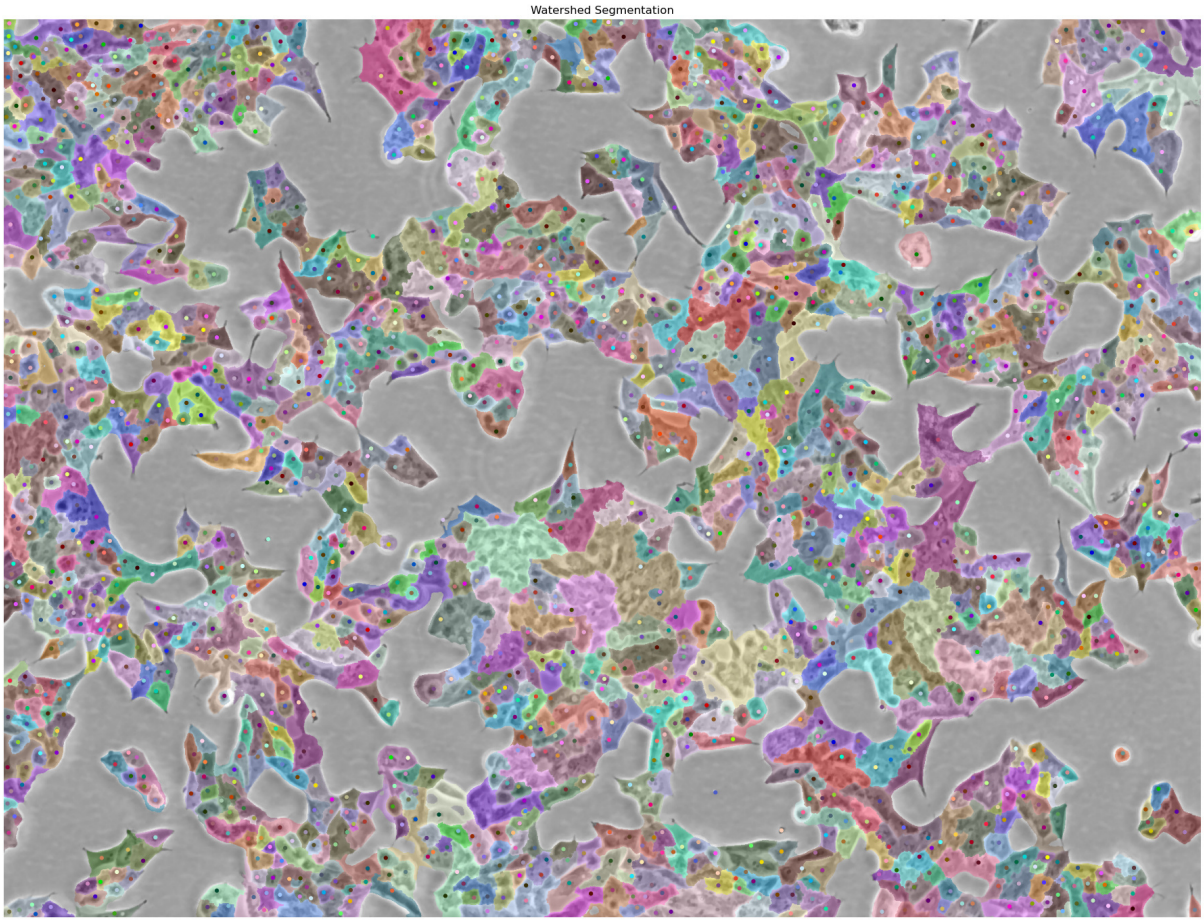

Figure 17: Watershed segmentation of the image. The background mask is in gray. Each object shape is in a color. The dots indicate the sources, corresponding to the point annotations.

A different method of computing watershed contributed by our coworker generated more reasonable cell boundaries. However, due to the lack of time and the method being implemented in a foreign language and framework, we leave an improved watershed for future work. We also computed watershed on both RGB and gray Sobel-filtered images, but the results were much worse.

9.1.4 Synthetic Segmentation Ground Truth using K-Means Clustering

As another approach to generate synthetic instance segmentation ground truth, we tried k-means clustering based on pixel intensity and distance to a point annotation. We surveyed several numbers of clusters up to 20, but found that 4 yielded the best separation of meaningful structures. The results are displayed in Figure 18. There we can see that the red colored cluster is very correlated with dark structures in cell interiors. I could be useful to aid machine learning model training. However, on itself it cannot be used directly for cell counting, because it produces very fragmented structures. In regions with missing annotations, such as in the image center, these relationships break. Neither of the clusters can be consistently associated with cell boundaries. Increasing the number of clusters lead only to even more fragmented structures.

9.1.5 Pronunciation of Spatial Features using Fourier Space

Cell boundaries and cell interiors have certain spatial properties that distinguish them from each other and from the background. For example, cell boundaries are similarly thick, most cells are enclosures of cell boundaries with similar size, and cell interiors often contain other small structures. We employ Fourier transformation to highlight spatial properties of cells.

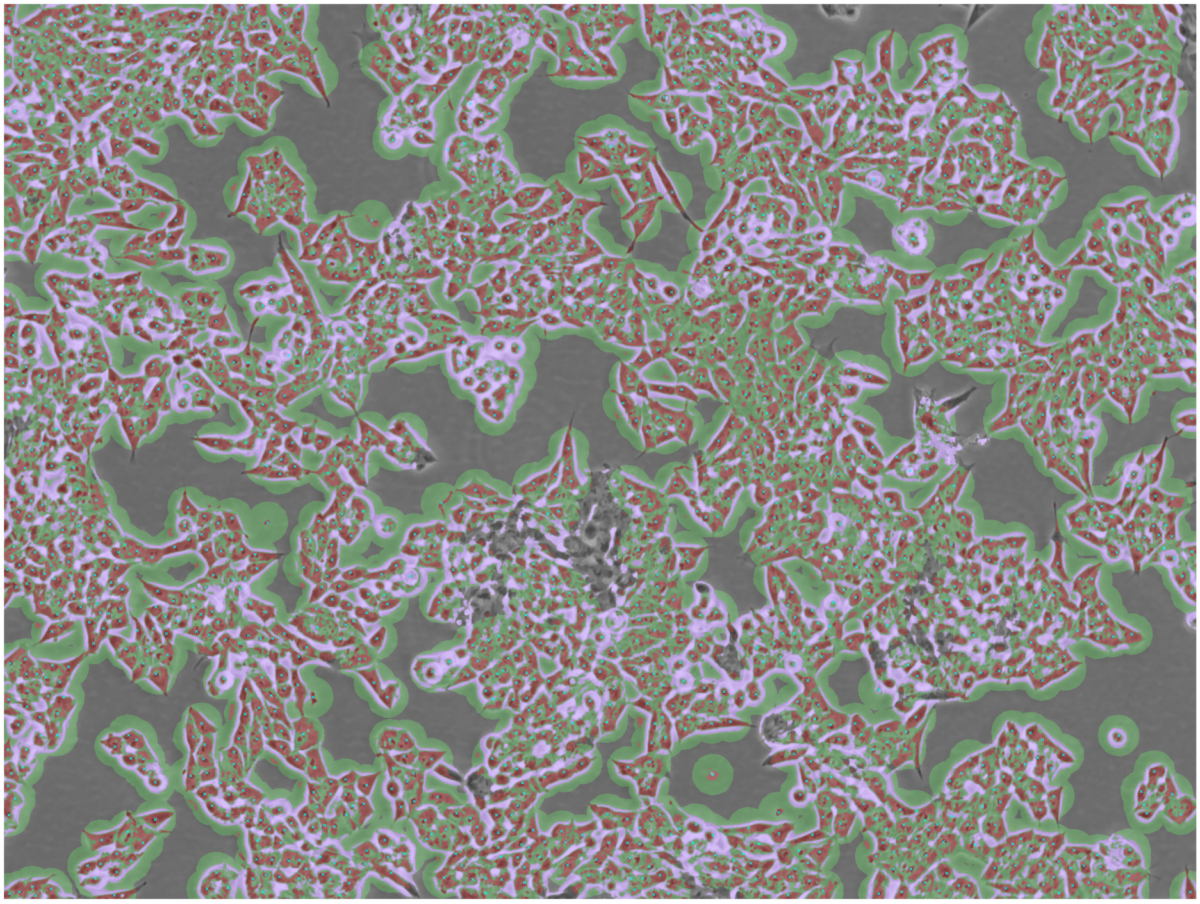

Figure 18: K-Means clustering of the image by pixel intensity and distance to the nearest annotated point. The colors indicate to which one of 4 clusters an image region is optimally belonging. The light blue dots are the annotated points.

In Music Information Retrieval, the Fourier Transformation is used as a powerful method to extract structural music information from an audio signal [55]. The time-to-air-pressure representation of an audio sample is converted to a frequency-to-intensity diagram, its *spectrogram*, in which the sample's composition of individual frequencies becomes visible, and musically relevant information such as harmony can be extracted much easier [56].

We apply this concept to two-dimensional images to extract spatial structural information. When computing the (2D) FT of an image patch, its composition out of *spatial frequencies* or *wavelengths* becomes apparent. For example, if an image contains only big structures, such as a one big texture-less blob, its 2D spectrogram will highlight only at places corresponding to long wavelengths. If an image contains small structures, the corresponding regions in the spectrogram will be highlighted. We base our self-developed *Localized Fourier Transformation* on this idea (Section 9.1.5.1).

The Fourier representation can not only be used to highlight structurally relevant information, but also to manipulate an image. For example, if we take a photography, compute its FT, mask the short wavelength regions in its spectrogram, and transform it back in to an image, it will look blurry and will not contain any small structures. This technique could be applied to reduce noise by filtering out only very short wavelengths. On the other hand, if the long wavelength regions of the spectrogram are masked, the resulting image will only contain small structures, practically highlighting edges in the image. By combining two such filters, one can select a subset of sizes of structures to filter out. For example, if chosen appropriately such a filter could *highlight cell boundaries*, while large structures, such as the cytoplasm vs. background, and small structures, such as subcellular structures or dust, get

removed from the image. We apply this idea using the Butterworth band-pass filter Section 9.1.5.2 and in a more general approach with a self-developed interactive wavelength filter Section 9.1.5.3

The computation of the Fourier Transformation produces visual artifacts, which are due to the discontinuity of the patch border. When transforming the spectrogram back without changing it, these get reintegrated into an exact reconstruction of the original image. However, when the spectrogram is altered, the reconstructed image will display undesired artifacts as well. To avoid these artifacts, we apply the Hann Window [57] to all image patches before computing their FT.

Localized Fourier Transformation.

Cells contain many small structures with high contrast, cell boundaries are mostly lines of similar width, often accompanied by large smooth halo artifacts (Section 3.3), and background contains only few low-contrast structures. Therefore, the composition of structures of different scales in the neighborhood of an image pixel correlated with whether it belongs to a cell inside, a cell boundary, or the background. The spatial Fourier transform can compute the composition of sinusoidal waves of different spatial wavelengths for an image patch, which, when added together, reconstruct the patch. A high concentration of short spatial wavelengths can be associated with a prevalence of small structures.

We introduce the Localized Fourier Transform (LFT) as a way to analyze how the neighborhood of any pixel is composed of structures of different sizes, in order to decide whether the pixel is a cell, a boundary, or background. Figure 19, top row shows, how the LFT is computed. For every pixel in the image we take a 64×64 pixel crop around it, apply the Hann window [57] (left) and compute the Fourier transform (middle). The Fourier spectrum is two-dimensional; the vector from a pixel in the spectrum to the spectrum's center corresponds to the direction and wavelength of the sinusoidal component that the pixel describes. The value of the pixel corresponds to the amplitude of the component, and therefore to the prevalence of structures of this size when measured in this direction. The cell microscopy images in our dataset are rotation isotropic. Therefore, it makes sense to analyze the spectrum only with regard to the intensities of different wavelengths and ignore their angle. We use a linear polar transform (right), such that the wavelength spectrum (red) independent of the angle, can be computed simply by summation along the y-axis.

The result, after close to 6 hours of single-CPU compute time, is a 64-dimensional vector for every pixel, the spatial wavelength spectrum of every pixel's neighborhood. As a method of visualization, we choose to interpret the spatial wavelength spectrum as a light wavelength spectrum in the humanly visible range and convert to an RGB color.[14] The result is displayed in Figure 19, bottom. There we can see that we mostly succeeded in converting a blurry image into an even blurrier image. The regions with cells are brighter than the background, which is expected due to the higher prevalence of structures of any scale. However, this visualization failed to clearly delineate borders between cells and the background. A possible reason the windowing function.

The windowing function reduces FT artifacts, which appear due to the discontinuity of image patch borders. However, smooth windowing also leads to a blurry definition of neighborhood for a pixel, which reflects in the pixel's spectrum. The impact of the artifacts would be more or less the same for every pixel, and because we are only interested in comparison of spectra, not their absolute composition, irrelevant. Therefore, as a future direction, windowing could be avoided, to increase the sharpness of the LFT. Another possible reason for the blurriness of the LFT could be, that the structures of interest correlate only with a part of the spectrum. The conversion of the whole image to RGB, effectively reduces the spectral resolution from 64 values to 3, which is insufficient to capture substructures in the spectrum. A possible improvement could be an interactive way to look only at a specific part of the spectrum at a time.

[14] For the spectrum to color conversion, we employ the Python package `color-science` (https://www.colour-science.org/).

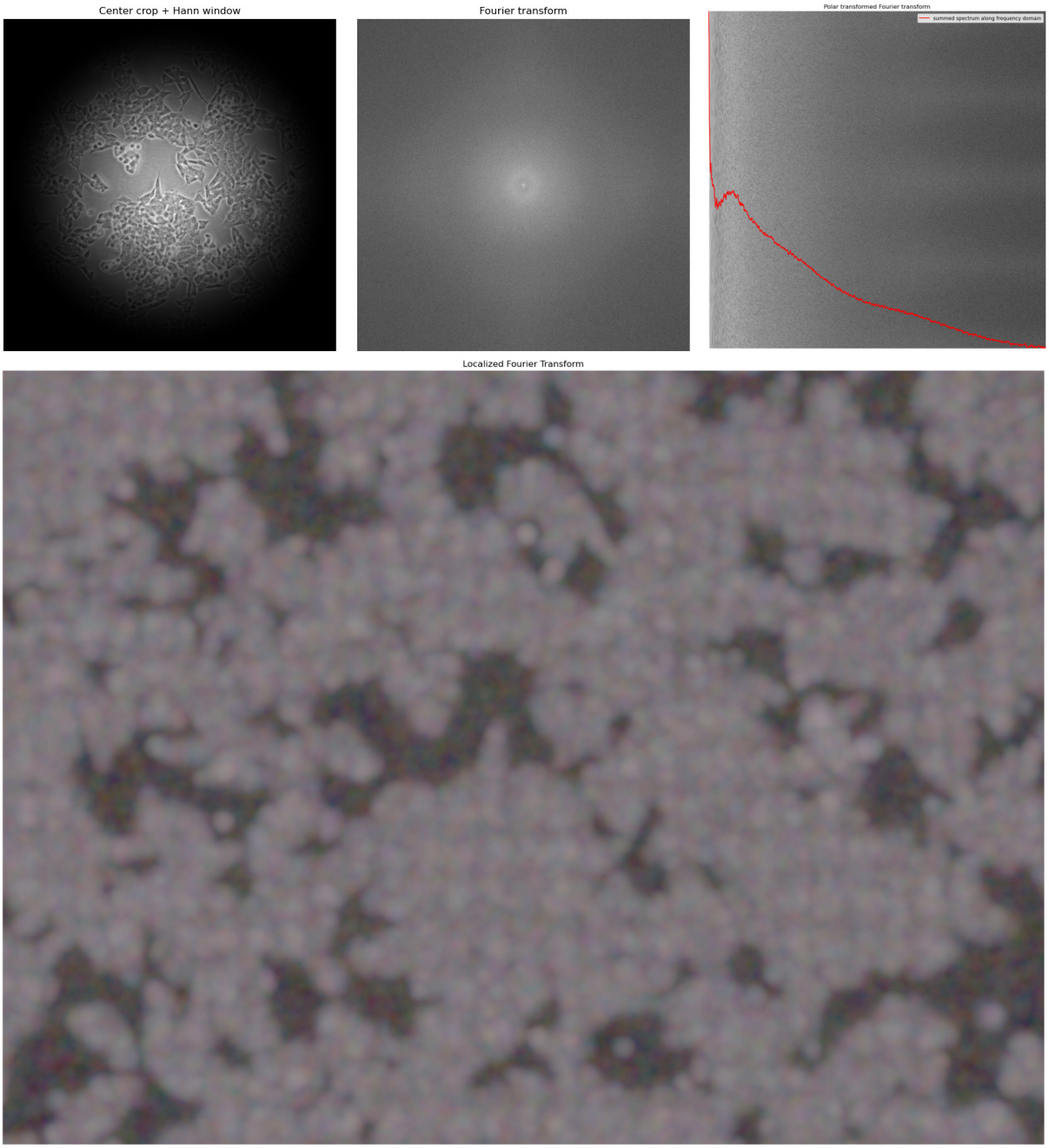

Figure 19: The Localized Fourier Transform (LFT). The upper row shows the main steps of computing the LFT from left to right. A part of the image was cropped and windowed (top left). A two-dimensional Fourier transformation was applied to the crop (top middle). The result was transformed into linear polar coordinates and summed along the angle (top right). The resulting amplitude spectrum of spatial wavelengths is converted to a color by treating it as a light spectrum. The computation was performed for every pixel in the image (bottom). The crop size in the top row is only exemplary, the results were computed using a 64×64 pixel crop around each pixel to extract only local features.

As a future direction, because it is not apparent with our method of visualization, it should be verified, if the spatial wavelength spectrum of any pixel is truly correlated with the belonging of a pixel to the classes of cell, boundary, or background. The fit quality of a linear regression model between spectral vector and pixel class could be used to measure the correlation. If very correlated, such a model would be an instance segmentation by boundary detection model (Section 5.3) without deep learning.

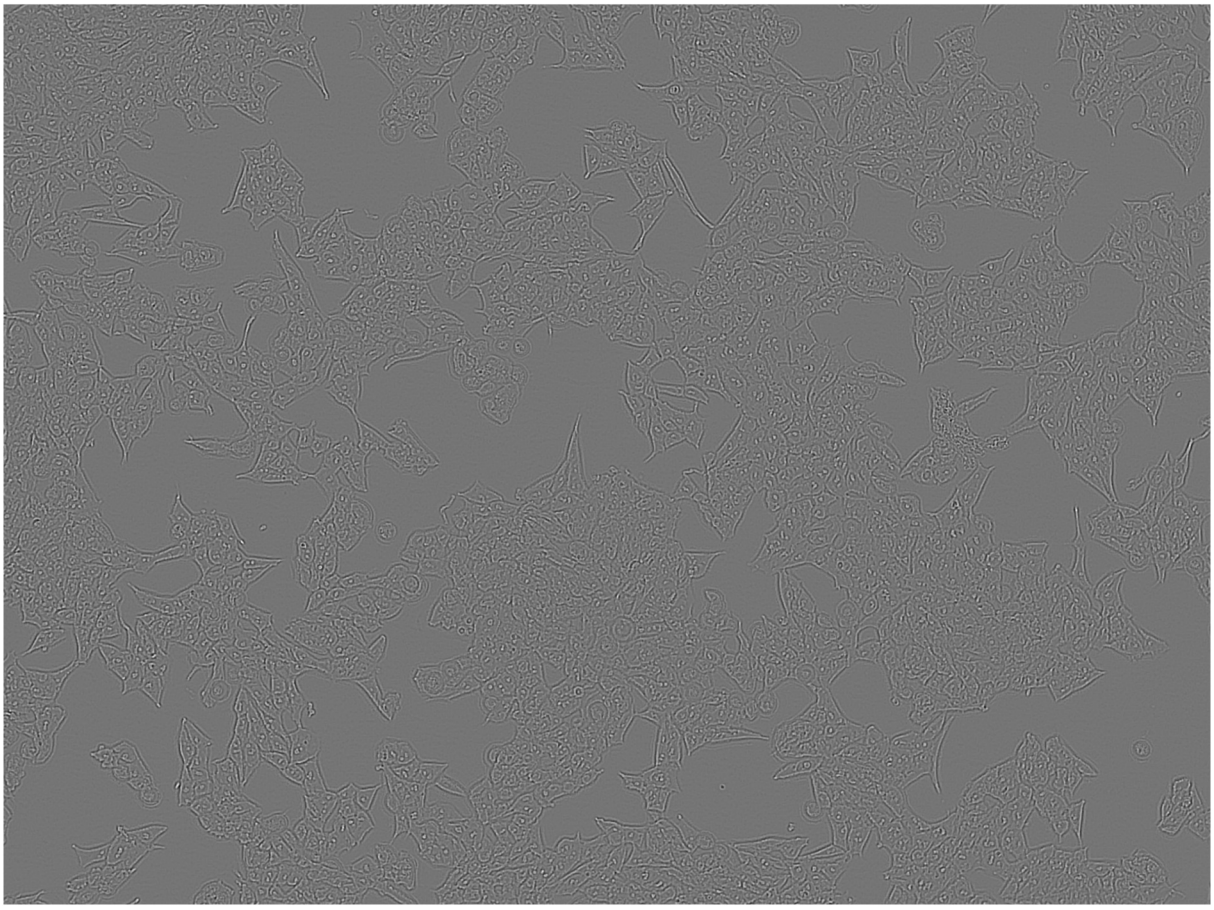

Figure 20: A microscopy image of cells from our dataset after applying a Butterworth band-pass filter. Cell boundaries are clearly pronounced, and the halo artifacts have been suppressed.

Due to the extremely unfeasible computational demand of computing a sufficiently wide-windowed FT for every image pixel, we abandoned this approach. However, we implement the idea of interactively selecting spatial frequencies with much less computational demand with real-time laptop performance in Section 9.1.5.3, not by displaying the spectrum of every pixel's neighborhood itself, but the reconstructed image after weighting its spectrum as a whole.

Butterworth Filter.

Figure 20 shows the image after applying a Butterworth band-pass filter as a sequence of a high-pass and a low-pass filter with cutoff frequencies ratios of 0.2 and 0.18 to the sampling frequency. Boundaries between cell material and background are highlighted and halo artifacts have been suppressed partly. However, intercellular boundaries are less pronounced and very hard to distinguish from the other intracellular structures.

Interactive Frequency Filter.

Figure 21 shows a selected state of an interactively controllable *free-form* Fourier filter. Using the panel in the top right, a generalization of a band-pass filter can be manually adjusted. All four panels display different methods to highlight image features using the filter. The red dots can be used to shape the red polynomial spline. The red spline defines the value along the radius of a radial symmetric filter mask. The filter mask is multiplied onto the two-dimensional Fourier spectrum of the input image. Thereby, a user can interactively design a custom shaped filter to pronounce any combination of spatial features. The other three panels show different ways to use this filter to pronounce image features. The most interesting is the one in the upper right. Many cells are clearly highlighted as dark round

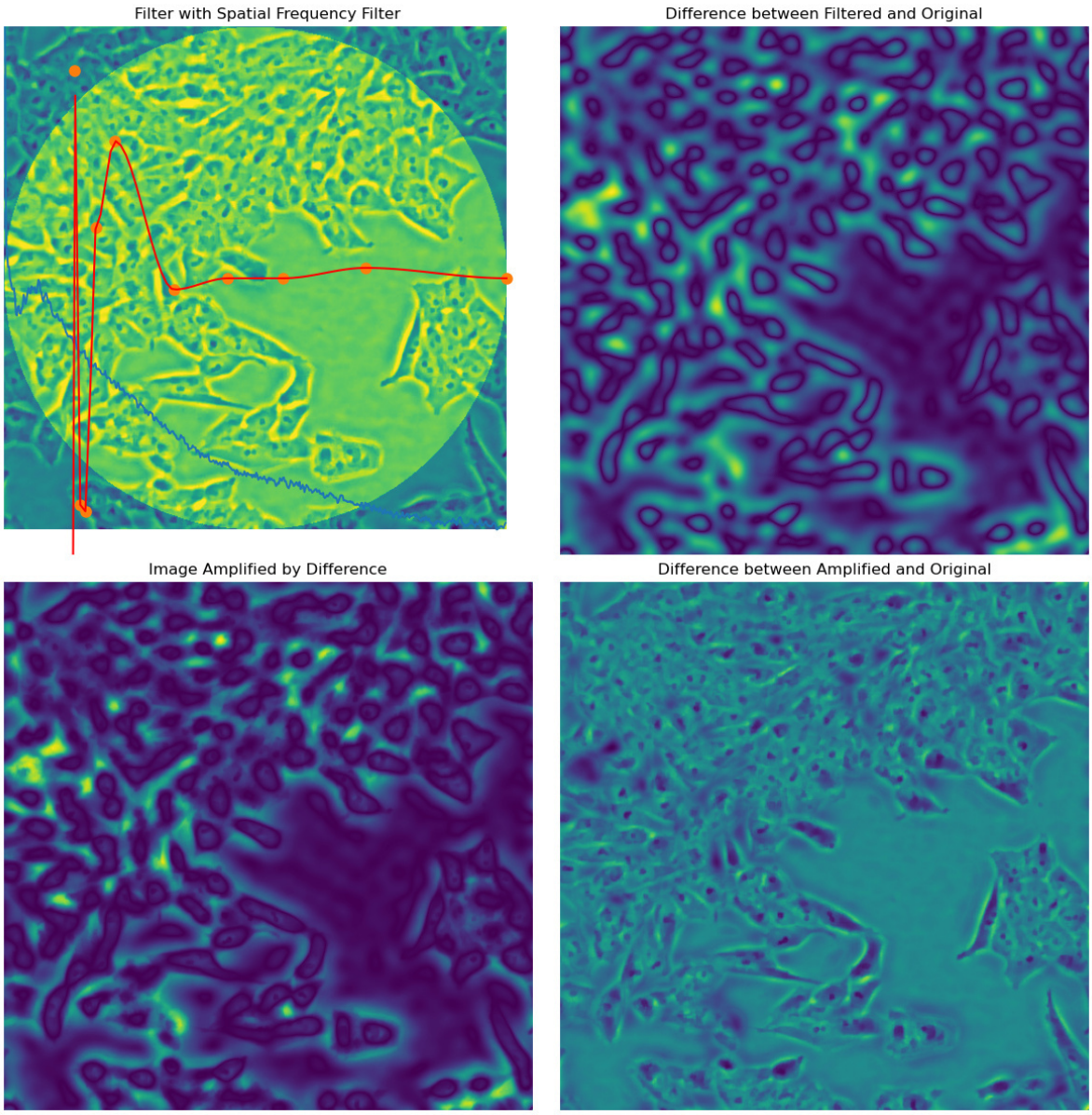

Figure 21: Screenshot from our interactively controllable Fourier filter. In the top left*, the background is the image after applying the current filter state. The red dots can be manually placed to shape a polynomial spline (red line), which weights the spatial frequencies in the image (blue line). To this end, the spline determines the intensity along the radius of a radial symmetric filter mask (yellow overlay) that gets multiplied onto the Fourier transformed image. | The* top right *is the difference image between the original image and the image after applying the filter. | The* bottom left *is the original image multiplied with this difference to highlight the selected features without suppressing their substructure. | The* bottom right *is the difference image between the original image and the amplified variant, and thereby highlights what the filter aimed to suppress.*

shape outlines. These outlines could be used to identify individual cells. However, many cells were not highlighted, while some clusters of cells got merged into one shape.

We conclude that this approach is promising, and maybe a better filter can be designed with more time, which can pronounce all cell shapes. However, because some intra cellular structures seem really similar to some clusters of cells in other image regions, it might not be possible to design a robust filter that works for all cells.

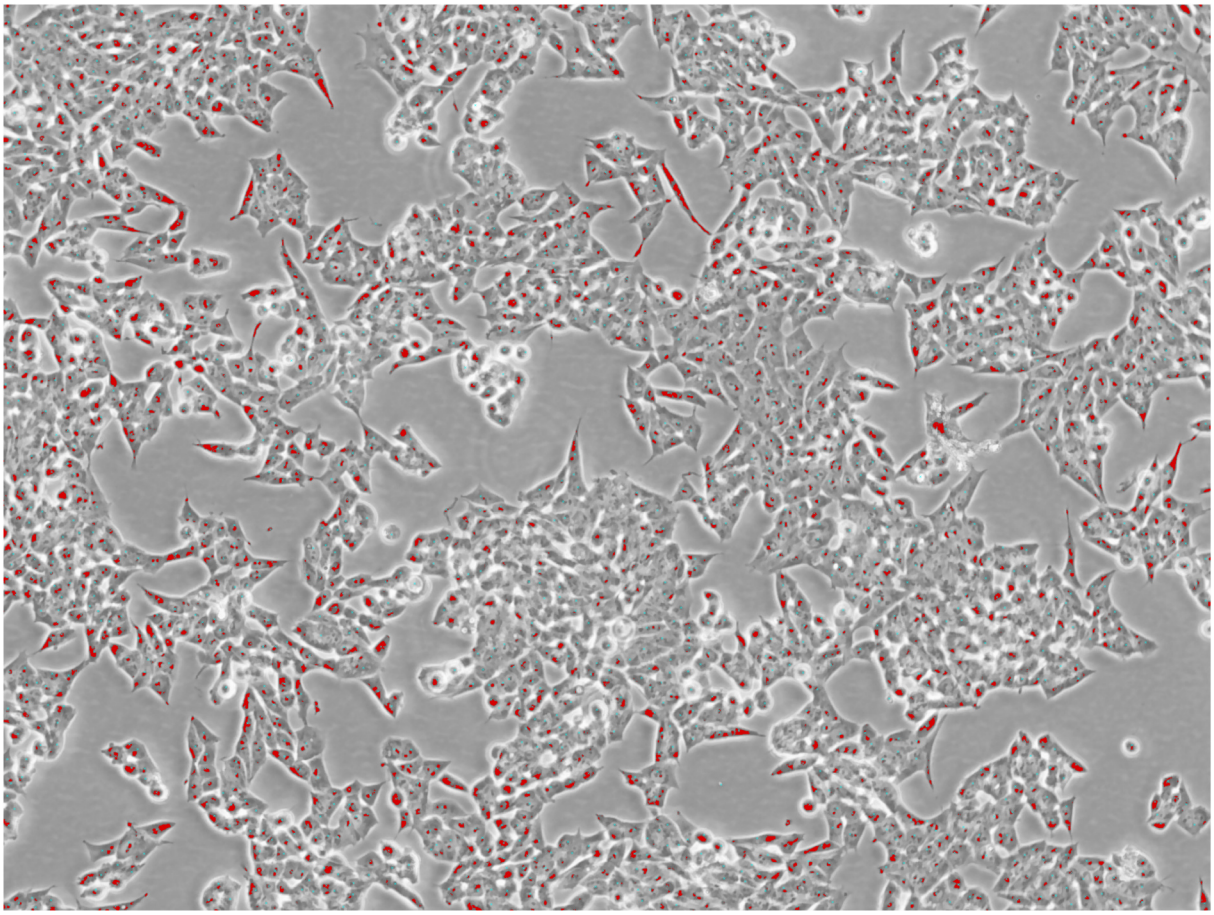

Figure 22: Thresholding. Dark pixels of $< 42\%$ intensity are highlighted in red. Ground truth cell annotations are in cyan.

9.1.6 Thresholding

Cells often contain dark spots. We aim to highlight them in Figure 22 by thresholding the image at 42% gray-scale intensity. Figure 22 shows, that some cells can be identified with a single threshold region. However, Some cells generally contain more and darker spots than other cells, such that thresholding both over-detects cells in some darker regions and misses many cells in some brighter regions. Therefore, it is not possible to count cells by counting the separate threshold regions. However, we develop the idea of identifying cells with intensity minima further in Section 9.1.7, where we determine local instead of global minima.

9.1.7 Minimum Filter and Local Minima

Cells contain a variable number of differently sized, darker substructures. In Figure 23, left, we aim to merge and pronounce them using a minimum filter with a kernel size of 5 pixels. The filter suppresses the complex substructures in cells, but also blurs the borders between many cells. We conclude that this approach is not suited to highlight cells. Many cells contain dark spots. However, the dark spots are not of the same absolute brightness across the whole image, such that some spots or even some cytoplasms are darker than other spots. Therefore, global thresholding by intensity Section 9.1.6 did not work well. Yet, the dark spots are always darker than their surroundings. To locate them, we computed local intensity minima across the image with a minimum distance of 20 pixels. The results are show in Figure 23, right. Many cells can be correctly identified with a local minimum. But because some cells are much larger or longer than others, they contain multiple minima, while smaller cells in dense clusters are under-detected. However, it was not apparent to us how this approach could be made more robust, therefore we abandoned it.

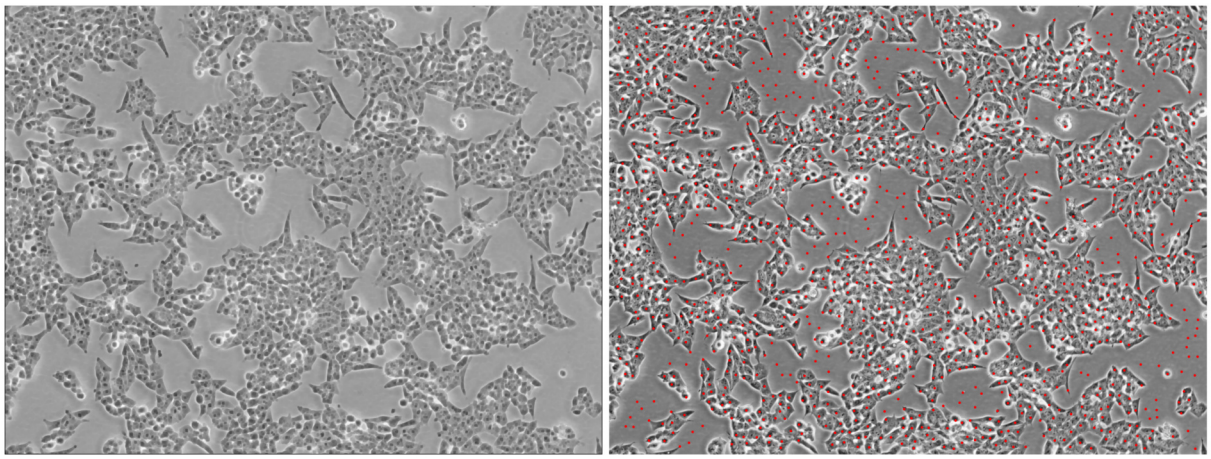

Figure 23: The image transformed using a minimum filter with a 5 pixel kernel (left). The locations of local intensity minima with a distance of at least 20 pixel (right).

9.2 Comparison of SAM and MicroSAM Backbones for Automatic Instance Segmentation

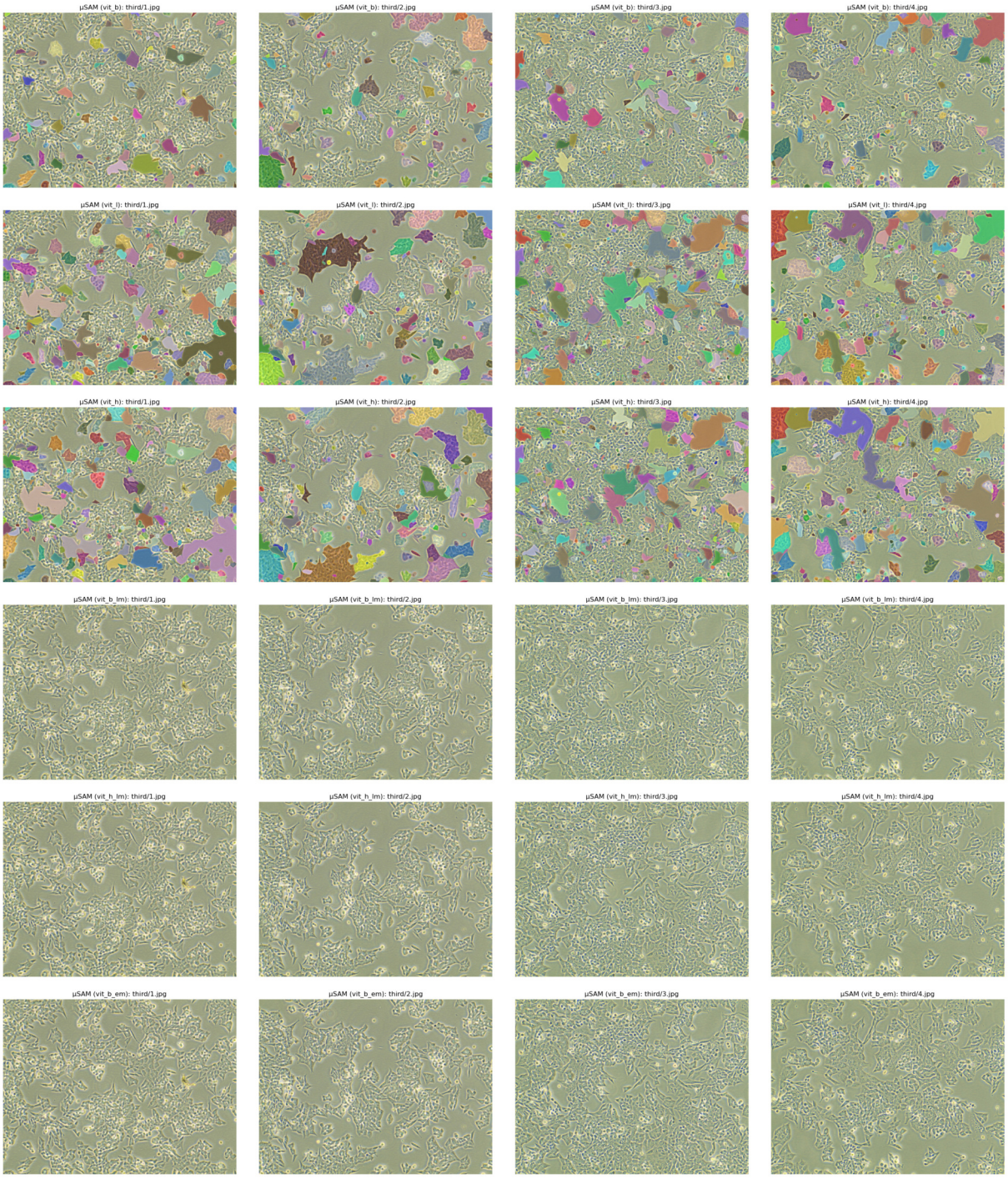

Figure 24: 0-shot instance segmentation results using different pretrained SAM and MicroSAM backbones. The columns show four of the collected images. The rows show the predictions using different backbones. The upper three correspond to the pretrained models by Kirillov et al. [11] sorted by size ascending downwards. The lower three correspond to the models fine-tuned on microscopy images by Archit et al. [19]. We include the two differently sized models pretrained on light microscopy (LM) and one pretrained on electron microscopy (EM).