MASARYK UNIVERSITY

FACULTY OF INFORMATICS

Segmentation and Tracking of Organoids in Brightfield Microscopy Image Data

Master's Thesis

LUCIA DUPKANIČOVÁ

Brno, Spring 2022

MASARYK UNIVERSITY

FACULTY OF INFORMATICS

Segmentation and Tracking of Organoids in Brightfield Microscopy Image Data

Master's Thesis

LUCIA DUPKANIČOVÁ

Advisor: Assoc. Prof. Martin Maška, Ph.D.

Department of Visual Computing

Brno, Spring 2022


FACULTAS ARTIS INFORMATICÆ
UNIVERSITAS MASARYKIANA

Declaration

Hereby I declare that this paper is my original authorial work, which I have worked out on my own. All sources, references, and literature used or excerpted during elaboration of this work are properly cited and listed in complete reference to the due source.

Lucia Dupkaničová

Advisor: Assoc. Prof. Martin Maška, Ph.D.

Acknowledgements

This work was partly supported by the Grant Agency of Masaryk University under the grant number MUNI/G/1446/2018.

I would like to sincerely thank my supervisor Dr. Martin Maška for the invaluable guidance in this project, namely his remarks and comments in the course of design of algorithms and experiments, and for his patience during the elaboration of this thesis.

I would also like to thank Dr. Zuzana Koledová for providing real image data analysed in this thesis and Dr. David Wiesner for providing synthetic datasets mimicking the real image data.

I am also most grateful for the support and encouragement that I received from my partner and family during my work on this thesis.

Abstract

Reliable automated image analysis has become crucial for biomedical research. Analysis of organoid images helps understand biochemical processes in human bodies and test influence of individual chemicals to human body tissues. To that end, the algorithms operating on the input image data must be reliable and robust to the high heterogeneity of the specimens and the input image data.

This thesis presents a family of deep-learning-based algorithms developed specifically for segmentation and tracking of an organoid of interest in 2D+t brightfield microscopy image data. The quantitative evaluation of these algorithms on two real image datasets shows that the best-performing algorithms are robust across different phenotypes of mammary epithelial organoids and thus can be used by the researchers to automate the image analysis process.

Keywords

biomedical image analysis, organoid, brightfield microscopy, image segmentation, object tracking, deep learning

Contents

List of Tables

List of Figures

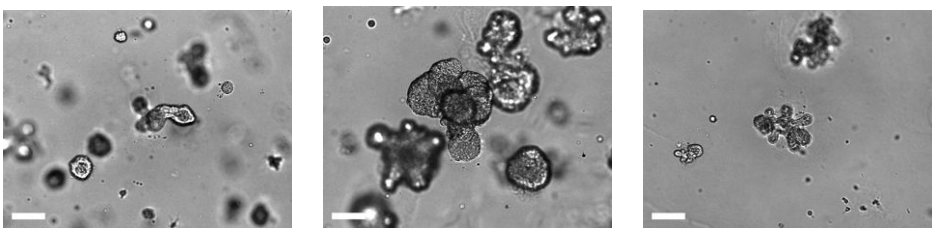
Introduction

Automated image analysis has become an essential component in biomedical research. With advances in microscopy and its increasing affordability, automated processing is necessary to facilitate the analysis of massive amounts of image data and to fully use the information in the raw image data [1].

Computerised analysis of organoid images plays an important role in a number of biomedical research applications, including developmental biology, disease biology, and drug toxicity and efficacy testing [2, 3]. Organoids are in-vitro cultures of real tissues with very similar function and cell organisation to the source tissue [2, 4]. They allow scientists to study tissue dynamics in a controlled environment [2]: organoids are grown freely in a medium, often supplied by a particular medication of interest, and imaged regularly to capture changes in their growth, morphology, or texture. Automated processing is thus crucial for fast and reliable analysis of organoid image data [2, 5]. The aim of the automated processing is to characterise organoids based on quantitative measurements of relevant biological properties [5, 4, 6].

In this thesis, I deal with automatic analysis of mammary epithelial organoids in brightfield microscopy image data (Table 1). The aim of this thesis is to detect and accurately segment organoids in temporal image sequences so that accurate quantitative measurements can subsequently be obtained and used to study organoid dynamics under the influence of particular growth-affecting proteins.

Table 1: Examples of the input image data analysed in this thesis. Contrast of the images was enhanced for visualisation purposes. The white bars correspond to 100 μm.



This task entails several challenges. In particular, organoids studied in this work exhibit high variability in morphology and texture – even a single organoid undergoes a considerable change in its characteristics throughout its lifetime. Organoids' location also changes due to their motion or manual intervention to imaging settings, which causes difficulties in organoid tracking. Organoid heterogeneity is further amplified by the imaging modality; due to imaging in a single focal plane, organoids can become partially or completely defocused and substantially change their texture characteristics. Finally, a number of organoids are present in each observed sample, which poses challenges such as identification of the organoid of interest and correct detection and segmentation of colliding organoids.

Recent works with a similar focus exist [2, 6, 7, 8], but they mostly do not concentrate on accurate organoid segmentation and high-resolution brightfield microscopy image data. Bian et al. [2] proposed a deep-learning-based framework for organoid detection and tracking in brightfield images without aiming at accurate segmentation. Lee et al. [7] provided a simple routine for segmentation and quantification of relatively homogeneous scenes in brightfield images. Hasain et al. [6] presented a framework for quantitative analysis of organoids based on their boundary coordinates, but the framework provides only a very simple routine for image segmentation. Bulin et al. [8] proposed an image analysis workflow that requires both brightfield and fluorescence image data and is designed for analysis of organoids with a considerably more homogeneous structure than organoids studied in this thesis.

This thesis is closely related to a manuscript [9] that I am the first author of and is currently under review for possible publication in a top-tier journal. In addition to the content of that manuscript, a comparison of several related approaches for reliable detection and accurate segmentation of the organoid of interest is presented, methodologies of different approaches are described, and a detailed overview of biomedical image analysis methods is provided. The method described in [9] is compared with other methods proposed in this thesis. A quantitative evaluation and discussion of the results is provided.

The thesis is organised as follows. In Chapter 1, a range of image analysis methods are described with highlighted differences between biomedical applications and general computer vision tasks. The review

of image segmentation methods is extended in Chapter 2 by describing the use of neural networks for biomedical image analysis; the strengths and weaknesses of neural networks in biomedical applications are discussed and several neural network architectures for image segmentation are presented. In Chapter 3, the input image data for this thesis are described in detail. A detailed description of methodology for both deep-learning-free and deep-learning-based approaches is provided in Chapter 4. Particular experiments as well as evaluation of the proposed algorithms can be found in Chapter 5. Finally, the achieved results and alternative approaches are discussed in Chapter 6.

An application with the implementations of the proposed algorithms was developed and is available as the supplementary material in the MUNI Thesis Archive[1].

1. `https://is.muni.cz/th/hpuxk/`

1 Biomedical Image Analysis

Biomedical image analysis is a special field of computer vision. Although most image processing methods are shared across tasks in real-world applications, biomedical applications exhibit several distinct characteristics that both constrain possible ways of processing and introduce assumptions improving specificity and sensitivity of algorithms [10, 11].

One of the distinctions of biomedical applications is the observed targets. Their characteristics impose a number of limitations on image acquisition and processing. These particularly include:

- distortions and blurs caused by living tissue deformability and motion during the acquisition process [12, 13, 11, 14],
- object occlusion and tissue overlay [12, 13],
- lack of certain information or bad-quality images caused by imaging technology characteristics [1, 15, 16, 17, 18], e.g. during live object scanning or scanning of inner layers of tissues.

Imaging instrumentation and image acquisition process can also introduce differences compared to general image processing. Each type of acquisition instrument, and thus each imaging modality, introduces specific kinds of noise, artefacts, or pitfalls [12, 19, 11]. Optical microscopy imaging serves as an example; since image intensities depend on exposure time and light source brightness [11, 16], a single microscope can produce considerably different image data of the same specimen.

Furthermore, multi-modal acquisition of a single scene is also common, as each modality captures specific kind of information [11, 20]. In such cases, specific processing techniques including image registration [21] and use of classifiers [10] might be necessary for automated analysis.

Another characteristic of biomedical image processing is the quantity of image data [15, 5]. Enormous numbers of biomedical images are recorded every day, often in high resolution [22, 12] or 3D [1, 4], including 2D and 3D temporal sequences [2, 23, 12, 20, 6]. The quantity and complexity of biomedical image data thus call for efficient and reliable algorithms.

At the same time, some biomedical image data are private due to their sensitive character or legal constraints [24, 25]. Moreover, only a small portion of the images is equipped with reference annotations suitable for development and evaluation of algorithms [26, 5]. Therefore, a number of publicly available datasets exist [27, 28, 29] that also provide reference annotations, benchmarks and comparison of the state-of-the-art algorithms. Federated learning for neural networks [24] also emerges as an approach to overcome the privacy of biomedical data as well as small amount of training data available in a single institution.

Biomedical image data, however, also entail certain rules or principles which can be viewed as an *a priori* knowledge and used to build reliable algorithms [11]. The more specific the task is, the more information can be assumed. Consequently, a higher amount of relevant *a priori* knowledge leads to better performing algorithms [10].

Generally, biomedical applications share a couple of requirements on the automated analysis and processing. These include:

- instance segmentation [17, 15] – identification of all object instances and their proper separation;
- accurate boundary delineation [15, 26, 5] – subpixel accuracy can be achieved e.g. using deformable models [30] or fuzzy segmentation methods [10];
- correct tracking results [13, 15] in temporal sequences;
- and efficient methods suitable for real-time applications [26, 5].

The following sections provide a brief categorisation and overview of basic image analysis and processing methods. Specifics regarding biomedical image data are highlighted when necessary.

1.1 Image Enhancement

Image enhancement [11] is a common preprocessing step employed prior to image analysis. It covers a range of techniques for instance for remapping of image intensities, noise reduction, or edge sharpening. When the goal is to revert a known image degradation, such as motion blur, the process is called image restoration [11]. Apart from enhancing the image for subsequent automatic processing [14], these techniques can be used to improve the image for a human observer. The goal

of image enhancement affects the choice of suitable filters [11]; for instance, while noise can substantially decrease performance of automated processing, it does not prevent humans from correct perception of the image contents in most cases.

In the biomedical domain, image enhancement is often crucial for performance of subsequent automated analysis [11, 14]. Images are often degraded due to the acquisition process, instrumentation setting, and physical limitations of acquisition, resulting in blur and artefacts in fluorescence microscopy imaging [31] or uneven background illumination in optical microscopy imaging [11]. In some cases, automated segmentation fails completely without proper preprocessing [32, 30]. In addition to general enhancement filters, there are also filters designed to suppress artefacts or noise specific to a particular modality [11].

1.2 Image Registration

Image registration [21, 11] is a process of determining correspondence between two images – which are seen as two different views of the same scene – using appropriate geometric transformations. The type of transformation sufficient to capture the relationship between the views depends hugely on the nature of the imaged objects [21]. Generally, non-rigid transformations must be employed for registration of images of deformable objects such as soft tissues [21, 33]. Image registration is crucial for certain applications as it allows one to combine information from images of the same object acquired using different modalities or acquired in different time [21].

Image registration methods can be divided into three categories based on the way they search for the transformation [21, 11]. Point-based and surface-based methods first find pairs of corresponding extreme points and corresponding surfaces, respectively, which are present in both views; other points and object interior, respectively, are then mapped using interpolation. Intensity-based methods use image intensities solely; the transformation is found by optimising a certain similarity measure such as correlation coefficient or entropy [21].

1.3 Feature Extraction

Feature extraction aims at extracting relevant information from the image. The obtained information can be viewed as a compressed representation of the image contents or can serve as a description of the input image, regions, or objects. The extracted features can be input to a classifier or constrain the subsequent processing steps [32, 34, 10].

Features can be extracted at several levels, from individual image elements to the whole image. They can be categorised as intensity-related and geometric [32, 35]. Intensity-related features include histograms and regional statistics such as entropy, moments, and texture measures. Geometric features are usually defined for regions and are based on their size (e.g. length, area), shape (e.g. curvature, circularity), location (e.g. bounding boxes, center of mass), and topology. As for the last category, connectivity can be measured on the image element level.

In practise, it may be necessary to perform noise reduction or image segmentation prior to feature extraction [32, 11]. This allows one to extract the same information irrespective of noise and artefacts.

It is often convenient to transform the image into a different space where certain characteristics can be found more easily. To name a few, Fourier transform [36, 11] is used for frequency analysis and to design image filters. Similarly, wavelets [11] allow one to study frequency spectrum, but also add localisation; they are typically used for multiscale analysis and to compress images. In comparison, Hough transform [11, 10] is a powerful tool for detecting parametrically represented shapes; the method works well even in noisy images.

Geometric information is usually extracted using mathematical morphology [37, 34]. Mathematical morphology provides a wide range of operators for binary as well as grey-scale images; the operators are specified by a shape probe that corresponds to desired shape and size constraints. A number of operators, such as openings or hit-or-miss transform, can be used to reduce noise or filter out unwanted objects based on their shape and size both before and after processing [20, 11]. Granulometry [34, 37, 38] measures properties of objects and texture via pattern spectrum. Skeletonisation helps classify shapes [11]. Finally, operators such as morphological reconstruction and top-hat transform are used to extract objects and image features [34, 38].

Although morphological operators are employed in many applications, most algorithms rely on the simplest ones, namely erosions, dilations, openings, and closings [34].

While morphological operators often analyse shapes during image processing, there is a number of measures that only quantify shapes after the computation has finished [11]. These include compactness measures (e.g. circularity), boundary regularity measures (e.g. boundary moments), aspect ratios and convex hull measures, Fourier descriptors, and connectivity measures (e.g. Euler number and skeleton properties).

Texture analysis also plays an important role [11]. Measures of texture coarseness, orientation, periodicity, and randomness can help with segmentation, classification, and interpretation of the image information. Many texture measures exist, for instance histogram statistics, analysis of frequency spectrum, and methods based on co-occurence matrices such as Haralick features.

Finally, color analysis can also be applied in some biomedical applications [11]. Depending on the imaging modality, image intensities in both color images and grey-scale images can bear particular biological significance [21, 33], or the color can be only a means of visualising several grey-scale images in overlay [18].

1.4 Image Segmentation

Image segmentation is a process of partitioning the image domain into a set of semantically consistent regions [10]. In biomedical applications, the desired regions represent real-world objects and thus recognition of the object class is also usually required [10, 11, 35]. In addition to semantic segmentation, the instance segmentation – identification and separation of individual class instances – is often required [17]. Image segmentation is usually followed by quantification of individual regions [38].

Most biomedical applications are specific enough to imply task-dependent *a priori* knowledge [10], such as specification of object location, texture, or shape as well as knowledge about the imaging process [1]. Generally, the *a priori* knowledge is used to increase the performance of algorithms [11]. It may also be used to build heuristics

in the algorithm [35] and can be embedded so subtly in the algorithm that it is hard to recognise it without deeper understanding of the task [10, 32, 35].

Algorithm performance, namely in biomedical applications, can also be improved by combining information from various modalities. In this case, segmentation is often performed using a classifier that considers features extracted from the co-registered images rather than raw image data [10].

In fact, it is sometimes inevitable to employ either *a priori* knowledge or a multi-modal algorithm to achieve reasonable results using automated processing [10, 35]. Biomedical images exhibit a high variability of image quality [39, 18, 30] and the observed objects [35, 30, 20] – variability of objects of the same kind as well as occurrence of unknown objects. Although most unsupervised segmentation methods rely hugely on *a priori* knowledge [10], they still require a specific task domain to produce reasonable results due to this variability [11]; and thus supervised machine-learning methods have become crucial in image segmentation, the leader being convolutional neural networks [17, 40, 11, 3].

There is a range of methods that are used to build segmentation algorithms [10]; a combination of several methods is used in most cases [29, 41, 11]. State-of-the-art algorithms are mostly based on deep-learning approaches, but still tend to combine them with traditional methods [42]; among the traditional methods, the basic ones are still the most frequently used [20].

The simplest method is thresholding [10, 11]; pixel or voxel classes are determined by comparing their intensities to one or more threshold values. Although a suitable threshold value might be known in advance in some cases, it is usually estimated by histogram analysis.

The second category are edge-based methods [10]. Edge detection, however, yields only candidates for region boundaries; these have to be filtered, thinned, and interconnected to obtain closed boundaries of individual regions. Expected gradient magnitude or orientation can be used to constrain the result if they are known *a priori* [35].

Thirdly, there are region-based methods [10, 11] that construct connected regions iteratively based on a set of homogeneity rules. This category includes region growing (iterative region growing from an input set of region seeds), region splitting (iterative splitting of

heterogeneous regions starting from a single region), and the split-and-merge method (region splitting with possible merging of similar adjacent regions).

Mathematical morphology can also be used for image segmentation [34, 38]. Individual morphological operators can be part of a segmentation pipeline to reduce noise, extract edges, or filter out components by shape [38]. In addition, skeleton by influence zones and watershed segmentation can be used directly for image segmentation. The latter is often considered a region-based method [20].

The next category are energy-minimisation methods [42, 14]. Segmentation result depends on the energy function definition and computation initialisation. This category includes deformable models [30, 14] that are based on evolution of a contour representing object boundary. Evolution is driven by minimising energy function consisting of internal and external terms; while the internal forces enforce contour smoothness, external forces drive the contour towards object boundaries. It is possible to embed shape priors to the internal force term to constrain global shape properties of the contour [30, 42, 14].

Graph cuts [43, 42] are another example of energy-minimisation methods. A graph representation allows one to encode regional, boundary and shape constraints in the graph edges. Optimal segmentation is found using a maximum flow algorithm. Graph cuts are capable of multiclass segmentation [42, 14] as well as segmentation of multimodal images [42].

Finally, one can employ classification methods [10] that assign class labels to the whole image or individual image elements based on features extracted from the image. The features form a feature vector that represents the image or individual image elements; classification methods then search for clusters in a feature space as each cluster represents a single class. Unsupervised methods rely solely on analysis of the feature space to identify clusters and their boundaries. In comparison, supervised methods learn to identify clusters in a training phase according to a labeled input.

In addition, fuzzy classifiers [10, 44] account for variability in biomedical images [35]. They assign a relative strength of each class to image elements [11]. This allows for subpixel classification and copes with image classification uncertainty [35]. Fuzzy methods can also be enhanced by incorporating domain knowledge constraints [35].

Some classifiers, such as neural networks, are capable of implicit feature extraction and can thus work on raw image data [40, 45]. However, most classifiers require already extracted feature vectors as an input, so proper feature selection is vital [32]. It is most often based on the measurement of feature discriminatory power; the calculation is either constrained by the former choice of classifier, or is data-driven and can be used to choose the suitable classifier as well. These measurements result in a ranking of features, which may determine a subset of features to use, or provide a guidance to relative importance of features for a human expert.

None of the above-mentioned methods is limited to the processing of raw images. Not only is it often beneficial to preprocess the input image to obtain a better segmentation result, it may be also necessary to compute a certain image transformation prior to the processing [10, 11, 20]. These include edge detection or watershed transform driven by the gradient magnitude image as well as thresholding of texture measure image.

1.5 Object Tracking

Object tracking is a process of following an object in a temporal image sequence [17]. As a result, object motion velocity or trajectory can be inferred [29]. In biomedical image analysis, tracking is often accompanied by the construction of lineage trees [29] that capture correspondences between instances in consecutive frames as well as other important events in objects' lifetime, such as cell division. The algorithms have to cope with events such as object occlusion and clustering as well as objects entering and leaving the scene [15].

There are two categories of tracking methods: tracking by detection and tracking by model evolution. Tracking by detection methods [29, 1] need a pre-computed segmentation or detection of each frame to establish association between corresponding instances in consecutive frames. Although the previous frames are not directly involved in the computation, they contribute to the result in a recursive manner [13]. On the contrary, tracking by model evolution methods [29, 13, 1] carry out image segmentation and object tracking simultaneously. This tracking paradigm is typically based on the evolution of active

contours. Since an active contour in frame t is initialised by the contour in frame $t-1$, these methods require strong spatio-temporal consistency between frames [29].

2 Image Segmentation via Neural Networks

Neural networks are representation-learning methods for image classification [40, 11, 46]. When presented a set of learning samples, they are able to extract relevant features from the data and thus learn to provide expected output for the samples. Their strength lies in implicit data representation [40] and their ability to generalise beyond the properties of the learning samples [5]. Classification can happen at different levels – from individual image elements to the whole image [39].

There is a number of different network architectures; deep neural networks with a number of hidden layers are usually used nowadays [11]. Individual layers allow the network to learn very complex functions by representing relevant features at multiple abstraction levels [40, 17].

Neural networks can learn either in an unsupervised [47, 46], a semi-supervised [48], or a fully-supervised manner. When using the fully-supervised approach, the learnt representations rely hugely on the training data. It is crucial to create a balanced dataset with a sufficient number of representative images for each class [17, 26, 5] and precise class labels [5]. Including images of different quality [5] as well as suitable data augmentation [5, 41] can also improve the network performance.

A suitable training dataset may, however, be a bottleneck, namely in biomedical applications [26]. Annotating images manually is very time-consuming [2, 5] and often yields imprecise or biased results [5, 49]. Several approaches may be chosen to tackle this limitation: the augmentation of the available training dataset [50, 51]; employing and unsupervised or a semisupervised model [46]; or creating a synthetic dataset that mimics the real image data [50, 22].

The following sections present several deep neural network architectures used for biomedical image segmentation.

2.1 Convolutional Neural Networks

Convolutional neural networks (CNNs) [52, 40, 53] are a natural choice for image classification and segmentation. A CNN processes its

input using convolution layers and gradually downsamples the image using pooling layers. Convolution layers learn distinctive features from the data while downsampling allows the network to learn these features at multiple levels of detail [17]. Thanks to pooling layers and the terminal fully-connected layer, the image can be reduced to a single vector representing class probabilities.

Biomedical applications, however, present specific challenges for the networks. Firstly, there is usually a considerable amount of image data available, but very few images are equipped with accurate reference annotations necessary for the training [26, 11, 5]. Secondly, many biomedical applications require classification at the image element level instead of classification of the whole image [17]. And thirdly, instance segmentation is a frequent requirement [17, 15], and thus objects of the same class must be carefully and accurately separated.

The following subsections present three state-of-the-art CNN architectures that output pixel-level classification of input images. These architectures were selected to show different approaches to the same task. All of them were successfully used in biomedical applications.

2.1.1 U-Net

U-Net [51, 54] is a fully-convolutional network of the encoder-decoder type designed for the instance segmentation of biomedical images. Its architecture (Figure 2.1) consists of two main parts: a contracting and an extensive path. The contracting path consists of several blocks with two convolution layers with non-linear normalisation and a max-pooling layer. On the other hand, the extensive path contains several blocks consisting of an up-sampling layer and two convolution layers with non-linear normalisation.

During up-sampling, a corresponding feature map from the contracting path is also retrieved, which contributes to accurate localisation of features [51]. Max-pooling and up-sampling layers gradually change the level of detail of the image representation, thus enabling the network to learn features at different scales [23].

U-Net also provides a mechanism to compensate for a small number of annotated training samples, typical for biomedical applications. Training dataset undergoes extensive data augmentation that creates images resembling the real image data [51, 54]. Such data augmenta-

tion helps to train a reasonably-performing network from scratch with only a few training samples [51, 55]; even fewer samples are necessary if a previously trained model is later fine-tuned with a new dataset [54]. Implicit data augmentation is also achieved via drop-out layers in the contracting path [51, 40].

U-Net has been adopted by many researchers to solve biomedical image segmentation tasks. Recent works deal with diverse tasks such as segmentation of vesicles in transmission electron microscopy (TEM) images [56], segmentation of cells in microscopy images [26], or nucleus segmentation in histology images [57]. Most of the works also propose modifications to the original U-Net architecture: [56] added residual layers to all levels on both paths; [58] proposed adding skip connections from a contracting block to all lower-level extensive blocks; [59] placed transformer modules between the contracting and extensive paths to exploit dependencies between consecutive frames in a 3D sequence; [60] replaced skip connections and up-convolution with up-sampling driven by indices remembered from the concate-

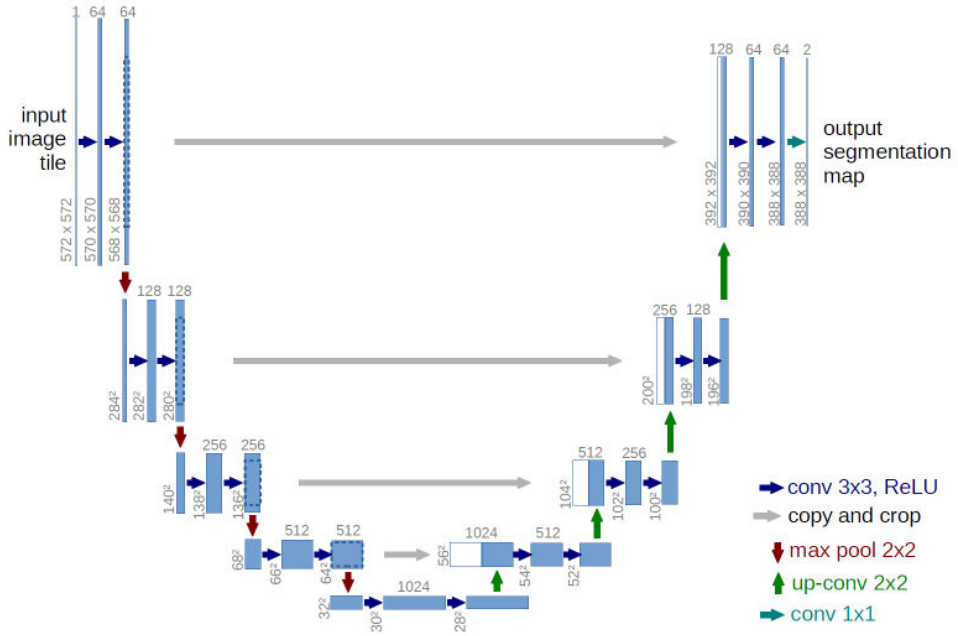


Figure 2.1: U-Net architecture. Taken from [51].

nation path; and [26] equipped the skip connections with additional convolution layers and residual connections and introduced a Classifier and Localizer module that operates alongside the extensive path to suppress false positive and false negative outputs. A U-Net architecture for processing volumetric image data has also been created [61] by extending all operations to three dimensions.

2.1.2 DeepCell

DeepCell [41] is a framework for cell segmentation. Unlike U-Net, DeepCell architecture contains only the contracting path. Pixel-wise classification is achieved by interpreting the network output as a class prediction for the central pixel only [5]. The input image is thus processed by small overlapping patches. This leads to lower memory requirements than with U-Net, but requires a careful choice of the patch size [41].

Similarly to U-Net, it is possible to train the network with a relatively small number of annotated images with suitable data augmentation [41, 5]. Segmentation accuracy can be increased by training several models and averaging their results [41, 5]. The authors of [41] also suggest that postprocessing of network's predictions is necessary to improve the results, and recommend to employ active contours for that purpose.

DeepCell was successfully used for segmentation of cell cytoplasm and cell nuclei in phase and fluorescence microscopy images [41, 5].

2.1.3 Mask R-CNN

Mask R-CNN [62] is a region-based convolutional network designed for segmentation and classification of general images. Its distinction lies in two-stage processing. Firstly, a set of proposed object bounding boxes is established using a fully-convolutional network. Each bounding box determines a cropped image that is fed to a CNN to output pixel-wise segmentation. Classification of the object is performed in a separate pipeline.

Mask R-CNN's performance was demonstrated on general images in the COCO challenge [63]; later, it was successfully used e.g. for nucleus segmentation in fluorescence and histology images [64]. It

was observed that R-CNN provides less accurate segmentation than U-Net, but performs much better in separation of clustered object instances [64].

2.2 Recurrent Neural Networks

Although CNNs are able to identify and learn many features, they cannot fully exploit long-range dependencies neither in a single image, nor in an image sequence [59]. However, such information might be of great importance for accurate segmentation [59]. Although there are three-dimensional CNNs for volumetric data [61, 65], they suffer from high resource requirements and their possibilities are therefore limited. This naturally leads to the use of recurrent neural networks (RNNs) [40] that are well suited for detection of spatial as well as temporal patterns in sequential image data.

Given an image sequence, an RNN processes it by individual frames [40]. The network maintains a hidden state that accumulates information from the previously seen images. Previous frames thus contribute to classification of the current frame. The hidden state is being updated with every frame.

One of the most important RNN architectures is the long short-term memory (LSTM) [66, 40]. Its hidden state is implemented as a memory cell. The memory and the cell's previous output as well as new input are processed by non-linear gating units that control the flow of data through the network, and non-linear activation functions. This architecture has been designed specifically to tackle the vanishing and exploding gradient problem that had prevented simple RNNs from being useful in practice [67].

Several concepts have been proposed to combine the U-Net architecture and the RNN scheme [23, 68]. Generally, the U-Net architecture is modified to contain convolutional LSTM (C-LSTM) blocks. While Arbelle and Raviv [23] propose to substitute convolution layers with C-LSTM layers either in the contracting path, the extensive path, or both, Payer et al. [68] suggest to insert C-LSTM blocks to skip connections between the two paths.

In biomedical applications, RNNs have successfully been used for example to segment and track cells in microscopy images [68, 23].

2.3 Adversarial Neural Networks

A neural network architecture inspired by the generative adversarial networks (GANs) for the instance segmentation of biomedical images has also been proposed [69]. Similarly to GANs, two competing networks are trained simultaneously: an estimator learns to output accurate segmentation of the input images whereas a discriminator learns to differentiate between human annotations and the estimator's output given the image and its segmentation. The authors claim that even a small number of annotated training samples is sufficient to train a well-performing model. Adversarial networks have been used e.g. to segment cells in fluorescence microscopy images [69] and nuclei in histopathology images [70].

2.4 Cellular Neural Networks

Cellular neural networks (CeNNs) [71] combine concepts of neural networks and cellular automata, resulting in a regularly-spaced grid of cells where each cell is connected only to a small number of neighbouring cells and represents the same non-linear dynamic system. Despite local connections, global dependencies can be learnt via information propagation [71, 72]. Cellular networks are capable of parallel processing [71, 73], real-time processing [73, 74, 75], and processing of noisy data [73, 75].

Cellular networks have been used to solve a wide range of image processing and pattern recognition tasks, such as pixel and image classification [72], image enhancement [72, 76], feature extraction [72, 73], edge detection [75, 76] and colour image processing [75]. In fact, CeNNs perform well at many image analysis tasks whereas convolutional neural networks are suited mostly for image classification and segmentation [74, 41]. An exmaple of use of CeNNs in the biomedical world is a fuzzy cellular neural network used for white blood cell detection [77].

3 Image Data

An overview of the input image data is provided in this chapter. Four datasets – two real datasets and two synthetic datasets – are described and their differences are highlighted.

3.1 Real Image Data

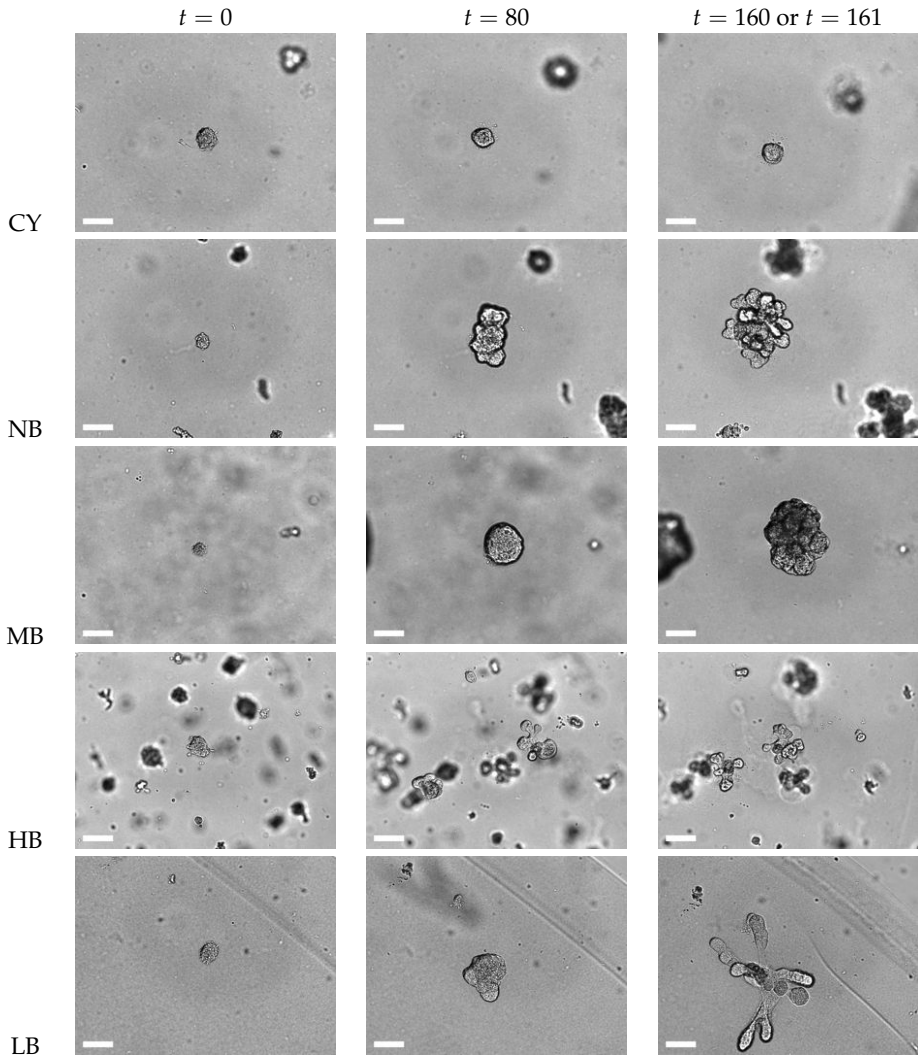
Input image data [9] capture the development of mammary epithelial organoids. The organoids were cultivated in controlled environments with varying levels of growth-affecting substances. Individual types of environments impact organoid characteristics and thus determine organoid phenotypes: cysts (CY), normally branching (NB) organoids, massively branching (MB) organoids, hollowly branching (HB) organoids, and long branching (LB) organoids (Table 3.1).

The CY phenotype is the only phenotype not induced with the growth-affecting factor. Shape, size, and texture of CY organoids thus change slowly; the organoids maintain approximately circular shape during the whole experiment and do not form any branches. Other organoid phenotypes have very similar appearance in the beginning of an experiment, but start to form branches over the course of imaging. The NB and MB organoids form a huge amount of short thin and thick branches, respectively. The LB organoids evolve in a similar manner but form substantially longer and thinner branches. The HB organoids start to grow branches sooner than the other phenotypes and the branches have different texture that is less distinct from the background. The LB and HB organoids generally form less branches than the NB and MB organoids. The MB organoids retain roundish shape during the whole experiment due to the thickness of their branches, while the NB, HB, and LB organoids have much higher perimeter-to-area ratio in the end of the experiment.

The specimens were imaged every hour for several days using an Olympus IX81 microscope with a dry $10\times/0.40$ objective lens, forming a sequence of two-dimensional brightfield microscopy images. Each sequence contains a single organoid of interest, yet there may be other organoids and structures present in its surrounding. Imaging settings were adjusted regularly during the experiment so that the organoid of

interest is mostly focused and approximately centered in the field of view over the course of imaging. As a result, abrupt changes in focus

Table 3.1: Real image data. Top to bottom: the CY, NB, MB, HB, and LB organoid. Left to right: the first frame, approximately the middle frame, and the last frame of individual sequences. Contrast of the images was enhanced for visualisation purposes. The white bars correspond to 100 μm.



and location of objects in the field of view might be present in the image data.

Five sequences were recorded for each phenotype. Each image sequence contains from 153 to 162 frames of 1344×1024, 1336×1024, or 1344×1014 pixels in size with pixel size of 0.647×0.647 µm.

Additionally, ten images from each sequence were randomly selected and annotated by three experts. These manual annotations were combined by majority voting to provide reference annotations for segmentation of the selected images. Annotators concentrated on segmentation of the organoid of interest only, classifying the pixels into two classes: *background* and *organoid*. In addition, one annotator created a detection marker for each frame in all real sequences (Table 3.2).

The CY, NB, and MB sequences form a *target dataset* (*TD*) and were used to optimise algorithm parameters as well as to evaluate algorithm performance. The HB and LB sequences form a *robustness validation*

Table 3.2: Comparison of three manual annotations for segmentation, the reference annotation and the detection marker on a single frame from a LB sequence ($t = 154$). The annotations and the detection marker are outlined in yellow. Contrast of the input image was enhanced for visualisation purposes. The white bar corresponds to 100 µm.

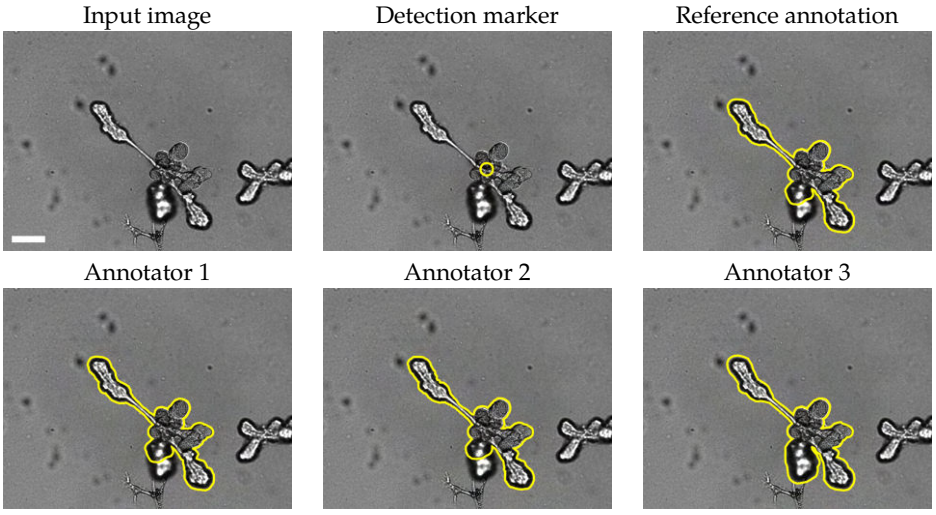
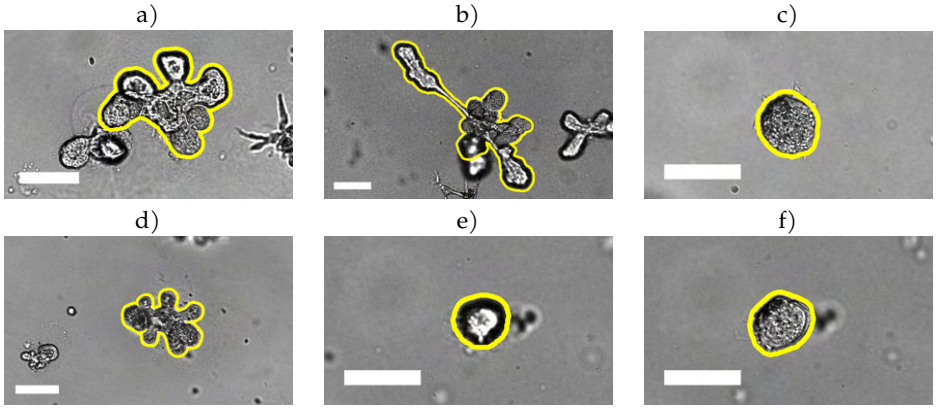


Table 3.3: Examples of challenging situations: a) two focused organoids in collision; b) organoid with focused body and defocused branch in collision with a defocused organoid; c) epithelial structures attached to the organoid's surface; d) a HB organoid with different texture and out of the focal plane; e+f) two consecutive frames that capture the organoid of interest before and after manual intervention to imaging settings. Yellow outline denotes the reference annotation. All images are cropped to show the necessary context. Contrast of the input images was enhanced for visualisation purposes. The white bars correspond to 100 µm.



dataset (*RVD*) to measure algorithm ability to generalise on unknown organoid phenotypes.

Common challenges (Table 3.3) include colliding organoids, epithelial structures attached to organoid surface, abrupt changes in focus and location of objects due to the manual interventions, presence of several focused organoids in a single frame, and high variability in organoid texture and shape among different phenotypes.

3.2 Synthetic Image Data

Synthetic 2D+t sequences plausibly mimicking real image data were generated [9] using a conditional generative adversarial network (GAN) and provided to me to facilitate the development of a deep-

learning-based algorithm described in Section 4.2. This section describes two synthetic datasets that were created in this manner.

The *G1 dataset* (Table 3.4) was generated by a GAN with pix2pix architecture [78]. The GAN was trained with images from the *target dataset*; semantic masks for the training were obtained using a deep-learning-free algorithm described in Section 4.1. Five sequences with the CY organoids and five sequences with the NB organoids were generated, each with 153 to 162 frames of 1024×512 pixels in size. The sequences were generated using the semantic masks from the training dataset. Individual frames were generated separately. Whereas morphological and positional inter-frame dependencies were imposed by the semantic masks, temporal texture consistency was not enforced.

The *G2 dataset* (Table 3.5) was generated by a GAN with pix2pixHD architecture [79]. The network was also trained with images from the *target dataset*, but the semantic masks were extended to contain two additional classes: *dark blurred structures* (*DBS*) and *bright blurred struc-*

Table 3.4: Example of the *G1 dataset* – three consecutive frames from two synthetic sequences. Top to bottom: the CY and NB organoids with their reference two-class annotations.

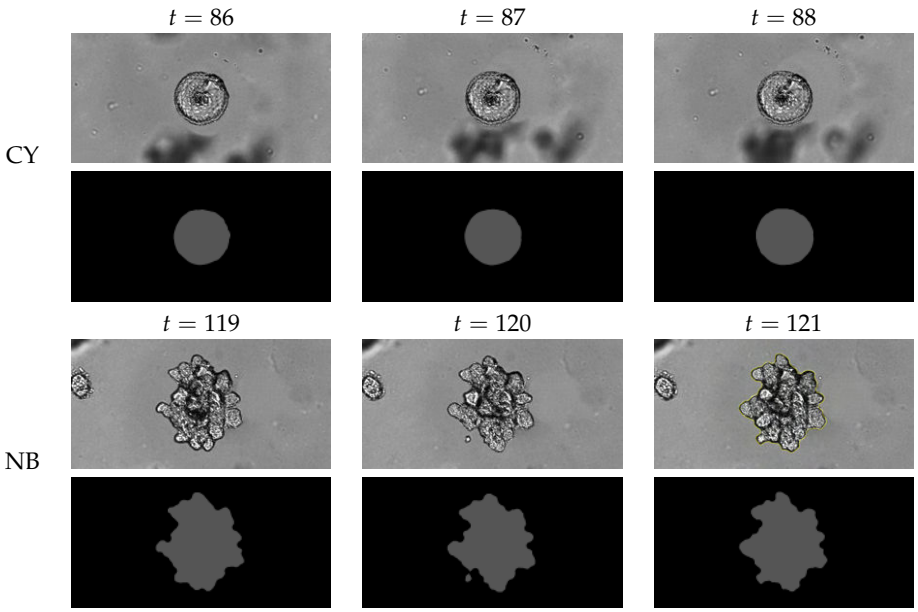
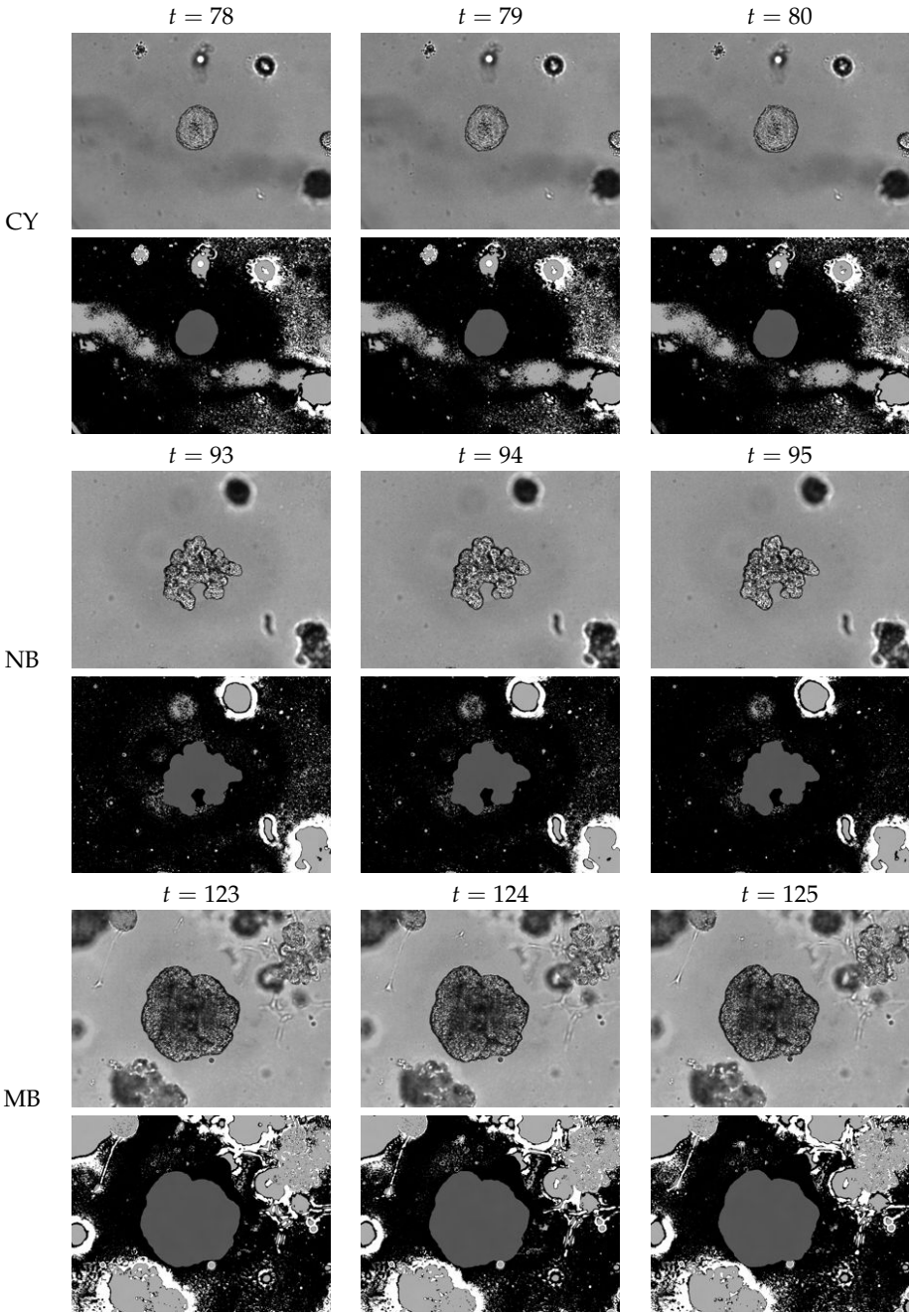


Table 3.5: Example of the *G2 dataset* – three consecutive frames from three synthetic sequences. Top to bottom: the CY, NB, and MB organoids with their reference four-class annotations.



tures (*BBS*). These were inferred by median-intensity-driven thresholding of the input images [9]. The training dataset thus classifies pixels into four classes. The *DBS* and *BBS* classes helped diminish the difference between texture evolution in synthetic sequences and in the real image data. Visual comparison of the *G1 dataset* and the *G2 dataset* can be done using Tables 3.4 and 3.5. Quantitative evaluation of the *G2 dataset* can be found in [9]. Five CY, five NB, and five MB sequences were generated in this manner, each with 153 to 162 frames of 1344×1024 pixels in size.

The semantic masks used to generate images also serve as reference annotations. The reference annotations are therefore fully defined for all synthetic images and classify pixels into two and four classes, respectively.

4 Methodology

Methodology for analysis of the input image data is described in this chapter. The aim of this thesis is to develop an algorithm for reliable segmentation of the organoid of interest in each input sequence. The algorithms were thus developed specifically for the input image data described in Section 3.1 using their characteristics to improve the algorithm performance. A single binary mask is computed for each input frame to classify image pixels as *organoid* or *background*.

A deep-learning-free (DL-free) baseline algorithm that was used to generate organoid mask for training of the conditional GANs for synthetic data generation (Section 3.2) and to set a baseline performance for evaluation of deep-learning-based algorithms (Section 4.2) is described in Section 4.1. A deep-learning-based (DL-based) approach is described in Section 4.2. The implementations of both approaches are available in the MUNI Thesis Archive.

4.1 Deep-Learning-Free Segmentation

The deep-learning-free algorithm can be divided into several parts (Figure 4.1). Firstly, a coarse segmentation of the organoid of interest is computed and used to establish a region of interest for subsequent segmentation refinement. In the refinement step, segmentation is performed according to certain texture characteristics of the organoid of interest. Outputs of both steps are combined and filtered to obtain a binary segmentation mask.

The baseline algorithm relies hugely on the *a priori* knowledge about the input image data. The most important assumptions that allow this algorithm to be useful in practise are:

- tracking of a single organoid in each sequence,
- organoid being a single object without holes,
- organoid having a fine-grained structure with highly varying pixel intensities,
- organoid being centered in the image plane,
- and slow motion of the organoid.

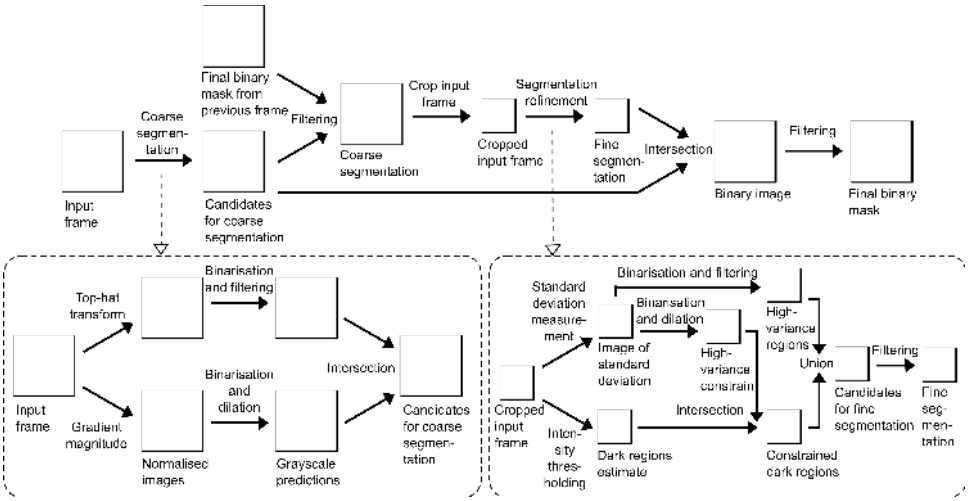


Figure 4.1: Deep-learning-free pipeline for 2D+t sequence segmentation.

On the contrary, some temporal dependencies between consecutive frames, such as smooth shape evolution, are not exploited in this algorithm.

4.1.1 Coarse Segmentation

Firstly, a coarse segmentation is computed (Table 4.1) to define an axis-aligned bounding box of the focused organoid. The bounding box defines an area where the subsequent steps (Sections 4.1.2 and 4.1.3) are performed to accelerate the subsequent computation.

The input image data present a clear pattern: the background is relatively smooth while the intensity of organoid pixels varies considerably. Therefore, each input frame undergoes two coarse-segmentation pipelines.

Firstly, black top-hat transform of the input frame is computed with closing with a circular structuring element (CSE) with diameter d_{TH}; this image is thresholded with the triangle method and morphologically filtered to refine the component shape.

In the second pipeline, gradient magnitude of the input frame is computed, thresholded with the triangle method, and morpho-

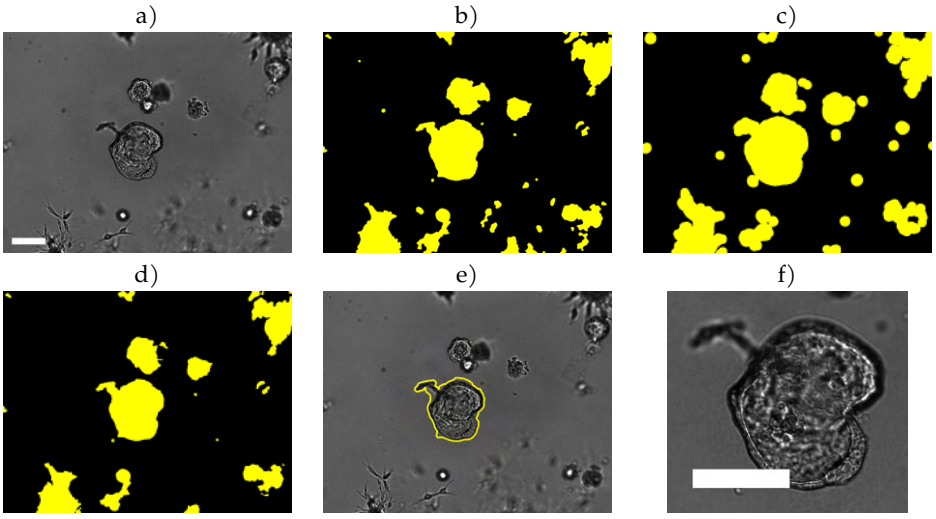
logically filtered. This image is then dilated with a CSE of diameter d_{D1} and intersected with the image from the top-hat pipeline to constrain segmentation to pixels lying close to regions with a considerable change in image intensities.

The resulting binary image is filtered morphologically, by component size, and by component location so that at most one connected component is present in the frame – a coarse segmentation S_c of the organoid of interest. Namely, components smaller than T_{AC} % of the largest component's size are removed and the component that minimises the following criterion is selected:

$$\left\| IDC - C_t^{i_t} \right\|_1 + \left\| C_{t-1} - C_t^{i_t} \right\|_1 \tag{4.1}$$

where IDC is the geometric center of the image domain, $C_t^{i_t}$ is the geometric center of i-th component in frame t, and C_{t-1} is the geometric

Table 4.1: Coarse segmentation: a) input image; b) binarised top-hat transform; c) binarised and dilated gradient magnitude; d) candidate components for the coarse segmentation; e) coarse segmentation S_C; and f) cropped input image. Yellow regions in b), c), and d) denote foreground pixels in binary images. Yellow outline in e) is used to overlay the binary segmentation over the input image. The white bars correspond to 100 μm.



center of the final component in frame $t-1$. For the first frame with $t=0$, $C_{t-1}=IDC$. The first term penalises components lying near the image domain borders while the second term penalises components whose location differs substantially from organoid's location in the previous frame. In case there are multiple components that minimise the criterion, the minimising component with the smallest label is chosen.

The bounding box of the chosen component is extended by $E_{BB}\%$ of the component's maximum diameter in all directions, which defines the subset of image domain where the subsequent steps are performed.

4.1.2 Focused Organoid Segmentation

As mentioned in the previous section, the organoid of interest usually exhibits much finer structure and higher variance of pixel intensities than its surrounding. The main body of the organoid is therefore determined by a measure of variance – or, more precisely, by standard deviation (Table 4.2).

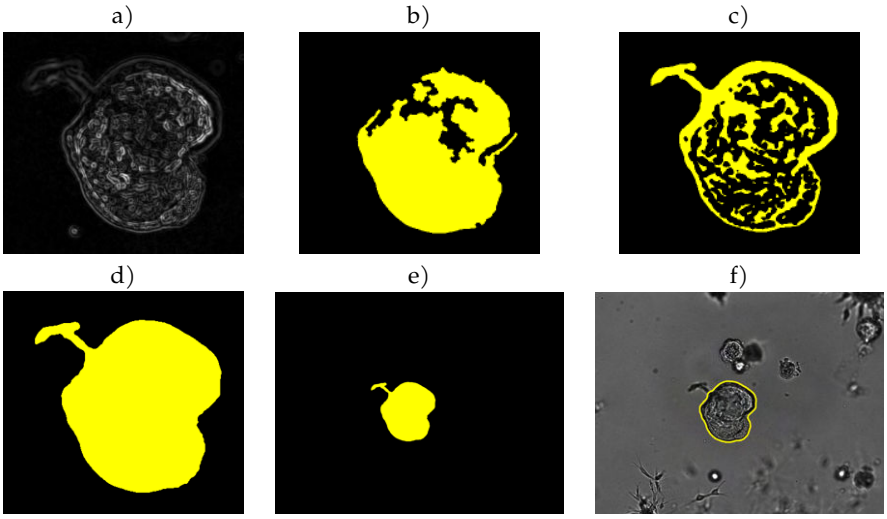
Standard deviation is computed in a $n \times n$ neighbourhood of each pixel in the bounding box. The result is thresholded by Otsu method with threshold T_1 and then filtered with morphological operators and by component size. The output is a binary image S_{HV} capturing all high-variance areas.

4.1.3 Segmentation of Dark Regions

Although the organoid of interest exhibits a distinct structure in most cases, it can become blurred due to motion or extending branches outside of the focal plane. Out-of-focus branches and organoids have smoother texture and are delineated by a dark blurred boundary region. These dark blurred regions are segmented in this step.

The cropped input image is smoothed with a Gaussian function, thresholded with the Otsu method, and inverted. At the same time, the image of standard deviation is thresholded with threshold $T_2 = r \cdot T_1$, filtered morphologically and by component size, and dilated with a CSE of diameter d_{D2}. The two binary images – segmentation of dark regions and dilated high-variance regions – are intersected. The result (S_{DB}) captures dark regions lying close to a focused region, which

Table 4.2: Segmentation refinement: a) standard deviation in the cropped input image; b) binarised and filtered standard deviation (S_{HV}); c) dark regions (S_{DB}); d) fine segmentation after morphological filtering; e) intersection of the coarse and fine segmentation; and f) final segmentation. Yellow regions in b), c), d), and e) denote foreground pixels in binary images. Yellow outline in f) is used to overlay the binary segmentation over the input image.



corresponds to segmentation of out-of-focus branches of the organoid of interest.

4.1.4 Integration of Local Segmentation

Finally, the images of high-variance areas (S_{HV}) and defocused parts of organoid (S_{DB}) are fused by union and morphologically filtered (Table 4.2). Then, the largest component is selected and intersected with the coarse segmentation S_c. Finally, an opening with a CSE of diameter d_O is computed to remove thin protrusions and to smooth the organoid boundary, and all components except for the largest one are removed.

4.2 Deep-Learning-Based Segmentation

The proposed DL-based pipeline for segmentation of 2D+t sequences of organoid images consists of the following steps (Figure 4.2):

- preprocessing of the input sequence;
- neural network prediction of pixel-level class labels;
- and postprocessing of the class prediction for binary output.

The individual steps are described in detail in the following sections and are illustrated in Table 4.3.

4.2.1 Preprocessing

Preprocessing of the input image data involves pixel value remapping to the interval $\langle 0, 1 \rangle$ as suggested in [51] to enhance the image data for

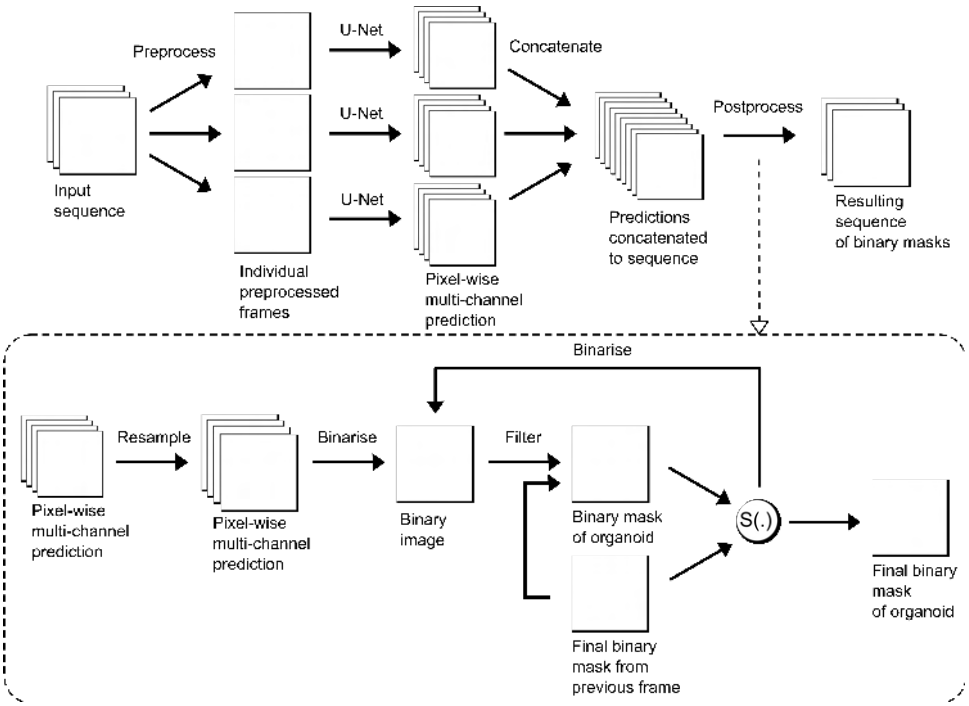


Figure 4.2: Deep-learning-based pipeline for 2D+t sequence segmentation.

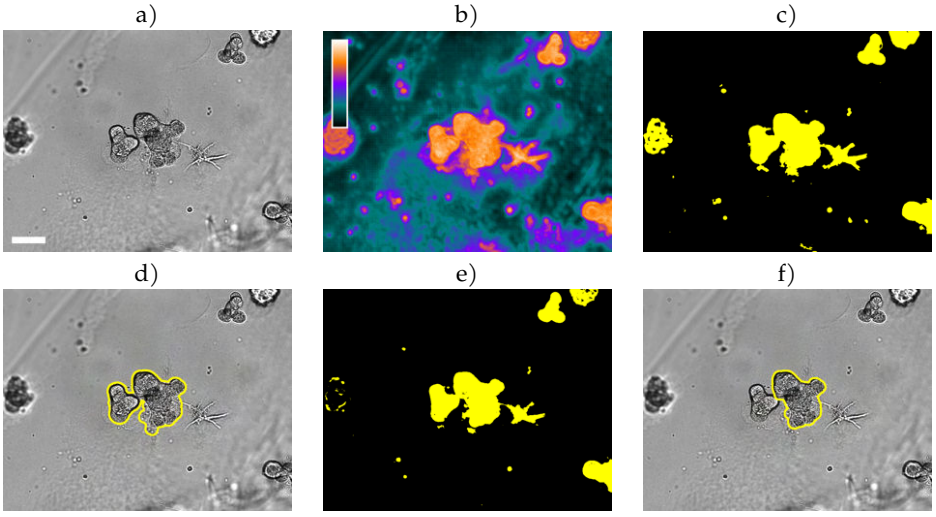
processing by a neural network. Pixel value normalisation is generally crucial for robustness and performance of neural networks [41, 80].

4.2.2 Neural Network

The preprocessed image data are provided as an input to a deep convolutional neural network with the U-Net architecture (Section 2.1.1). The network processes a single frame at a time and outputs a multi-channel image where each channel holds a pixel-wise prediction score for a single object class. Number of classes learnt depends on the training dataset. Training of the network is described in Section 5.2.2.

Since the network processes individual frames separately, the temporal dependencies between consecutive images in the sequence are ignored, which may result in inaccurate segmentation in some cases

Table 4.3: Segmentation using deep learning: a) input image; b) U-Net prediction displayed with a LUT and with scale in the top left corner; c) binarised prediction; d) final segmentation after the component filtering step; e) prediction binarised with threshold $T_{org} + \Delta T_{org}$; f) final segmentation after the segmentation correction step. Contrast of the input image was enhanced for visualisation purposes. The white bar corresponds to 100 μm.



(Chapter 6). To compensate for that, the postprocessing steps make use of the inter-frame dependencies.

The design of postprocessing is exceptionally important; it has been observed that although the convolutional networks generally perform well in classification tasks, raw outputs still need refinement to be useful in practical tasks [41, 11]. While the neural network learns the distinctive features on its own, the postprocessing step allows one to embed the *a priori* knowledge explicitly in the algorithm. Namely, the presence of a single object of interest in the whole sequence, organoid's minimum diameter and area, organoid's expected location, and differences between consecutive frames are considered in the following steps.

4.2.3 Prediction Binarisation

Firstly, the network predictions are resampled to match the resolution of the input image data. Although U-Net is designed to produce per-pixel scores, its output is smaller than the input image by a few pixels due to unpadded convolutions [51].

I have proposed several ways of prediction binarisation:

- ***Simple***: Only predictions for *organoid* class are used in the subsequent processing. The *organoid* score map is thresholded with a fixed threshold T_{org} that represents a decision threshold of the particular model.
- ***Multi***: Predictions for multiple classes are processed and combined to obtain a single binary image; binarisation may happen either before or after the combination of individual images. One of the possibilities is to compute argmax over all output channels [54] and extract pixels classified as *organoid*.
- ***Hybrid***: Prediction for *organoid* class is enhanced by explicit feature score extracted from the original image. This approach thus consists of two independent pipelines operating on the input image sequence: one is based on U-Net, the other resembles the baseline algorithm (Section 4.1). Both pipelines output a real-valued score map that is normalised, combined (e.g. by a weighted sum), and binarised.
- ***Prev***: U-Net prediction can be also enhanced by the prediction score or binary result of the previous frame since the change

between consecutive frames is usually very small. This approach results in worse performance on organoid boundary, but may prevent serious under- or over-segmentation in case of network uncertainty in a complicated case.

4.2.4 Component Filtering

The binarised image is processed with a sequence of morphological, size-based, and location-based filters to refine component shape and select component representing the organoid of interest.

- ***O-filtering***: Firstly, the image is processed with a morphological closing with a CSE of diameter d_C, the hole-filling operator, and a morphological opening with a CSE of diameter d_O. Secondly, area opening is performed to remove components smaller than T_A pixels. Finally, location-based filtering is used to select a single component that minimises the criterion in Equation (4.1).
- ***DE-filtering***: The morphological opening in the *O-filtering* approach is replaced with erosion with a CSE of diameter d_{DE}. After the area- and location-based filtering, morphological dilation with a CSE of diameter d_{DE} is applied. This approach tends to cut off protrusions thinner than d_{DE} pixels more effectively than the *O-filtering* approach (Section 6.2).

4.2.5 Segmentation Correction

The binarisation threshold T_{org} is chosen to reflect the network decision threshold. Although this approach generally works well (Section 5.2), under- and over-segmentation can occur in some cases. To compensate for the network uncertainty in classification, the following correction procedure can optionally be applied.

After establishing the result for frame $t > 0$, a similarity measure $S(\cdot)$ is computed for the results of frames t and $t-1$. If the change in the value of $S(\cdot)$ is greater than $T_S \cdot 100\%$, segmentation for frame t is recomputed with $T_{org} \pm \Delta T_{org}$. Note that only steps described in Sections 4.2.3 and 4.2.4 are repeated. The number of iterations of this correction procedure can be constrained to be at most R_{max} to control time efficiency.

This approach accounts for under- as well as over-segmentation. Alternatively, the correction mechanism may be applied in one of these cases only.

5 Experimental Results

An evaluation of the proposed algorithms is presented in this chapter. Evaluation measures are defined in Section 5.1. Training of neural networks for the DL-based algorithm and parameter optimisation for both DL-free and DL-based algorithms are described in Section 5.2. Finally, quantitative evaluation of selected algorithms is presented in Section 5.3.

5.1 Evaluation Measures

All algorithms are evaluated using the real image datasets described in Section 3.1. For algorithms with a neural network trained with the real image data, the evaluation on the *target dataset* is done using five-fold cross-validation [49]. Five different networks were trained with a different subset of annotated images from the *target dataset* and evaluated using the rest of annotated images (one sequence for each phenotype from the *target dataset* and all sequences from the *robustness validation dataset*). All five networks share the same postprocessing algorithm and hyper-parameter values. The overall performance of such an approach is defined as the average performance of these five networks.

Let $N = 0, 1, ..., n-1$ be a set of indices of all frames in a single sequence and let $j \in J \subseteq N$ be a set of indices of frames with the reference segmentation annotation. Let $A = A_0, A_1, ..., A_{n-1}$ be a sequence of automatically computed binary masks. Let R_j be a reference binary segmentation mask corresponding to $A_j, j \in J$. Let $D_i, i \in N$ be a binary detection marker corresponding to A_i. Let R be a sequence of all $R_j, j \in J$ and let D be a sequence of all $D_i, i \in N$. Each reference annotation R_j and each detection marker D_i always contains one connected component. The terms "connected component" and "object" are used interchangeably in this section.

The following measures are used to quantify the performance and accuracy of the proposed algorithms:

- Segmentation Accuracy (SEG) [29, 81, 82, 83] quantifies the overlap between reference annotation R_j and automatic segmen-

tation A_j using the Jaccard similarity index for pairs of matching objects $U_k \in R_j$ and $V_l \in A_j$:

$$JC(U_k, V_l) = \frac{|U_k \cap V_l|}{|U_k \cup V_l|}. \tag{5.1}$$

Objects $U_k \in R_j$ and $V_l \in A_j$ are considered matching if and only if

$$|U_k \cap V_l| > 0.5 \cdot |U_k|. \tag{5.2}$$

Thus, there can be at most one object $V_k \in A_j$ matching each object $U_l \in R_j$. False positive objects are not penalised.

Since the reference annotations for the input image data always contain exactly one object $U \in R_j$, JC is measured for at most one object $V \in A_j$ for each annotated frame A_j:

$$SEG(R_j, A_j) = JC(U, V) \tag{5.3}$$

where U, V is the pair of matching objects for frame j. If no object V matching U can be found, $SEG(R_j, A_j)$ is set to zero.

The SEG score of an image sequence is computed by averaging the SEG scores for all frames $j \in J$:

$$SEG(R, A) = \frac{1}{|J|} \sum_{j \in J} SEG(R_j, A_j). \tag{5.4}$$

The result is a real number in the interval $\langle 0, 1 \rangle$, with a higher number meaning better segmentation accuracy.

- Detection Accuracy (DET) [81, 82] penalises differences between the detection markers D and the automatic segmentation A in a graph-based representation:

$$DET(D, A) = 1 - \frac{\min(\text{AOGM-D}(G_D, G_A), \text{AOGM-D}(G_D, G_E))}{\text{AOGM-D}(G_D, G_E)} \tag{5.5}$$

where AOGM-D(F, G) is the Acyclic Oriented Graph Measure for detection [84] computed for graphs F and G, G_D is the graph constructed from D, G_A is the graph constructed from A, and

G_E is an empty graph. The default penalty configuration of (10.0, 5.0, 1.0) was used to penalise missing, nonsplit, and spurious objects.

The result is a real number in the interval $\langle 0, 1 \rangle$, with a higher number meaning better detection performance. Unlike the SEG measure, the value of DET decreases with an increasing number of false positive objects.

- Hausdorff distance (HD) [85, 83, 86] measures shape dissimilarity of two sets S_1, S_2 as the maximum distance between each point pair of S_1, S_2:

$$HD(S_1, S_2) = \max\{h(S_1, S_2), h(S_2, S_1)\} \tag{5.6}$$

$$h(X, Y) = \max_{x \in X} \min_{y \in Y} d(x, y) \tag{5.7}$$

where $d(x, y)$ is the distance between the points x and y.

The result is a non-negative number. The lower the value of $HD(S_1, S_2)$, the more similar are the sets S_1 and S_2.

There are many variants of the Hausdorff distance. This work uses the percentile Hausdorff distance (HD^k) [86, 87, 83] that is less sensitive to outliers than HD:

$$HD^k(S_1, S_2) = \max\{h^k(S_1, S_2), h^k(S_2, S_1)\} \tag{5.8}$$

$$h^k(X, Y) = \mathrm{P}^K_{x \in X} \min_{y \in Y} d(x, y) \tag{5.9}$$

where $\mathrm{P}^k_{x \in X}$ denotes the k-th percentile of X.

In this thesis, I use $k = 0.95$ and Euclidean distance d. All foreground pixels in a frame are considered to be a single set of pixels. Thus, false positive objects are hugely penalised.

Instead of the whole organoid masks R_j and A_j, only mask boundaries R_j^B, A_j^B are compared [88]. Boundary of a mask is defined as a set of mask pixels where each pixel has at least one background pixel in its 8-neighbourhood.

If A_j is an empty image (contains only background pixels), I set $HD^k(R_j^B, A_j^B) = 0$. To compensate for that, a number of empty

frames in set $A_j, j \in J$ output by each algorithm is computed and presented separately. False negative objects thus are not penalised directly by this measure.

The percentile Hausdorff distance of two image sequences is computed as

$$HD^k(R, A) = \max_{j \in J} HD^k(R_j^B, A_j^B). \tag{5.10}$$

These three evaluation measures were chosen to reflect different quality requirements for the automatic segmentation results. Firstly, correct detection of organoid in each frame is measured by DET. Secondly, segmentation accuracy in terms of area overlap is measured by SEG. And finally, shape similarity and accurate boundary delineation is measured by HD^{95}.

In addition, the computation time was measured. All computations were done on a single workstation with AMD Opteron 6348, 120 GB of RAM, Ubuntu 20.4 LTS, and equipped with a single NVIDIA Quadro P6000 graphics card with 24 GB of RAM.

5.2 Experiments

Experiments conducted for optimisation of algorithm performance are described in this section. Namely, the parameter optimisation for the baseline algorithm, the training of neural networks, and the parameter optimisation for the postprocessing routines are described.

5.2.1 Baseline Algorithm

In this subsection, the process of determining values of individual parameters of the baseline algorithm (Section 4.1) is summarised.

Several parameters, such as SE diameters for morphological operators and minimum component size for size-based filtering, were set according to their biological significance (Chapter 6).

Other parameter values were also decided by their meaning. The parameter σ of the Gaussian blur operator in the dark region segmentation was designed to slightly refine the raw image data. Computation of threshold $T_2 = r \cdot T_1$ in the same part of the algorithm allows one to

Table 5.1: Parameter values selected for the baseline algorithm based on a parameter study.

Parameter	d_{TH}	d_{D1}	T_{AC}	E_{BB}	n	σ	r	d_{D2}	d_O
Value	40	30	75	10	3	2	0.75	40	35

include a different amount of pixels into the segmentation of the high-variance regions and the high-variance constraint for blurred organoid regions. Pixel neighbourhood size n was set to balance computation speed and segmentation accuracy. Bounding box extension ratio E_{BB} accounts for inaccurate coarse segmentation near the organoid boundary.

The parameters d_{TH}, d_{D1}, d_{D2}, and T_{AC} were optimised in a grid-search manner. The parameter study was evaluated with respect to the SEG measure on the *target dataset* and its evaluation can be found in the MUNI Thesis Archive.

The final parameter values are summarised in Table 5.1, not listing the diameters of morphological operators. These are available in code in the MUNI Thesis Archive.

5.2.2 Training of Neural Networks

The training process of U-Net models for the DL-based algorithms (Section 4.2) is described in this subsection, including the training protocol followed, hyper-parameter values chosen, and training datasets used.

U-Net is trained using the stochastic gradient descent [51]. The loss function E is a weighted cross-entropy loss computed over pixel-wise soft-max of the output image:

$$E = \sum_{x \in \Omega} w(x) \log \frac{\exp(a_{l(x)}(x))}{\sum_{k=1}^{K} \exp(a_k(x))} \tag{5.11}$$

where $\Omega \subset \mathbb{Z} \times \mathbb{Z}$ is the image domain, $w(x)$ is a weight of pixel x, $l(x)$ is the expected label of pixel x, $a_k(x)$ is the activation of the k-th channel at x, and K is the number of classes. The weights $w(x)$ are computed in advance for each reference segmentation; they compensate for the pixel class imbalance in the training dataset and amplify the importance of certain image regions.

The training hyper-parameters and training datasets of the individual models are summarised in Table 5.2. All models were trained using the ImageJ plugin by Falk et al. [54] running on GPU. Training of the *Synth-5* model required approximately 30 minutes and 10 GB of memory. Training of a single *Real* model took approximately 15 minutes and consumed approximately 4 GB of memory.

The models were trained to classify pixels into two or four classes. For datasets with two-class reference annotations, only the first case applies. The *G2 dataset* was used to train networks for two as well as four classes. In the two-class case, the *organoid* class served as a foreground, while the *background*, *DBS*, and *BBS* classes were merged into the background.

The images in all training datasets were sampled regularly from the available sequences to balance training dataset size, data variability, phenotype representation, and time required for training. All training images were augmented to create even more diverse dataset [51, 54,

Table 5.2: Training parameters for individual U-Net models: number and type of training images; number of validation images; number of classes; training mode – transfer learning (TL) or training from scratch (TFS); tiling (in pixels); and number of epochs.

Model	Training images	Valid. images	Classes	Mode	Tiling	Epochs
Real	120 real (*TD*): 40 CY, 40 NB, 40 MB	3	2	TL	508 × 508	1200
Synth-1	479 synthetic (*G1*): 257 CY, 222 NB	10	2	TL	508 × 508	300
Synth-2	335 synthetic (*G1*): 335 NB	5	2	TL	508 × 508	300
Synth-3	335 synthetic (*G1*): 335 NB	5	2	TFS	1036 × 524	1200
Synth-4	370 synthetic (*G2*): 134 CY, 121 NB, 115 MB	8	2	TL	700 × 1036	300
Synth-5	399 synthetic (*G2*): 135 CY, 133 NB, 131 MB	8	4	TL	700 × 1036	300

89]. In general, diversity of the training data as well as balanced class representation are crucial for good performance of the network [17, 54, 5]. Although it is claimed [51, 54] that very few images are necessary for successful transfer learning with U-Net, it has been observed that network performance still tends to improve with more training data [5].

The validation images were used to estimate network performance during the training. In order to create validation datasets for individual networks, approximately 2% of training images were randomly chosen and left out from the training dataset (Table 5.2).

The *Real* model was trained and evaluated in a five-fold cross-validation manner. Five *Real-X* models were trained with the same hyper-parameters, differing only in the training data: for each model, images from one sequence per phenotype were excluded from the training data to form an evaluation dataset. All other models were trained with synthetic image data, and thus evaluated using all annotated real images.

The models were trained in two manners: from scratch or by transfer learning. When training from scratch, model weights were initialised randomly from a Gaussian distribution with the standard deviation of $\sqrt{2/N}$ where N is a number of nodes incoming to one neuron [51]. The base network for transfer learning was a model trained with 11 datasets of microscopy cell images by Ronneberger et al. [90], including brightfield image data.

Tiling of the input images was used to balance resource requirements and computation time during the training phase. The tiles overlapped in a few pixels and each tile was padded with mirroring if necessary. The tiles were processed separately by the network.

All models used the learning rate of 0.0001. Other hyper-parameters not included in Table 5.2 were fixed at their default values provided by the software [54].

5.2.3 Postprocessing Routines

Five postprocessing routines proposed for the DL-based approach are presented in this subsection. While their general outline is described in Chapter 4, particular methods used in the individual postprocessing routines are summarised in Table 5.3.

The parameter values in the postprocessing routines were optimised in combination with the network models described in Section 5.2.2. Similarly to Section 5.2.1, the parameter studies were evaluated with respect to the SEG measure on the *target dataset*.

Namely, the threshold T_{org} for prediction binarisation and the parameters T_S and ΔT_{org} for the segmentation correction were optimised.

Firstly, SE diameters d_C, d_O, and d_{DE} for morphological operators and area-based filtering threshold T_A were set to reflect the biological properties of the input image data (Chapter 6).

Table 5.3: Implementation of individual postprocessing routine. If the segmentation correction is applied, the detected segmentation errors are indicated (under-segmentation US, over-segmentation OS).

Postprocessing routine	Binarisation	Filtering	Segmentation correction
PP-1	*Simple*	*DE-filtering*	No
PP-2	*Simple*	*O-filtering*	Yes (US)
PP-3	*Simple*	*DE-filtering*	Yes (US & OS)
PP-4	*Simple + Hybrid + Prev*	*DE-filtering*	No
PP-5	*Multi*	*DE-filtering*	No

Table 5.4: Values of the decision threshold T_{org} optimised for individual network models.

U-Net model	*Real*	*Synth-1*	*Synth-2*	*Synth-3*	*Synth-4*	*Synth-5*
Threshold	0	-0.5	0	0	1.7	2.4

Table 5.5: Parameter values chosen for the individual postprocessing routines based on parameter studies. Missing parameters are denoted by a hyphen (-).

Postprocessing algorithm	d_C	d_O	d_{DE}	T_A	$S(\cdot)$	T_S	ΔT_{org}	R_{max}
PP-1	4	-	25	300	-	-	-	-
PP-2	4	35	-	300	area	0.1	0.15	2
PP-3 (A)	4	-	25	300	area	0.1	0.1	2
PP-3 (F)	4	-	25	300	Feret	0.1	0.2	2
PP-4	4	-	25	300	-	-	-	-
PP-5	4	-	25	300	-	-	-	-

The decision threshold T_{org} was then optimised for each U-Net model combined with the *PP-1* routine with $d_C = 4$ and $d_{DE} = 25$. Consequently, parameter studies for d_C and d_{DE} were conducted to measure the impact of these parameters to algorithm performance. The experiments showed that changing the value of d_C does not result in a considerable increase in the SEG score whereas increase in the value

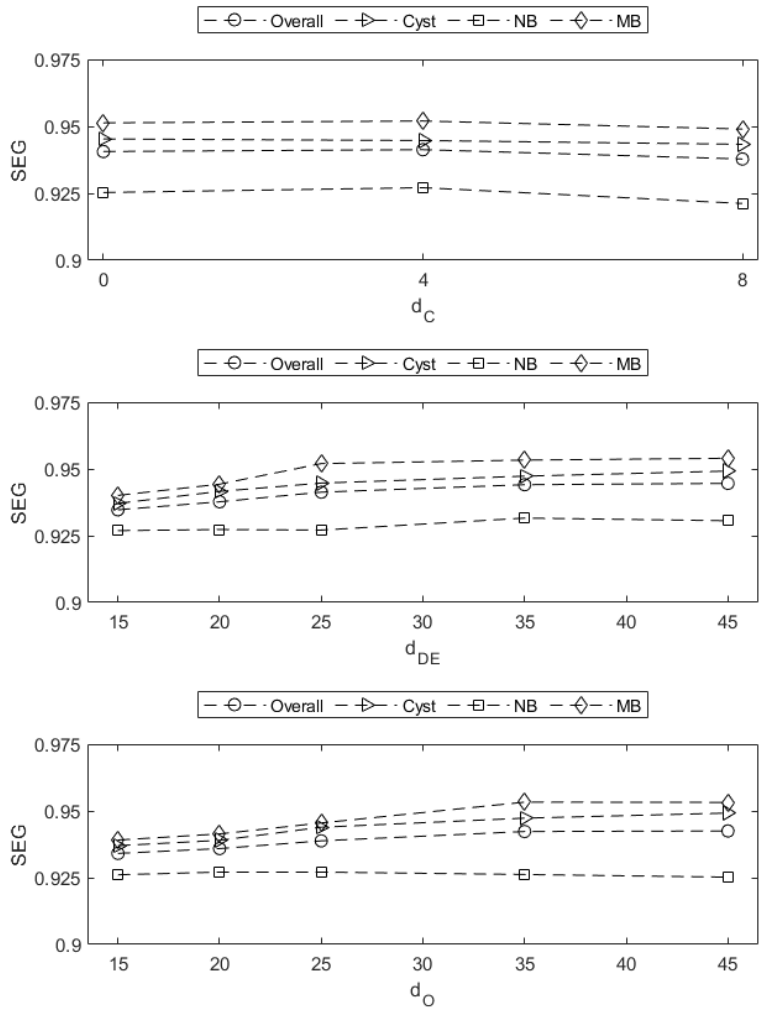


Figure 5.1: Evaluation of the impact of d_C, d_{DE}, and d_O in the *Synth-5 + PP-1* algorithm.

of d_{DE} or d_O slightly increases the SEG score (Figure 5.1). However, I have decided not to alter the d_{DE} and d_O values due to reasons explained in detail in Chapter 6.

Finally, the parameters T_S and ΔT_{org} for the *PP-3* routine were optimised in combination with three different U-Net models using a grid-search strategy, with R_{max} fixed at 2 to balance algorithm performance and computation speed. Two shape descriptors $S(\cdot)$ – area and Feret diameters – were considered and each was assigned its own optimal parameter values. The number of frames that would require more than $R_{max} = 2$ iterations is relatively low – less than 2% of all frames for the established parameter values.

Optimisations of the *PP-2* routine were performed with the *Synth-4* model and are described in more detail in [9].

The *PP-4* routine is a combination of a DL-based approach and the baseline algorithm and thus shares parameters with both approaches. The parameter values for *PP-4* were not optimised; instead, the optimal values established for its individual pipelines were used and the optimisation concentrated solely on the weights of the individual pipelines of *PP-4*. The optimal values of weights for the *PP-4* routine as well as its modifications by omitting a certain pipeline can be found in Table 5.6 and in the supplementary material in the MUNI Thesis Archive.

Finally, the *PP-5* routine implements binarisation with the argmax operation. No parameter optimisation was done for this postprocessing routine; parameters d_C, d_{DE}, and T_A were assigned the values estimated for *PP-1*.

Table 5.6: Values of the weights w_1, w_2, and w_3 for the individual pipelines of the *PP-4* routine (network prediction, baseline segmentation, and previous frame segmentation) and the fused score map threshold w_4 optimised in a parameter study. Omitted pipelines are denoted by a hyphen (-).

Algorithm	w_1	w_2	w_3	w_4
Simple + Hybrid + Prev	1	0.25	0.25	1.5
Simple + Hybrid	1	0.25	-	1.25
Simple + Prev	1	-	0.25	1.25
Hybrid + Prev	-	1	0.5	0.75

The established parameter values for individual postprocessing routines are shown in Tables 5.4 and 5.5. Evaluation of the performed parameter studies can be found in Figure 5.1 for chosen parameters and in the MUNI Thesis Archive for all considered parameter values.

5.3 Results

The quantitative evaluation of both DL-based and DL-free algorithms proposed in this thesis is summarised in this section.

The algorithms were run with the parameter values as chosen in Section 5.2. While evaluation of the experiments was done using the

Table 5.7: Quantitative evaluation of chosen algorithms and annotator performance on the *target dataset* (mean or maximum ± standard deviation for each measure). Best score achieved by an automatic method is highlighted in bold for each measure. Label $^{(x)}$ in the HD^{95} column denotes the number x of empty evaluation frames output by the algorithm (Chapter 5.1).

Algorithm	*DET*	*SEG*	*HD*^{95}
Human annotators	*1.000 ± 0.000*	*0.970 ± 0.023*	*88.65 ± 16.30*
Baseline	0.972 ± 0.071	0.932 ± 0.042	213.18 ± 61.69
Real + PP-1	0.991 ± 0.32	0.879 ± 0.072	516.13 ± 135.10
Synth-1 + PP-1	0.968 ± 0.111	0.910 ± 0.062	429.96 ± 118.72$^{(1)}$
Synth-2 + PP-1	**1.000 ± 0.000**	0.925 ± 0.048	282.14 ± 71.16
Synth-3 + PP-1	0.979 ± 0.076	0.880 ± 0.083	518.05 ± 124.46
Synth-4 + PP-1	0.979 ± 0.074	0.904 ± 0.061	147.10 ± 41.88$^{(1)}$
Synth-5 + PP-1	**1.000 ± 0.000**	**0.941 ± 0.028**	140.00 ± 40.84
Synth-4 + PP-2	**1.000 ± 0.000**	0.910 ± 0.059	147.34 ± 41.69
Synth-2 + PP-3 (A)	**1.000 ± 0.000**	0.928 ± 0.043	139.03 ± 39.81
Synth-4 + PP-3 (A)	0.997 ± 0.010	0.907 ± 0.059	147.10 ± 40.88
Synth-5 + PP-3 (A)	**1.000 ± 0.000**	0.940 ± 0.031	139.45 ± 40.76
Synth-2 + PP-3 (F)	**1.000 ± 0.000**	0.930 ± 0.038	139.06 ± 39.02
Synth-4 + PP-3 (F)	0.997 ± 0.010	0.908 ± 0.059	147.10 ± 40.82
Synth-5 + PP-3 (F)	**1.000 ± 0.000**	0.940 ± 0.032	**138.16 ± 40.51**
Synth-5 + PP-4	0.998 ± 0.007	0.935 ± 0.064	139.06 ± 41.29
Synth-5 + PP-5	0.948 ± 0.147	0.833 ± 0.099	458.46 ± 110.90
Synth-5 + PP-0	0.922 ± 0.187	0.709 ± 0.171	951.76 ± 215.65

SEG measure on images from the *target dataset* only, evaluation with two additional measures – the *DET* measure and the 95-percentile Hausdorff distance HD^{95} – and evaluation over the *robustness validation dataset* are presented in this section. The comparison also includes a *Synth-5 + PP-0* algorithm, where the network multi-channel score is

Table 5.8: Quantitative evaluation of chosen algorithms and annotator performance on the *robustness validation dataset* (mean or maximum $\pm$ standard deviation for each measure). Best score achieved by an automatic method is highlighted in bold for each measure. Label $^{(x)}$ in the HD^{95} column denotes the number x of empty evaluation frames output by the algorithm (Chapter 5.1).

Algorithm	*DET*	*SEG*	HD^{95}
Human annotators	*1.000 ± 0.000*	*0.968 ± 0.014*	*98.27 ± 19.69*
Baseline	**1.000 ± 0.000**	**0.936 ± 0.028**	306.31 ± 91.56
Real + PP-1	0.988 ± 0.036	0.889 ± 0.029	324.47 ± 74.08
Synth-1 + PP-1	0.996 ± 0.010	0.917 ± 0.045	189.40 ± 68.25
Synth-2 + PP-1	0.973 ± 0.082	0.925 ± 0.034	259.49 ± 78.75
Synth-3 + PP-1	0.980 ± 0.061	0.892 ± 0.063	332.23 ± 90.17
Synth-4 + PP-1	**1.000 ± 0.000**	0.913 ± 0.028	293.69 ± 87.45
Synth-5 + PP-1	0.968 ± 0.083	0.816 ± 0.158	267.66 ± 83.14
Synth-4 + PP-2	**1.000 ± 0.000**	0.915 ± 0.023	180.56 ± 60.82
Synth-2 + PP-3 (A)	0.971 ± 0.088	0.935 ± 0.016	**74.30 ± 20.68**
Synth-4 + PP-3 (A)	**1.000 ± 0.000**	0.917 ± 0.020	175.92 ± 59.85
Synth-5 + PP-3 (A)	0.972 ± 0.080	0.826 ± 0.144	263.36 ± 79.62
Synth-2 + PP-3 (F)	0.974 ± 0.078	0.929 ± 0.028	303.13 ± 82.59
Synth-4 + PP-3 (F)	0.999 ± 0.002	0.910 ± 0.021	175.92 ± 59.85
Synth-5 + PP-3 (F)	0.964 ± 0.087	0.832 ± 0.148	262.36 ± 78.96
Synth-5 + PP-4	0.998 ± 0.006	0.913 ± 0.034	294.63 ± 86.11
Synth-5 + PP-5	0.969 ± 0.081	0.861 ± 0.119	561.08 ± 162.15
Synth-5 + PP-0	0.997 ± 0.006	0.636 ± 0.159	762.64 ± 189.53

Table 5.9: Average computation time of individual algorithms or their parts per frame in seconds.

Algorithm	Baseline	Preproc. & U-Net	*PP-1*	*PP-2*	*PP-3 (F)*	*PP-4*	*PP-5*	*PP-0*
Time	2.35	1.10	1.76	1.85	1.94	2.79	1.86	1.68

Table 5.10: Qualitative comparison of the analysed algorithms in a cropped region around a NB organoid in frame $t = 151$. Yellow mask outline is overlaid over the input frame. Contrast of the input image was enhanced for visualisation purposes. The white bar corresponds to 100 µm.

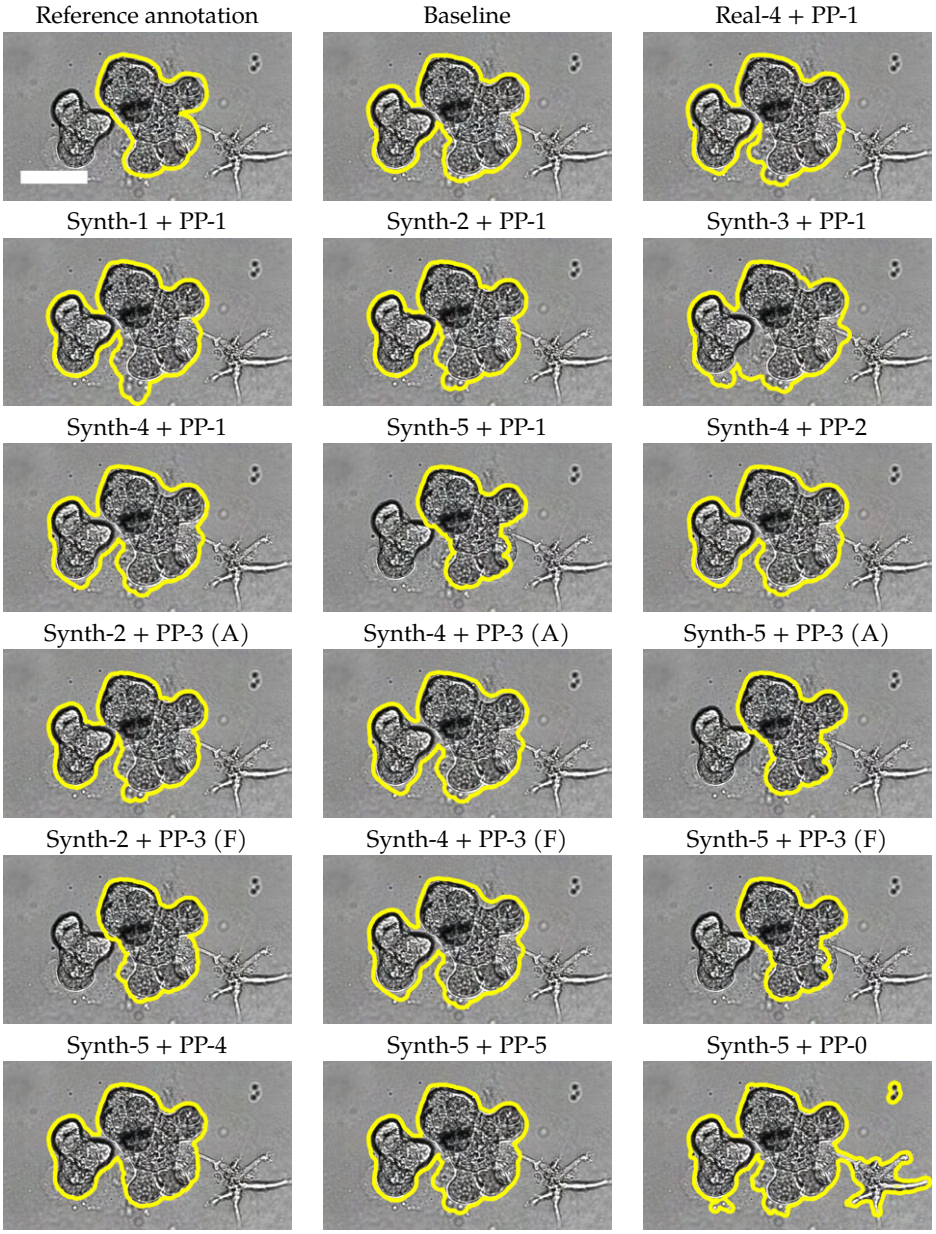
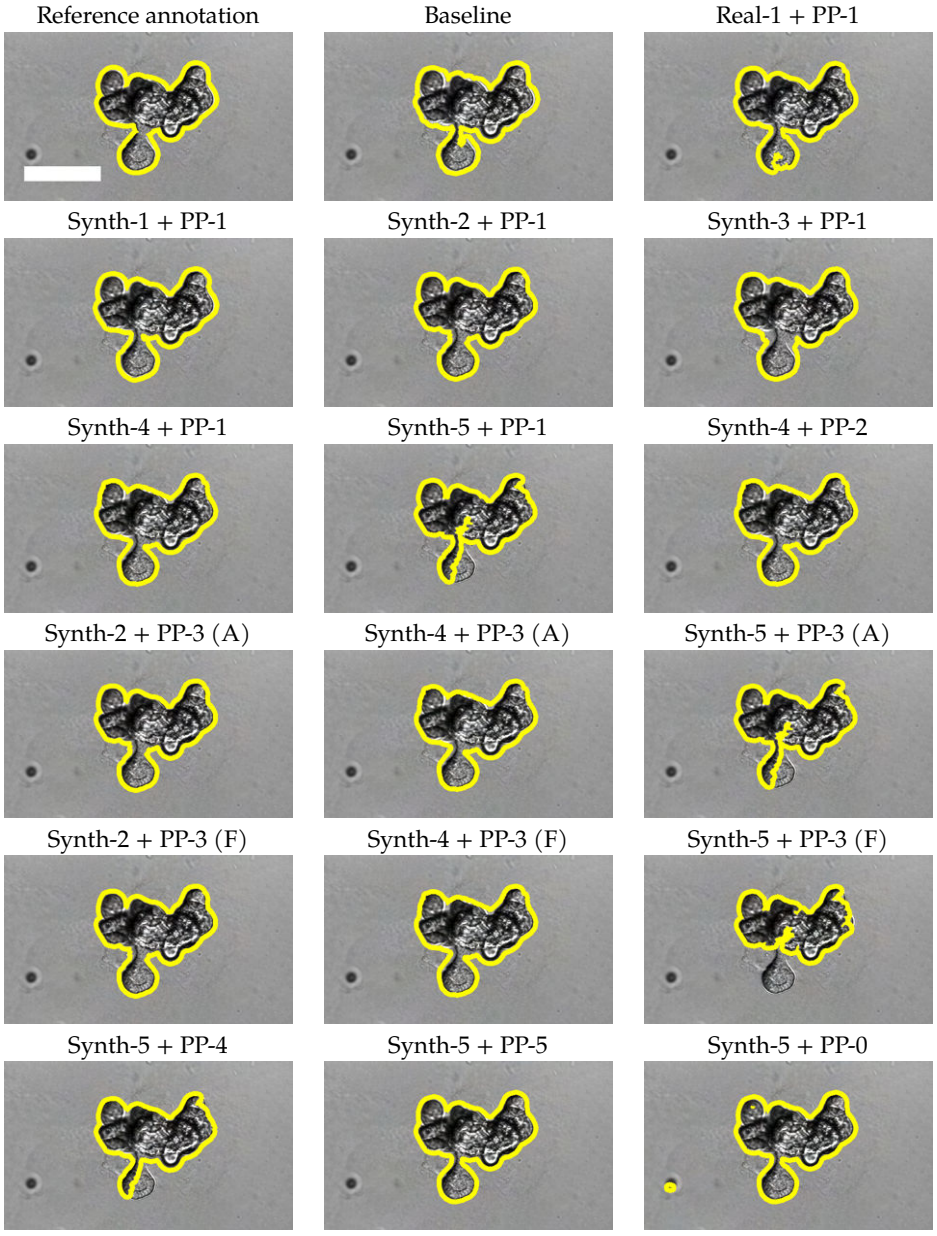


Table 5.11: Qualitative comparison of the analysed algorithms in a cropped region around a HB organoid in frame $t = 121$. Yellow mask outline is overlaid over the input frame. Contrast of the input image was enhanced for visualisation purposes. The white bar corresponds to 100 µm.



processed with the argmax operation without any further refinement. This algorithm is included to demonstrate the necessity of a suitable postprocessing routine (Section 6.3).

The evaluation also includes a measurement of a *human performance* that accounts for the inter-annotator variability [49]. The human performance was computed by considering individual human annotations as an input to the evaluation measures together with the reference annotations. Quantitative results are summarised in Tables 5.7 and 5.8.

Additionally, the computation time was measured on the *robustness validation dataset*. For the DL-based algorithms, the time of processing using U-Net and the time of the DL-free postprocessing routine were measured separately. All sequences were segmented with the same U-Net segmentation hyper-parameters: no input image tiling, no averaging, in the GPU mode. Segmentation of a single sequence of 161 frames consumed approximately 3.9 GB of memory. The measured values are listed in Table 5.9.

As mentioned in Section 5.2.2, the *Real* model is in fact a common name for five different models that were trained with the same hyper-parameters and different training data. Each model was evaluated with a subset of the *target dataset* that was excluded from the training process and the results for all five models were averaged. The evaluation dataset thus contains one CY sequence, one NB sequence, one MB

Table 5.12: Evolution of organoid size in pixels. Mean and standard deviation manually estimated from the real image data in time $t = 0$, $t = 55$, $t = 110$, and in the last frame of a sequence are presented.

Time	CY	NB	MB	HB	LB
0	4750 ± 2528	3520 ± 1775	3940 ± 1801	5200 ± 816	3320 ± 893
55	8000 ± 2632	15460 ± 7291	22540 ± 12130	11780 ± 4544	9660 ± 3054
110	13080 ± 15787	33480 ± 10630	53280 ± 23745	12400 ± 2998	24180 ± 5130
Last	12680 ± 15536	33740 ± 10838	70160 ± 26942	12340 ± 2478	41140 ± 14085

sequence, and all HB and LB sequences for each *Real-X* model; only the HB and LB sequences are shared among the five models.

For the evaluation of the *PP-2* and *PP-3* routines, the number of frames that require more than two iterations of the segmentation correction procedure is relatively low for the established parameter value configurations for both considered shape descriptors: from 0.08% to 2.52% for the *target dataset* and from 0.06% to 6.19% for the *robustness validation dataset*. All measurements for both considered shape descriptors can be found in the MUNI Thesis Archive.

The HD^{95} measure can be interpreted as a 95-percentile maximum distance between two mask contours measured in pixels. To provide a better understanding of the measured HD^{95} values, approximate organoid sizes in four time points in the sequence are presented in Table 5.12.

In addition to the quantitative evaluation, examples of the segmentation results achieved using different algorithms are presented in Tables 5.10 and 5.11. For the *Real* model, an output of a single model from the *Real* family is included; while each sequence from the *target dataset* was segmented by only one *Real-X* model, sequences in the *robustness validation dataset* were segmented with all five *Real-X* models. Therefore, output of a single randomly chosen *Real-X* model is presented in Table 5.11.

Remarks on algorithm performance and comparison of the proposed approaches can be found in Chapter 6.

6 Discussion

In this chapter, I discuss the methodology, experiments, and evaluation results presented in Chapters 4 and 5. I also propose alternative approaches to analysing the input image data. Assumptions used to build the deep-learning-free routines are discussed in Section 6.1. The parameters of the deep-learning-free routines are analysed in Section 6.2. Algorithm comparison and a discussion of the achieved results can be found in Section 6.3. Finally, the alternative approaches are discussed in Section 6.4.

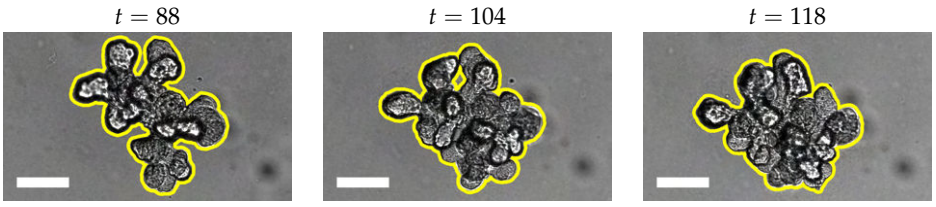
6.1 Assumptions

A few comments on the assumptions about the input image data are presented in this section. The assumptions were used to construct both the baseline algorithm and the postprocessing routines and were listed in Sections 4.1 and 4.2.2.

The assumption of tracking a single organoid of interest is one of the properties of the input sequences (Section 3.1). The presence of other organoids and epithelial structures introduces challenges for correct detection and tracking, but does not change the aim of the analysis.

The assumption of the organoid being a single object without holes might seem trivial, but need not hold in the image data. Although the organoid does not have any holes, its branches can appear to touch or

Table 6.1: Organoid with a hole. Segmentation mask is outlined in yellow. All images are cropped to show the necessary context. Contrast of the input images was enhanced for visualisation purposes. The white bars correspond to 100 μm.



overlap in the images and thus create a hole in the expected segmentation mask (Table 6.1). Nevertheless, this assumption is embedded in the algorithms because such situations are rare and its usage improves the quality of segmentation.

The expected organoid location involves both the assumption of organoid centered in the field of view and the assumption of organoid's slow motion. Both are encoded in location-based filtering in Equation (4.1): the first term penalises components that are not centered in the view, and the second term penalises abrupt changes in organoid location. Note that both assumptions about organoid location can be broken; the first by organoid motion over the course of time, the other due to microscope settings adjustment. Nevertheless, the experiments did not show any decrease in algorithm performance in these cases. An example of a correct detection of organoid of interest lying farther from the center of the image domain than another organoid can be seen in image i) in Table 6.7.

The minimum diameter and minimum area of organoid is another knowledge prior embedded in both the baseline algorithm and the postprocessing routines. Both minimum diameter and minimum area have different values for individual organoid phenotypes. Moreover, the expected organoid area changes considerably in the course of the experiment. Both knowledge priors are further discussed in Section 6.2.

The texture characteristics of the organoid of interest, other structures in the field of view, and the background are used in the baseline algorithm and the hybrid postprocessing routine (*PP-4*). They are embedded in all of their steps: the coarse top-hat segmentation (Section 4.1.1), both refinement pipelines (Sections 4.1.2 and 4.1.3), and the high-variance constraints for coarse segmentation and dark regions (Sections 4.1.1 and 4.1.3).

6.2 Parameters

The parameters d_O and d_{DE} are related to the expected minimum diameter of organoids and their branches. They should be chosen as high as possible to effectively exclude small or thin structures from further processing, but should not be higher than the expected branch

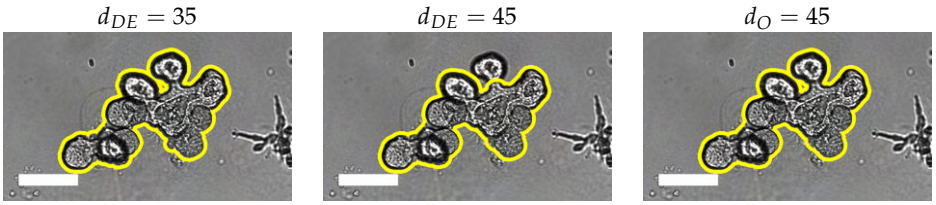
diameter. At the same time, these parameters also control boundary smoothness – the higher the values of these parameters, the higher the SEG score (Figure 5.1). The values of d_O and d_{DE} should therefore be chosen to preserve organoid branches and whole organoids in the output segmentation and at the same time to be as high as possible under this constraint. In my experiments, I have manually estimated the minimum branch or organoid diameter from the input image data; the values vary considerably for different phenotypes (Table 6.2).

One can notice that I have decided to use the value of $d_O = 35$ for the CY, NB, and MB organoids in my experiments although this value is higher than the minimum branch diameter estimate for the NB organoids. The NB organoids tend to form relatively short branches that grow close to each other and often overlap. As a result, the individual branches are not separated by background pixels and form a mass considerably thicker than their minimum diameter. The branch shape and the behaviour of the opening operator also contributed to this decision; even if a branch is separated from other branches, its

Table 6.2: Minimum branch or organoid diameter (in pixels) estimated from the real image data and values of d_{DE} and d_O used in algorithms.

Phenotype	CY	NB	MB	HB	LB
Min. branch diameter estimate	35	25	35	6	10
Value of d_{DE} used	25	25	25	5	5
Value of d_O used	35	35	35	5	5

Table 6.3: Influence of the choice and value of parameter d_{DE} and d_O on the segmentation. Output of the same algorithm for a single input frame for $d_{DE} = 35$, $d_{DE} = 45$, and $d_O = 45$. Segmentation mask is outlined in yellow. Contrast of the input image was enhanced for visualisation purposes. All images are cropped to show the necessary context. The white bars correspond to 100 μm.

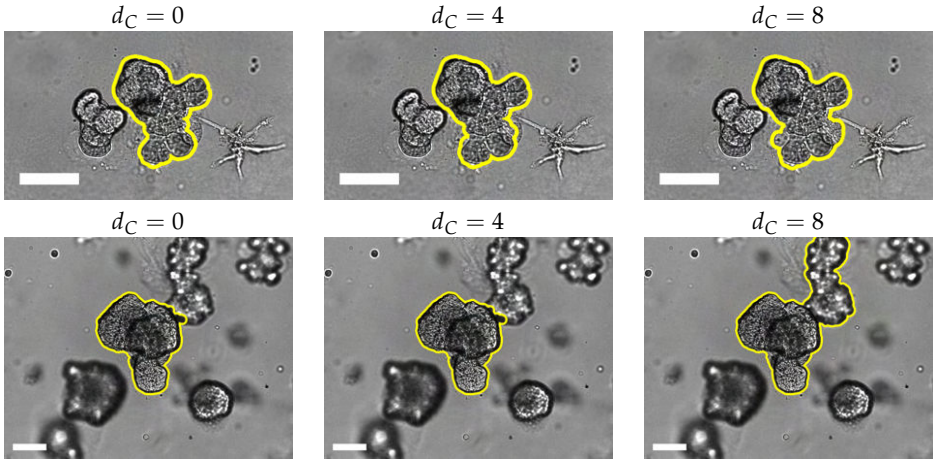


thinnest part is too short to be discontinued by opening with a CSE of diameter $d_O = 35$ (Table 6.3).

The parameter d_C controls the effect of morphological closing – a preprocessing step for the hole-filling operator. I have chosen the values of this parameter to reflect the need of each morphological filtering step: the values are relatively low in most cases, but the value of $d_C = 10$ is used for filtering the coarse segmentation after top-hat transform in the baseline algorithm, being suitable for this kind of image data. Evaluation of several values of d_C for the *Synth-5 + PP-1* algorithm shows that a slight change in the value of d_C does not substantially affect the overall algorithm performance (Figure 5.1). In fact, the chosen value of $d_C = 4$ has a slightly lower overall SEG score than $d_C = 0$ (no closing applied), but exhibits better behaviour in qualitative comparison (Table 6.4).

The size-based filtering parameters were also not optimised in most cases because they define the expected minimum size of organoids. The size threshold T_A in the postprocessing routines is an example

Table 6.4: Influence of the value of parameter d_C on the segmentation. Output of the same algorithm for two different input frames for $d_C = 0$, $d_C = 4$, and $d_C = 8$. Segmentation mask is outlined in yellow. Images in the first row are cropped to show the necessary context. Contrast of the input images was enhanced for visualisation purposes. The white bars correspond to 100 µm.



of such a case; together with d_O and d_{DE}, these three parameters constrain the expected organoid shape in the DL-based algorithms: T_A by minimum area, d_O and d_{DE} by minimum diameter. This filters out small objects as well as elongated epithelial structures. Similar intuition holds for the size-based filtering constraints in the baseline algorithm. Their values were chosen with enough room for possible mistake in the estimate and variations in the image data, yet they still form an effective component size filter. The only size-based constraint that was optimised in parameter studies was the T_{CA} parameter in the baseline algorithm; the evaluation included in the MUNI Thesis Archive showed that the considered values had minor effect on the SEG score. In case that *DE-filtering* (Section 4.2.4) is used, the value of the size-based filtering parameter T_A must be chosen with respect to the value of d_{DE} since the size filtering is applied to the eroded candidate instance masks.

The parameters d_{D1} and d_{D2} of the baseline algorithm were optimised, but they have their intrinsic meaning as well. Both are used to constrain pixels classified as *organoid* by their distance to certain image features. The parameters d_{D1} and d_{D2} serve as a threshold distance from a high-variance region to a low-variance *organoid* pixel.

Finally, it is important to note that the network decision threshold T_{org} in the postprocessing routines has one important distinction from the other parameters: it is associated with a particular U-Net model, whereas all other parameters are associated with the image data or the postprocessing routine.

6.3 Algorithm Comparison

In this section, I present comments on algorithm performance and the quantitative evaluation. Since there is no absolute ranking of the algorithms, I provide further comparison and analysis of the experimental results.

Although the *target dataset* was used to optimise parameter values in the baseline algorithm and the deep-learning-free postprocessing routines, the optimisation was done with respect to the SEG measure only, so the other two measures still provide a useful comparison of the individual algorithms on the *target dataset*.

One also has to keep in mind that the evaluation was done only on a small subset of the real image data because creating manual annotations is a relatively time-consuming process. Since the frames for annotation were selected randomly, the dataset is not biased by human perception of a representative dataset, but a representative ratio of all kinds of situations cannot not guaranteed. It is thus possible that some characteristics of the algorithms are not captured in the quantitative evaluation.

The baseline algorithm achieves high SEG and DET scores for both real image datasets. However, the HD^{95} score is relatively high for the *target dataset* and exceeds most results for deep-learning-based algorithms on the *robustness validation dataset*. The baseline algorithm can thus generalise reasonably well in terms of the properties measured by SEG and DET, but its performance in terms of shape similarity (HD^{95}) decreases rapidly for new organoid phenotypes.

The purely deep-learning-based algorithm *Synth-5 + PP-0* yields considerably worse results than other any presented algorithm. The *PP-0* routine was designed according to [54] and was included in the evaluation to verify the observation of [41] that a postprocessing step improves the segmentation accuracy and other desired properties of the result. The evaluation shows that even very simple postprocessing (*PP-1*, *PP-5*) is better than none. Suitable methodology and parameter values can improve the results considerably. In fact, all U-Net models trained for this thesis output a number of components classified as *organoid* including orphaned pixels and do not enforce shape constraints which hold in the input image data. This may be caused by an inappropriate training dataset, wrong configuration of training hyper-parameters, or may be a characteristic of U-Net.

The *PP-0* routine performs relatively well in detection because the DET score does not penalise false positive objects, but exhibits bad scores of both SEG and HD^{95}. In fact, with frame size at most 1344×1024 pixels and organoid of interest approximately centered in the frame, the values $HD^{95} = 951.76$ and $HD^{95} = 762.64$ for the *target dataset* and *robustness validation dataset*, respectively, are very close to the expected maximum score which can be achieved in this measure.

The *Synth-1* routine was used to compare the individual U-Net models. It was chosen because it is simple, but at the same time contains some parameters that can be tuned and thus used to get the best

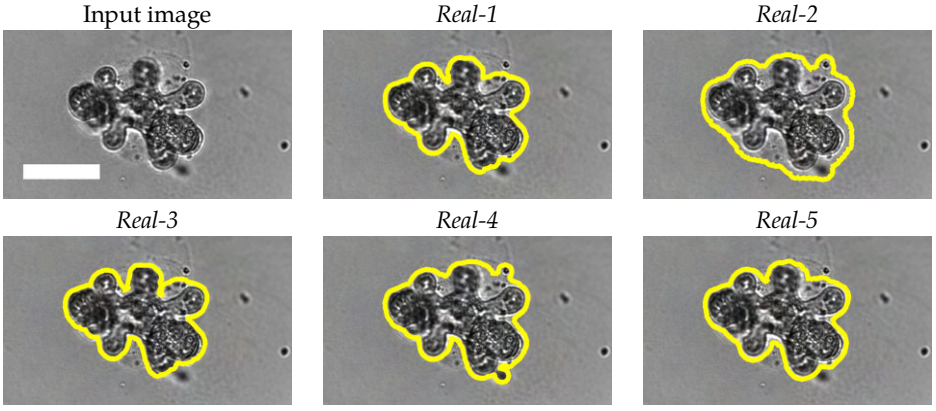
performance out of each network model. The U-Net models with the best scores were then paired with other postprocessing routines.

The *Real* model has generally the worst score out of the six evaluated networks. The reason may lie in the lack of training data (approximately three times less than for other models) or the fact that five different models were used with the same postprocessing parameter values and then averaged. The evaluation on the *robustness validation dataset* indicates that the latter might be the reason; the quantitative results for individual *Real-X* models are fairly different. An example of output of five *Real-X + PP-1* algorithms for the same input frame can be seen in Table 6.5 and the quantitative evaluation is available in the MUNI Thesis Archive.

Training the U-Net with multi-class semantic masks helped the network (*Synth-5*) to learn more relevant features for the *target dataset*. However, algorithms employing this model exhibit a considerable performance drop in all measures for the *robustness validation dataset*. It could be an indication of network overfitting or unsuitable value of the decision threshold T_{org}.

The algorithm with the *PP-5* routine, in fact a refinement of the output of the *PP-0* routine, also shows worse performance than most of the other algorithms with respect to all three measures. Compared

Table 6.5: Comparison of outputs of five *Real-X + PP-1* algorithms on a single frame from a HB sequence ($t = 132$). The annotations are outlined in yellow. All images are cropped to show the necessary context. The white bar corresponds to 100 µm.



to other postprocessing routines, these two routines lack parameters associated with the network that could be tuned. The number of pixels considered in the postprocessing thus cannot be controlled and depends only on the network.

On the other hand, the segmentation correction procedure (the *PP-2* and *PP-3* routines) has proved to be an improvement to the *PP-1* routine; on average, the scores of all three measures improved for both real image datasets. The *PP-2* routine accounts only for under-segmentation, so it cannot solve situations such as the one in Table 5.10. On the contrary, the *PP-3* routine accounts for both under- and over-segmentation. Nevertheless, the routine does not always solve such cases (Table 5.10) – one of the reasons might be the maximum number of iterations of the correction procedure. The algorithm also rarely outputs a mask which exhibits both under- and over-segmentation at the same time (Table 6.6, image d)).

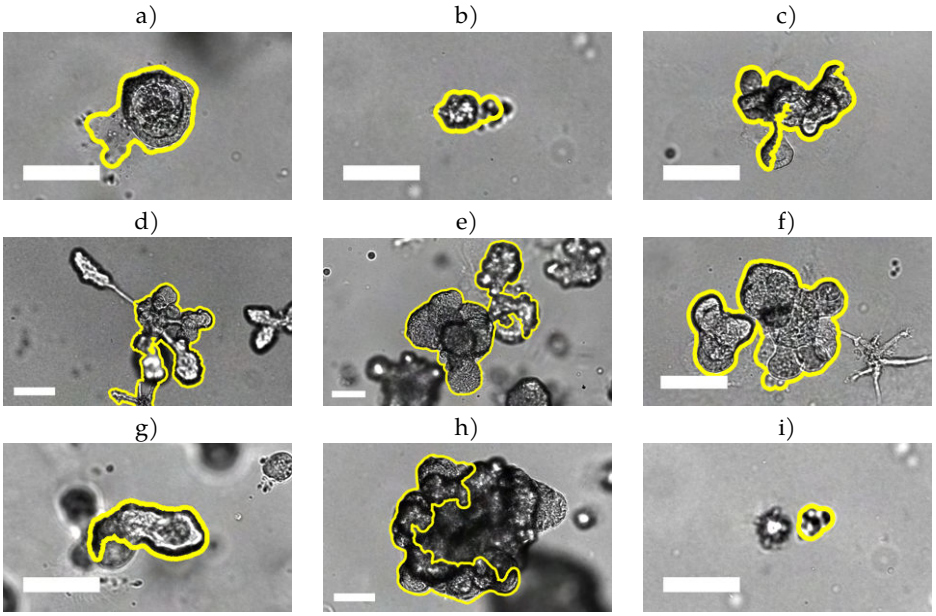
The *PP-2* routine produces reliable results under the constraint of maximum two iterations of the segmentation correction procedure, with less than 0.1% frames requiring more than $R_{max} = 2$ iterations in both datasets. By contrast, the *PP-3* routine exhibits much less stable behaviour in this aspect, with $0.56 - 6.19\%$ frames requiring more iterations.

On average, there are only small differences in the qualitative evaluation of the *PP-2* and *PP-3* routines, with the exception of the *Synth-2 + PP-3* (A) algorithm. This algorithm achieved the same HD^{95} score as human annotators on the *robustness validation dataset*, but the related *Synth-2 + PP-3* (F) algorithm performs relatively poorly with respect to the HD^{95} measure.

The hybrid approach (the *Synth-5 + PP-4* algorithm) has slightly worse score than its individual pipelines for all considered measures on the *target dataset* where it was optimised. However, the combination of a neural network and deep-learning-free processing exhibits interesting behaviour on the *robustness validation dataset*, where it managed to improve the SEG score of the DL-based pipeline (*Synth-5*) and the HD^{95} score of the DL-free pipeline, with similar performance as the individual pipelines with respect to other measures. The hybrid algorithm even performs better than the algorithms with the same U-Net model and the segmentation correction according to DET and SEG with only a slight decrease of performance according to HD^{95}.

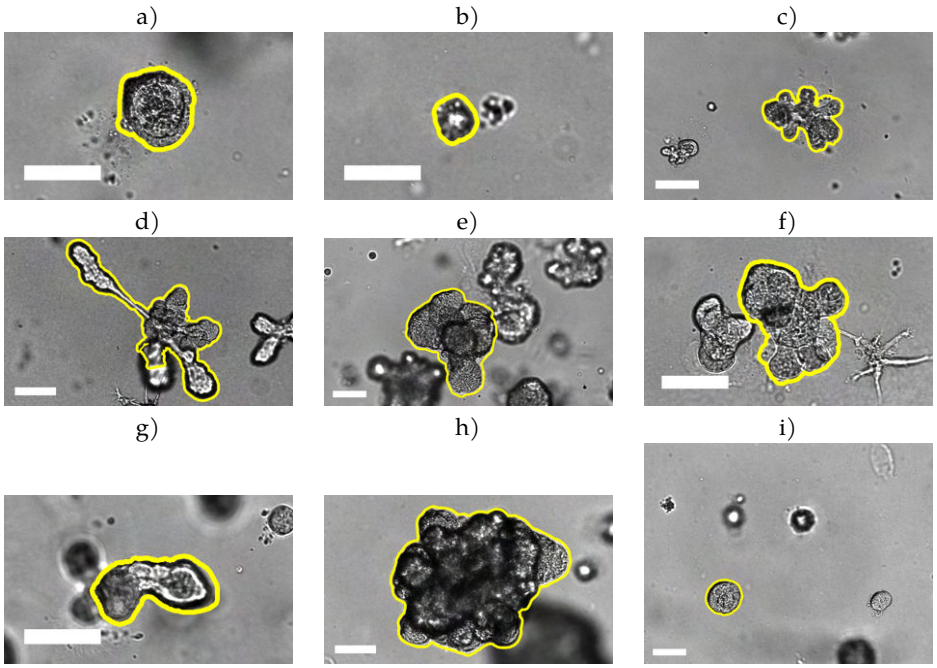
As mentioned in Section 5.2.3, the deep-learning-free part of the *PP-4* routine is based on the baseline algorithm. However, the DL-free pipeline of *PP-4* should produce a real-valued score map whereas baseline outputs a final binary segmentation. The baseline algorithm was thus adjusted in the simplest manner: the final segmentation mask is smoothed using a Gaussian filter with $\sigma = 21$. Additionally,

Table 6.6: Examples of failure of the automated approaches: a) Epithelial structures attached to the organoid of interest; b) defocused organoid of interest lying close to another defocused organoid; c) branch not segmented properly (a HB organoid); d) collision of a defocused organoid-of-interest branch with a defocused organoid and branch not segmented; e) collision of the organoid of interest with defocused organoids; f) collision of the organoid of interest with a focused organoid; g) a HB organoid with less distinct texture partially overlapping with a defocused organoid; h) a MB organoid with less distinct texture partially overlapping with a defocused organoid; and i) wrong detection. All images are cropped to show the necessary context. Contrast of the input images was enhanced for visualisation purposes. The white bars correspond to 100 μm.



the first size filtering operation described in Section 4.1.4 is modified to include all components larger than threshold T_A, and the second size filtering in this step is omitted completely. This yields a better

Table 6.7: Challenging situation with high-quality results: a) Epithelial structures attached to the organoid of interest; b) defocused organoid of interest lying close to another defocused organoid; c) defocused organoid of interest lying close to a focused organoid; d) collision of a defocused organoid-of-interest branch with a defocused organoid; e) collision of the organoid of interest with defocused organoids; f) collision of the organoid of interest with a focused organoid; g) a HB organoid with less distinct texture partially overlapping with a defocused organoid; h) a MB organoid with less distinct texture partially overlapping with a defocused organoid; and i) organoid of interest moving away from the center of the field of view. All images except for the last one are cropped to show the necessary context. Contrast of the input images was enhanced for visualisation purposes. The white bars correspond to 100 μm.



approximation of a feature map and allows better segmentation in case of under-segmentation.

The quantitative comparison of performance of the *PP-4* routine and its partial variants (where one of the three pipelines is omitted) in combination with the same U-Net model (*Synth-5*) is available in the MUNI Thesis Archive. For the *robustness vaidation dataset*, omitting the *Prev* pipeline improved the results slightly according to all measures. On the other hand, omitting the baseline pipeline slightly decreases both SEG and HD^{95} scores and omitting the DL-based pipeline generally results in worse performance.

In general, quantitative results for the *robustness validation dataset* are slightly worse than for the *target dataset*. There are less algorithms with perfect DET score and the HD^{95} score is in average higher than for the *target dataset*. Namely the algorithms with the *Synth-5* model achieved much worse performance on the *robustness validation dataset* with respect to all three measures.

However, some algorithms perform better on the *robustness validation dataset* than on the *target dataset*: for instance *Synth-4 + PP-2* with respect to the SEG score; *Synth-1 + PP-1* with respect to the HD^{95} score; and *Synth-2 + PP-3* (A) with respect to all measures. As it was already mentioned, the *Synth-2 + PP-3* (A) algorithm even achieved human performance according the HD^{95} measure. Other algorithms do not achieve such a low score of the HD^{95} measure, but there are some whose performance on certain frames or sequences is very close to the human performance in both DET and SEG measures. Quantitative evaluation of all algorithms on both dataset using all measures can be found in the MUNI Thesis Archive.

Examples of failures as well as strengths of the automated approach can be inspected in Tables 6.6 and 6.7, respectively.

6.4 Alternative Approaches

In this section, proposals of alternative image analysis methods that could improve the reported algorithm performance are described. Their advantages and disadvantages compared to the approaches described in Chapter 4 are discussed.

As it was already mentioned, there are strong temporal dependencies in the input image data, although they do not hold in a few frames due to manual adjustment of microscope settings. A natural choice would thus be a recurrent neural network. It could solve the cases of focused objects colliding with the organoid of interest if trained with appropriate training data. The training data could, however, become a bottleneck in this application. Firstly, the collision of several focused organoids is quite rare in both real datasets. Secondly, there are neither manual nor synthetic segmentation annotations for such situations. And thirdly, there are no manual segmentation annotations for sequential data. Therefore, training of an RNN model would require extending the manually annotated dataset substantially or training a GAN with appropriate image data – with more than one *organoid* instance allowed in each frame – to create a training dataset for the RNN. However, the algorithm performance evaluation should still be done on the real image data [49].

Extensions of the training data could also improve performance of the U-Net models. This would not solve the situations of several focused organoids in collision, but could improve the segmentation of the organoid of interest colliding with defocused objects and segmentation of the organoid of interest with a focused body and defocused branches. With consistent annotations for all objects in the scene, the U-Net could learn to separate them better [51].

Finally, one could make use of more advanced deep-learning-free image processing methods. For example, active contours could be employed to enforce shape coherence between frames and increase the speed of computation [11]. This is possible thanks to the assumptions of a smooth shape evolution and slow motion of the organoid. However, the speed function for contour evolution would have to be designed very carefully with respect to the texture characteristics of organoids and their surrounding. The algorithm would also have to deal with the asynchronous adjustments of microscope view for efficient contour initialisation in a particular frame.

Conclusion

Several algorithms for automatic segmentation and tracking of organoids in brightfield microscopy image data were presented in this thesis. The input image data present a number of challenges that have to be tackled by the algorithms, including colliding and occluding object, out-of-focus structures, and high heterogeneity in shape and texture across organoid phenotypes.

Following the review of the state-of-the-art methods for biomedical image analysis, a deep-learning-free algorithm and a family of deep-learning-based algorithms were developed. Parameter values in the deep-learning-free routines were optimised for a single real image dataset. Selected methods were subsequently evaluated on two real image datasets using three different quantitative measures to quantify algorithm performance and ability to generalise on new organoid phenotypes. A simple application was developed to demonstrate the approaches described in this thesis.

Six U-Net models and five deep-learning-free postprocessing routines were created and evaluated to compare the influence of different training hyper-parameters and postprocessing options on the algorithm output. While some of the algorithms perform relatively poorly, several have proven to be robust with respect to variations in both known and new organoid phenotypes. These include namely a group of seven deep-learning-based algorithms with postprocessing driven by the input data characteristics and a segmentation correction routine.

I recommend to choose one of the algorithms called *Synth-2 + PP-3* (A), *Synth-4 + PP-3* (A), *Synth-4 + PP-3* (F), and *Synth-4 + PP-2* for practical use. Their output is robust and of high quality with respect to all three evaluation measures. I advise the users to process a representative subset of their image data with these four algorithms and subsequently choose one of them based on their desired output quality measurement. The measures used in this thesis quantify several desired properties of the algorithm output but might still omit a certain characteristic that is crucial for other users.

The recommended *Synth-4 + PP-2* algorithm is the method described in our manuscript [9]. Although it performs slightly worse with respect to the SEG measure compared to the other recommended

algorithms, its output is of high quality on average and it is robust across different phenotypes of organoids.

Alternative approaches to organoid detection and segmentation were discussed in Chapter 6. To name a few, recurrent neural networks can be considered to increase segmentation accuracy in cases when a focused organoid collides with the organoid of interest, and segmentation based on the active contours was proposed to increase the speed of computation and to enforce shape coherence of segmentation masks in consecutive frames.

Bibliography

1. ORTIZ-DE-SOLÓRZANO, C.; MUÑOZ-BARRUTIA, A.; MEIJERING, E.; KOZUBEK, M. Toward a Morphodynamic Model of the Cell: Signal processing for cell modeling. *IEEE Signal Processing Magazine*. 2015, vol. 32, no. 1, pp. 20–29. Available from DOI: 10.1109/MSP.2014.2358263.
2. BIAN, X.; LI, G.; WANG, C., et al. A deep learning model for detection and tracking in high-throughput images of organoid. *Computers in Biology and Medicine*. 2021, vol. 134, p. 104490. ISSN 0010-4825. Available from DOI: https://doi.org/10.1016/j.compbiomed.2021.104490.
3. XING, F.; XIE, Y.; SU, H., et al. Deep Learning in Microscopy Image Analysis: A Survey. *IEEE Transactions on Neural Networks and Learning Systems*. 2018, vol. 29, no. 10, pp. 4550–4568. Available from DOI: 10.1109/TNNLS.2017.2766168.
4. MARTIN, C. B.; CHANE, C. S.; CLOUCHOUX, C.; HISTACE, A. Recent Trends and Perspectives in Cerebral Organoids Imaging and Analysis. *Frontier in Neuroscience*. 2021, vol. 15. Available from DOI: 10.3389/fnins.2021.629067.
5. CAICEDO, J. C.; ROTH, J.; GOODMAN, A., et al. Evaluation of Deep Learning Strategies for Nucleus Segmentation in Fluorescence Images. *Cytometry. Part A : the journal of the International Society for Analytical Cytology*. 2019, vol. 95, no. 9, pp. 952–965. Available from DOI: https://doi.org/10.1002/cyto.a.23863.
6. HASNAIN, Z.; FRASER, A. K.; GEORGESS, D., et al. OrgDyn: feature- and model-based characterization of spatial and temporal organoid dynamics. *Bioinformatics*. 2020, vol. 36, no. 10, pp. 3292–3294. ISSN 1367-4803. Available from DOI: 10.1093/bioinformatics/btaa096.
7. LEE, S.; CHANG, J.; KANG, S.-M., et al. High-throughput formation and image-based analysis of basal-in mammary organoids in 384-well plates. *Scientific reports*. 2022, vol. 12. Available from DOI: https://doi.org/10.1038/s41598-021-03739-1.

8. BULIN, A.-L.; BROEKGAARDEN, M.; HASAN, T. Comprehensive high-throughput image analysis for therapeutic efficacy of architecturally complex heterotypic organoids. *Scientific reports*. 2017, vol. 7. Available from DOI: https://doi.org/10.1038/s41598-017-16622-9.

9. DUPKANIČOVÁ, L.; WIESNER, D.; SUMBAL, J.; KOLEDOVA, Z.; MAŠKA, M. Segmentation and Shape Tracking of Mammary Epithelial Organoids in Brightfield Microscopy. [N.d.]. *IEEE Transactions on Medical Imaging*, under review, 2022.

10. DAWANT, B. M.; ZIJDENBOS, A. P. Image Segmentation. In: SONKA, M.; FITZPATRICK, J. M. (eds.). *Handbook of medical imaging*. SPIE Optical Engineering Press, 2000, pp. 71–127. ISBN 0-8194-3622-4. Available from DOI: https://doi.org/10.1117/3.831079.

11. HAIDEKKER, Mark. *Advanced biomedical image analysis*. John Wiley & Sons, 2010. ISBN 978-0-470-62458-6.

12. SONKA, M.; FITZPATRICK, J. M. (eds.). *Handbook of medical imaging*. SPIE Optical Engineering Press, 2000. ISBN 0-8194-3622-4. Available from DOI: https://doi.org/10.1117/3.831079.

13. ACTON, S. T.; RAY, N. *Biomedical Image Analysis: Tracking*. Ed. by BOVIK, A. C. Morgan & Claypool Publishers, 2006. ISBN 1-598290-18-5. Available from DOI: https://doi.org/10.2200/S00002ED1V01Y200508IVM002.

14. ACTON, S. T.; RAY, N. *Biomedical Image Analysis: Segmentation*. Ed. by BOVIK, A. C. Morgan & Claypool Publishers, 2009. ISBN 9781598290202. Available from DOI: doi:10.2200/s00133ed1v01y200807ivm009.

15. DZYUBACHYK, O.; VAN CAPPELLEN, W. A.; ESSERS, J., et al. Advanced Level-Set-Based Cell Tracking in Time-Lapse Fluorescence Microscopy. *IEEE Transactions on Medical Imaging*. 2010, vol. 29, no. 3, pp. 852–867. Available from DOI: 10.1109/TMI.2009.2038693.

16. CHEN, X.; ZHENG, B.; LIU, H. Optical and digital microscopic imaging techniques and applications in pathology. *Analytical cellular pathology (Amsterdam)*. 2011, vol. 34, pp. 5–18. Available from DOI: 10.3233/ACP-2011-0006.

17. MOEN, E.; BANNON, D.; KUDO, T., et al. Deep learning for cellular image analysis. *Nature Methods*. 2019, vol. 16, pp. 1233–1246. Available from DOI: https://doi.org/10.1038/s41592-019-0403-1.

18. KHERLOPIAN, A. R.; SONG, T.; DUAN, Q., et al. A review of imaging techniques for systems biology. *BMC systems biology*. 2008, vol. 2. Available from DOI: https://doi.org/10.1186/1752-0509-2-74.

19. MAYER, J.; ROBERT-MORENO, A.; SHARPE, J.; SWOGER, J. Attenuation artifacts in light sheet fluorescence microscopy corrected by OPTiSPIM. *Light: Science & Applications*. 2018. Available from DOI: https://doi.org/10.1038/s41377-018-0068-z.

20. MEIJERING, E. Cell Segmentation: 50 Years Down the Road [Life Sciences]. *IEEE Signal Processing Magazine*. 2012, vol. 29, no. 5, pp. 140–145. Available from DOI: 10.1109/MSP.2012.2204190.

21. FITZPATRICK, J. M.; HILL, D. L. G.; MAURER JR., C. R. Image Registration. In: SONKA, M.; FITZPATRICK, J. M. (eds.). *Handbook of medical imaging*. SPIE Optical Engineering Press, 2000, pp. 447–513. ISBN 0-8194-3622-4. Available from DOI: https://doi.org/10.1117/3.831079.

22. DUNN, K. W.; FU, C.; HO, D. J., et al. DeepSynth: Three-dimensional nuclear segmentation of biological images using neural networks trained with synthetic data. *Scientific reports*. 2019, vol. 9. Available from DOI: https://doi.org/10.1038/s41598-019-54244-5.

23. ARBELLE, A.; RAVIV, T. R. Microscopy Cell Segmentation Via Convolutional LSTM Networks. In: *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. 2019, pp. 1008–1012. Available from DOI: 10.1109/ISBI.2019.8759447.

24. LI, T.; SAHU, A. K.; TALWALKAR, A.; SMITH, V. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Processing Magazine*. 2020, vol. 37, no. 3, pp. 50–60. Available from DOI: 10.1109/MSP.2020.2975749.

25. MALIN, B. A.; EMAM, K. E.; O'KEEFE, C. M. Biomedical data privacy: problems, perspectives, and recent advances. *Journal of the American Medical Informatics Association : JAMIA*. 2013, vol. 20, no. 1, pp. 2–6. Available from DOI: https://doi.org/10.1136/amiajnl-2012-001509.

26. SHUVO, B.; AHOMMED, R.; REZA, S.; HASHEM, M.M.A. CNL-UNet: A novel lightweight deep learning architecture for multimodal biomedical image segmentation with false output suppression. *Biomedical Signal Processing and Control*. 2021, vol. 70, p. 102959. ISSN 1746-8094. Available from DOI: https://doi.org/10.1016/j.bspc.2021.102959.

27. WILLIAMS, E.; MOORE, J.; LI, S., et al. Image Data Resource: a bioimage data integration and publication platform. *Nature Methods*. 2017, vol. 14, pp. 775–781. Available from DOI: https://doi.org/10.1038/nmeth.4326.

28. LJOSA, V.; SOKOLNICKI, K. L.; CARPENTER, A. E. Annotated high-throughput microscopy image sets for validation. *Nature Methods*. 2012, vol. 9. Available from DOI: http://dx.doi.org/10.1038/nmeth.2083.

29. ULMAN, V.; MAŠKA, M.; MAGNUSSON, K. E. G., et al. An objective comparison of cell-tracking algorithms. 2017, vol. 14, pp. 1141–1152. Available from DOI: https://doi.org/10.1038/nmeth.4473.

30. XU, C.; PHAM, D. L.; PRINCE, J. L. Image Segmentation Using Deformable Models. In: SONKA, M.; FITZPATRICK, J. M. (eds.). *Handbook of medical imaging*. SPIE Optical Engineering Press, 2000, pp. 129–174. ISBN 0-8194-3622-4. Available from DOI: https://doi.org/10.1117/3.831079.

31. RICCI, P.; GAVRYUSEV, V.; MÜLLENBROICH, C., et al. Removing striping artifacts in light-sheet fluorescence microscopy: a review. *Progress in Biophysics and Molecular Biology*. 2022, vol. 168,

pp. 52–65. ISSN 0079-6107. Available from DOI: https://doi.org/10.1016/j.pbiomolbio.2021.07.003.

32. LOEW, M. H. Feature Extraction. In: SONKA, M.; FITZPATRICK, J. M. (eds.). *Handbook of medical imaging*. SPIE Optical Engineering Press, 2000, pp. 273–341. ISBN 0-8194-3622-4. Available from DOI: https://doi.org/10.1117/3.831079.

33. MCCANN, M. T.; OZOLEK, J. A.; CASTRO, C. A., et al. Automated Histology Analysis: Opportunities for signal processing. *IEEE Signal Processing Magazine*. 2015, vol. 32, no. 1, pp. 78–87. Available from DOI: 10.1109/MSP.2014.2346443.

34. GOUTSIAS, J.; BATMAN, S. Morphological methods for biomedical image analysis. In: SONKA, M.; FITZPATRICK, J. M. (eds.). *Handbook of medical imaging*. SPIE Optical Engineering Press, 2000, pp. 175–272. ISBN 0-8194-3622-4. Available from DOI: https://doi.org/10.1117/3.831079.

35. BROWN, M. S.; MCNITT-GRAY, M. F. Medical Image Interpretation. In: SONKA, M.; FITZPATRICK, J. M. (eds.). *Handbook of medical imaging*. SPIE Optical Engineering Press, 2000, pp. 399–445. ISBN 0-8194-3622-4. Available from DOI: https://doi.org/10.1117/3.831079.

36. BRACEWELL, R. N. *The Fourier Transform and Its Applications*. McGraw-Hill, New York, 1986. ISBN 0-07-303938-1.

37. SERRA, J. Introduction to Mathematical Morphology. In: *Computer Vision, Graphics, and Image Processing*. 1986, vol. 35, pp. 283–305.

38. SOILLE, P. *Morphological Image Analysis: Principles and Applications*. Springer Science & Business Media, 2013. ISBN 978-3-540-42988-3.

39. PINHEIRO, P. O.; COLLOBERT, R. From image-level to pixel-level labeling with Convolutional Networks. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1713–1721. Available from DOI: 10.1109/CVPR.2015.7298780.

40. LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*. 2015, vol. 521, pp. 436–444. Available from DOI: https://doi.org/10.1038/nature14539.

41. VAN VALEN, D. A.; KUDO, T.; LANE, K. M., et al. Deep Learning Automates the Quantitative Analysis of Individual Cells in Live-Cell Imaging Experiments. *PLoS Comput Biol.* 2016, vol. 12, no. 11. Available from DOI: 10.1371/journal.pcbi.1005177.

42. CHEN, X.; PAN, L. A Survey of Graph Cuts/Graph Search Based Medical Image Segmentation. *IEEE Reviews in Biomedical Engineering*. 2018, vol. 11, pp. 112–124. Available from DOI: 10.1109/RBME.2018.2798701.

43. BOYKOV, Y. Y.; JOLLY, M.-P. In: *Proceedings eighth IEEE international conference on computer vision. ICCV 2001*. 2001, vol. 1, pp. 105–112. Available also from: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=937505.

44. AIZENBERG, I.; AIZENBERG, N.; HILTNER, J., et al. Cellular neural networks and computational intelligence in medical image processing. *Image and Vision Computing*. 2001, vol. 19, no. 4, pp. 177–183. ISSN 0262-8856. Available from DOI: https://doi.org/10.1016/S0262-8856(00)00066-4.

45. *A.I. Wiki* [online]. Pathmind, Inc., 2020 [visited on 2022-04-18]. Available from: https://wiki.pathmind.com/.

46. NGUYEN, K.; FOOKES, C.; SRIDHARAN, S. Improving deep convolutional neural networks with unsupervised feature learning. In: *2015 IEEE International Conference on Image Processing (ICIP)*. 2015, pp. 2270–2274. Available from DOI: 10.1109/ICIP.2015.7351206.

47. BECKER, S.; PLUMBLEY, M. Unsupervised neural network learning procedures for feature extraction and classification. *Applied Intelligence*. 1996, pp. 185–203. Available from DOI: https://doi.org/10.1007/BF00126625.

48. VESELÝ, K.; HANNEMANN, M.; BURGET, L. Semi-supervised training of Deep Neural Networks. In: *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*. 2013, pp. 267–272. Available from DOI: 10.1109/ASRU.2013.6707741.

49. BOWYER, K. W. Validation of Medical Image Analysis Techniques. In: SONKA, M.; FITZPATRICK, J. M. (eds.). *Handbook of medical imaging*. SPIE Optical Engineering Press, 2000, pp. 567–607. ISBN 0-8194-3622-4. Available from DOI: https://doi.org/10.1117/3.831079.

50. LE, T. A.; BAYDIN, A. G.; ZINKOV, R.; WOOD, F. Using synthetic data to train neural networks is model-based reasoning. In: *2017 International Joint Conference on Neural Networks (IJCNN)*. 2017, pp. 3514–3521. Available from DOI: 10.1109/IJCNN.2017.7966298.

51. RONNEBERGER, O.; FISCHER, P.; BROX, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In: NAVAB, N.; HORNEGGER, J.; WELLS, W. M.; FRANGI, A. F. (eds.). *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Cham: Springer International Publishing, 2015, pp. 234–241. ISBN 978-3-319-24574-4. Available from DOI: doi.org/10.1007/978-3-319-24574-4_28.

52. ALBAWI, S.; MOHAMMED, T. Abed; AL-ZAWI, S. Understanding of a convolutional neural network. In: *2017 International Conference on Engineering and Technology (ICET)*. 2017, pp. 1–6. Available from DOI: 10.1109/ICEngTechnol.2017.8308186.

53. SHELHAMER, E.; LONG, J.; DARRELL, T. Fully Convolutional Networks for Semantic Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2017, vol. 39, no. 4, pp. 640–651. Available from DOI: 10.1109/TPAMI.2016.2572683.

54. FALK, T.; MAI, D.; BENSCH, R., et al. U-Net: deep learning for cell counting, detection, and morphometry. In: *Nature Methods*. 2019, vol. 16, pp. 67–70. Available from DOI: https://doi.org/10.1038/s41592-018-0261-2.

55. STACHOWICZ, J.; SÖDERBERG, M. J.; IVARSSON, A. *Recreation, U-net: Convolutional networks for biomedical image segmentation* [online]. 2020 [visited on 2021-11-18]. Available from: https://github.com/jacobstac/Recreation-of-U-net-Convolutional-networks-for-biomedical-image-segmentation.

56. GÓMEZ-DE-MARISCAL, E.; MAŠKA, M.; KOTRBOVÁ, A., et al. Deep-Learning-Based Segmentation of Small Extracellular Vesicles in Transmission Electron Microscopy Images. In: *Scientific Reports*. 2019, vol. 9. Available from DOI: https://doi.org/10.1038/s41598-019-49431-3.

57. ZENG, Z.; XIE, W.; ZHANG, Y.; LU, Y. RIC-Unet: An Improved Neural Network Based on Unet for Nuclei Segmentation in Histology Images. *IEEE Access*. 2019, vol. 7, pp. 21420–21428. Available from DOI: 10.1109/ACCESS.2019.2896920.

58. HUANG, H.; LIN, L.; TONG, R., et al. UNet 3+: A Full-Scale Connected UNet for Medical Image Segmentation. In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 1055–1059. Available from DOI: 10.1109/ICASSP40776.2020.9053405.

59. YAN, X.; TANG, H.; SUN, S., et al. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. AFTer-UNet: Axial Fusion Transformer UNet for Medical Image Segmentation. 2022.

60. BADRINARAYANAN, V.; KENDALL, A.; CIPOLLA, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2017, vol. 39, no. 12, pp. 2481–2495. Available from DOI: 10.1109/TPAMI.2016.2644615.

61. ÇIÇEK, Ö.; ABDULKADIR, A.; LIENKAMP, S. S., et al. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. In: OURSELIN, S.; WELLS, W.S.; SABUNCU, M. R.; UNAL, G.; JOSKOWICZ, L. (eds.). *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Springer, 2016, vol. 9901, pp. 424–432. LNCS. Available also from: http://lmb.informatik.uni-freiburg.de/Publications/2016/CABR16.

62. HE, K.; GKIOXARI, G.; DOLLÁR, P.; GIRSHICK, R. Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 2020, vol. 42, no. 2, pp. 386–397. Available from DOI: 10.1109/TPAMI.2018.2844175.

63. LIN, T.-Y.; MAIRE, M.; BELONGIE, S., et al. Microsoft COCO: Common Objects in Context. In: FLEET, D.; PAJDLA, T.; SCHIELE, B.; TUYTELAARS, T. (eds.). *Computer Vision – ECCV 2014*. Cham: Springer International Publishing, 2014, pp. 740–755. ISBN 978-3-319-10602-1. Available from DOI: https://doi.org/10.1007/978-3-319-10602-1_48.

64. VUOLA, A. O.; AKRAM, S. U.; KANNALA, J. Mask-RCNN and U-Net Ensembled for Nuclei Segmentation. In: *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. 2019, pp. 208–212. Available from DOI: 10.1109/ISBI.2019.8759574.

65. MILLETARI, F.; NAVAB, N.; AHMADI, S.-A. V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. In: *2016 Fourth International Conference on 3D Vision (3DV)*. 2016, pp. 565–571. Available from DOI: 10.1109/3DV.2016.79.

66. HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural computation*. 1997, vol. 9, no. 8, pp. 1735–1780. ISSN 0899-7667. Available from DOI: 10.1162/neco.1997.9.8.1735.

67. GREFF, K.; SRIVASTAVA, R. K.; KOUTNÍK, J., et al. LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*. 2017, vol. 28, no. 10, pp. 2222–2232. Available from DOI: 10.1109/TNNLS.2016.2582924.

68. PAYER, C.; ŠTERN, D.; NEFF, T., et al. Instance Segmentation and Tracking with Cosine Embeddings and Recurrent Hourglass Networks. In: FRANGI, A. F.; SCHNABEL, J. A.; DAVATZIKOS, C.; ALBEROLA-LÓPEZ, C.; FICHTINGER, G. (eds.). *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*. Cham: Springer International Publishing, 2018, pp. 3–11. ISBN 978-3-030-00934-2. Available from DOI: https://doi.org/10.1007/978-3-030-00934-2_1.

69. ARBELLE, A.; RAVIV, T. R. Microscopy cell segmentation via adversarial neural networks. In: *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. 2018, pp. 645–648. Available from DOI: 10.1109/ISBI.2018.8363657.

70. MAHMOOD, F.; BORDERS, D.; CHEN, R. J., et al. Deep Adversarial Training for Multi-Organ Nuclei Segmentation in Histopathology Images. *IEEE Transactions on Medical Imaging*. 2020, vol. 39, no. 11, pp. 3257–3267. Available from DOI: 10.1109/TMI.2019.2927182.

71. CHUA, L. O.; YANG, L. Cellular neural networks: theory. *IEEE Transactions on Circuits and Systems*. 1988, vol. 35, no. 10, pp. 1257–1272.

72. CHUA, L. O.; YANG, L. Cellular neural networks: applications. *IEEE Transactions on Circuits and Systems*. 1988, vol. 35, no. 10, pp. 1273–1290. Available from DOI: 10.1109/31.7601.

73. CUEVAS, E.; DÍAZ-CORTES, M.-A.; MEZURA-MONTES, E. Corner detection of intensity images with cellular neural networks (CNN) and evolutionary techniques. *Neurocomputing*. 2019, vol. 347, pp. 82–93. ISSN 0925-2312. Available from DOI: https://doi.org/10.1016/j.neucom.2019.03.014.

74. XU, X.; LU, Q.; WANG, T., et al. Efficient Hardware Implementation of Cellular Neural Networks with Incremental Quantization and Early Exit. *J. Emerg. Technol. Comput. Syst.* 2018, vol. 14, no. 4. ISSN 1550-4832. Available from DOI: 10.1145/3264817.

75. HU, X.; FENG, G.; DUAN, S.; LIU, L. A Memristive Multilayer Cellular Neural Network With Applications to Image Processing. *IEEE Transactions on Neural Networks and Learning Systems*. 2017, vol. 28, no. 8, pp. 1889–1901. Available from DOI: 10.1109/TNNLS.2016.2552640.

76. AIZENBERG, I. N. Processing of noisy and small-detailed gray-scale images using cellular neural networks. *Journal of Electronic Imaging*. 1997, vol. 6, no. 3, pp. 272–285.

77. SHITONG, W.; MIN, W. A new detection algorithm (NDA) based on fuzzy cellular neural networks for white blood cell detection. *IEEE Transactions on Information Technology in Biomedicine*. 2006, vol. 10, no. 1, pp. 5–10. Available from DOI: 10.1109/TITB.2005.855545.

78. ISOLA, P.; ZHU, J.-Y.; ZHOU, T.; EFROS, A. A. Image-to-Image Translation with Conditional Adversarial Networks. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5967–5976. Available from DOI: 10.1109/CVPR.2017.632.

79. WANG, T.-C.; LIU, M.-Y.; ZHU, J.-Y., et al. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 8798–8807.

80. PAL, K. K.; SUDEEP, K. S. Preprocessing for image classification by convolutional neural networks. In: *2016 IEEE International Conference on Recent Trends in Electronics, Information Communication Technology (RTEICT)*. 2016, pp. 1778–1781. Available from DOI: 10.1109/RTEICT.2016.7808140.

81. *Evaluation Methodology* [online]. Cell Tracking Challenge organisers, 2020 [visited on 2021-12-08]. Available from: http://celltrackingchallenge.net/evaluation-methodology/.

82. MAŠKA, M.; ULMAN, V.; SVOBODA, D.; MATULA, P.; MATULA, P., et al. A benchmark for comparison of cell tracking algorithms. *Bioinformatics*. 2014, vol. 30, no. 11, pp. 1609–1617. ISSN 1367-4803. Available from DOI: 10.1093/bioinformatics/btu080.

83. TAHA, A. A.; HANBURY, A. Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool. *BMC Medical Imaging*. 2015, vol. 15, no. 29. Available from DOI: https://doi.org/10.1186/s12880-015-0068-x.

84. MATULA, P.; MAŠKA, M.; SOROKIN, D. V.; MATULA, P., et al. Cell Tracking Accuracy Measurement Based on Comparison of Acyclic Oriented Graphs. *PLOS ONE*. 2015. Available from DOI: https://doi.org/10.1371/journal.pone.0144959.

85. KLETTE, R.; ROSENFELD, A. *Digital Geometry*. Morgan Kaufmann, 2004. ISBN 978-1-55860-861-0. Available from DOI: https://doi.org/10.1016/B978-1-55860-861-0.X5000-7.

86. HUTTENLOCHER, D. P.; KLANDERMAN, G. A.; RUCKLIDGE, W. J. Comparing images using the Hausdorff distance. *IEEE Transactions on pattern analysis and machine intelligence*. 1993, vol. 15, no. 9, pp. 850–863.

87. ALT, H.; GUIBAS, L. J. Discrete Geometric Shapes: Matching, Interpolation, and Approximation. In: SACK, J.-R.; URRUTIA, J. (eds.). *Handbook of Computational Geometry*. Amsterdam: North-Holland, 2000, pp. 121–153. ISBN 978-0-444-82537-7. Available from DOI: https://doi.org/10.1016/B978-044482537-7/50004-8.

88. YEGHIAZARYAN, V.; VOICULESCU, I. Family of boundary overlap metrics for the evaluation of medical image segmentation. *Journal of medical imaging (Bellingham, Wash.)* 2018, vol. 5, no. 1. Available from DOI: 10.1117/1.JMI.5.1.015006.

89. SCHINDELIN, J.; ARGANDA-CARRERAS, I.; FRISE, E., et al. Fiji: an open-source platform for biological-image analysis. In: *Nature Methods*. 2012, vol. 9, pp. 676–682. Available from DOI: https://doi.org/10.1038/nmeth.2019.

90. *Computer Vision Group, Uni Freiburg* [online]. Freiburg: LMB, University of Freiburg, 2021 [visited on 2021-02-04]. Available from: https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/.

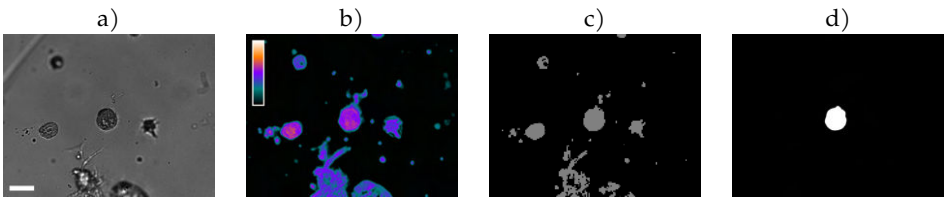
A Appendix 1: Application Documentation

A simple console application that provides the implementation of algorithms evaluated and presented in Section 5.3 can be downloaded from the MUNI Thesis Archive.

The application requires a UNIX-based operating system, a MATLAB licence (MATLAB Version R2020a and DIPimage toolbox Version 3.0.b9) and Fiji software with the U-Net segmentation plug-in by Falk et al. [54]. Deep-learning segmentation is run in Fiji with the U-Net plug-in and uses GPU if any is detected. All deep-learning-free codes are written in MATLAB and run on CPU. The application was tested on the same workstation that was used for algorithm evaluation: a workstation with AMD Opteron 6348, 120 GB of RAM, Ubuntu 20.4 LTS, and equipped with a single NVIDIA Quadro P6000 graphics card with 24 GB of RAM.

The application accepts a directory with the input image data, performs segmentation of all image sequences in the input directory, and saves U-Net scores, argmax output, and the final binary segmentation of the organoid of interest in a given output directory (Table A.1). The user can choose from a number of U-Net models and postprocessing scripts presented in this thesis, use the baseline algorithm presented in this thesis, or use their own U-Net model if its definition is in suitable format. In the latter case, the decision threshold T_{org} of the network must be also input to the software. The user is also asked to input two parameters irrespective of the chosen algorithm: the minimum branch

Table A.1: Examples of output images of the application demonstration: a) an input frame; b) U-Net prediction for the organoid channel displayed with a LUT and with scale in the top left corner; c) argmax of the U-Net prediction; and d) the final binary mask of the organoid of interest. The white bar corresponds to 100 µm.



diameter d_{DE} or d_O and the minimum organoid size T_A. Parameter T_A must be chosen with respect to the value of d_{DE} since the size filtering is applied to the eroded binary masks.

An example use case of this application can be found in the supplementary material in the MUNI Thesis Archive. Example input dataset contains one real image sequence per organoid phenotype (directory `input_data/`). Each sequence was processed with a different algorithm proposed in this thesis using the following commands:

- the CY sequence using the *Synth-4 + PP-2* algorithm with values of $d_{DE} = 35$, $T_A = 300$
  ```
  ./unet_codes_for_thesis/automatic/segmentation.sh -i
  /home/xuser/input_data/CY/ -o /home/xuser/results/ -a
  4 -p 2 -d 35 -s 300
  ```
- the NB sequence using the *Synth-2 + PP-3* (A) algorithm with values of $d_{DE} = 25$, $T_A = 300$
  ```
  ./unet_codes_for_thesis/automatic/segmentation.sh -i
  /home/xuser/input_data/NB/ -o /home/xuser/results/ -a
  2 -p 31 -d 25 -s 300
  ```
- the MB sequence using the *Synth-5 + PP-4* algorithm with values of $d_{DE} = 35$, $T_A = 300$
  ```
  ./unet_codes_for_thesis/automatic/segmentation.sh -i
  /home/xuser/input_data/MB/ -o /home/xuser/results/ -a
  5 -p 4 -d 35 -s 300
  ```
- the HB sequence using the *baseline* algorithm with values of $d_O = 5$, $T_A = 300$
  ```
  ./unet_codes_for_thesis/automatic/segmentation.sh -i
  /home/xuser/input_data/HB/ -o /home/xuser/results/ -a
  0 -d 5 -s 300
  ```
- the LB sequence using the *Real-5 + PP-1* algorithm with values of $d_{DE} = 5$, $T_A = 300$
  ```
  ./unet_codes_for_thesis/automatic/segmentation.sh -i
  /home/xuser/input_data/LB/ -o /home/xuser/results/ -a
  r5 -p 1 -d 5 -s 300
  ```

The results are saved in the `results/` directory: U-Net scores in the `results/score/` directory, argmax output in the `results/argmax/` directory, and the final binary segmentation of the organoid of interest in the `results/bin/` directory. Examples of the output images are presented in Table A.1.