

INSTITUTE OF COMPUTER
SCIENCES
Master in Artificial Intelligence and Data
Science

Universitätsstr. 1 D–40225 Düsseldorf



Heinrich Heine
Universität
Düsseldorf

AI-based fluorescent labeling for cell line development

Hanna Pankova

Master thesis

Date of issue: 01. April 2022
Date of submission: 29. August 2022
Reviewers: Prof. Dr. Markus Kollmann
Dr. Wolfgang Halter

Erklärung

Hiermit versichere ich, dass ich diese Master thesis selbstständig verfasst habe. Ich habe dazu keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Düsseldorf, den 29. August 2022

Hanna Pankova

Abstract

Cell line development is an expensive and time-consuming process, however that is the most modern approach for producing the proteins needed in pharmaceuticals.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Notation	2
2	Domain knowledge	3
2.1	Biology	3
2.1.1	Cell line development process	3
2.1.2	Project specifications of cell line development for Merck KgaA . . .	3
2.2	Deep learning and machine learning basics	3
2.2.1	Neural networks	3
2.2.2	Clustering	4
2.3	Imaging	4
2.3.1	Digital imaging	4
2.3.2	Microscopy imaging	4
3	Model training	5
3.1	Neural network architecture	5
3.2	Loss functions	5
3.3	Available data	5
3.4	Training costs estimation	5
3.5	Augmentations	6
3.5.1	Smart augmentations for rotation and scaling	6
3.6	Convergence	6
3.7	Model setup	6
3.7.1	Weight Initialization	6
3.7.2	Regularization	6
3.7.3	Optimizers	7
4	Nuclei	8
4.1	Preprocessing	8
4.1.1	Thresholding algorithms	8
4.2	Training and predictions	9

4.2.1	Convergence	9
4.2.2	Predictions quality	9
4.3	Postprocessing for nuclei segmentation	10
4.4	Influence of scaling on predictions quality	10
5	Actin	11
5.1	Preprocessing	11
5.2	Training and predictions	12
5.3	Postprocessing	12
5.4	Combination of nuclei and actin predictions	12
5.5	Prediction quality on the crop's border	12
5.6	Generalizability across phenotypes	12
6	Golgi	13
6.1	Preprocessing	13
6.1.1	Background removal algorithms	13
6.2	Predictions	14
6.3	Postprocessing	14
6.4	Alternative ways to improve predictions	14
6.4.1	Noise reduction methods	14
6.4.2	Asymmetrical losses	14
6.4.3	Use of gradient in loss	14
7	Nucleolus and full cell prediction	15
7.1	Preprocessing	15
7.2	Predictions	15
7.3	Postprocessing	15
7.4	Combination of nucleolus and nuclei	15
8	Model Evaluation	16
8.1	Crops combination technique	16
8.2	Metrics for downstream tasks	16
8.3	Influence of different loss functions on metrics for downstream tasks	16
9	Stability	17

9.1 Artificial corruptions	17
9.2 Real corruptions	17
9.3 Influence of corruptions on metrics for downstream tasks	17
9.4 Improving predictions with additional corruption augmentations	17
10 Information in the UNET embeddings	18
10.1 Dimensionality reduction methods	18
10.2 UMAP, t-SNE, PCA	18
10.3 Autoencoder embeddings as an alternative	18
10.4 Clustering of corrupted samples	21
11 Drift detection	22
11.1 A need to detect drift	22
11.2 MMD	22
11.2.1 Offline drift detection	22
11.2.2 Online drift detection	22
11.3 MMD on UNet embeddings	22
11.4 Clustering on UNet embeddings	22
11.5 Clustering on autoencoder embeddings	22
12 Software Tools	23
12.1 Foundry, Palantir	23
12.2 AWS	23
12.3 Streamlit	23
12.4 ImageJ, CellProfiler	23
13 Summary	24
List of Figures	25
List of Tables	25

1 Introduction

1.1 Motivation

1.2 Notation

2 Domain knowledge

2.1 Biology

2.1.1 Cell line development process

General theory behind the cell line development process. Starting from what proteins are. How cells are developed. Difficulties of the cell line development process and timelines.

2.1.2 Project specifications of cell line development for Merck KgaA

Description of my project, why is it useful, what are the processes here. How my neural network can be used for further stability predictions.

2.2 Deep learning and machine learning basics

Introduction of the notation for the dataset, parameters, predictions.

2.2.1 Neural networks

Convolutional neural network, Autoencoder, embedding, optimizers, regularization, descriptions of how each layer works.

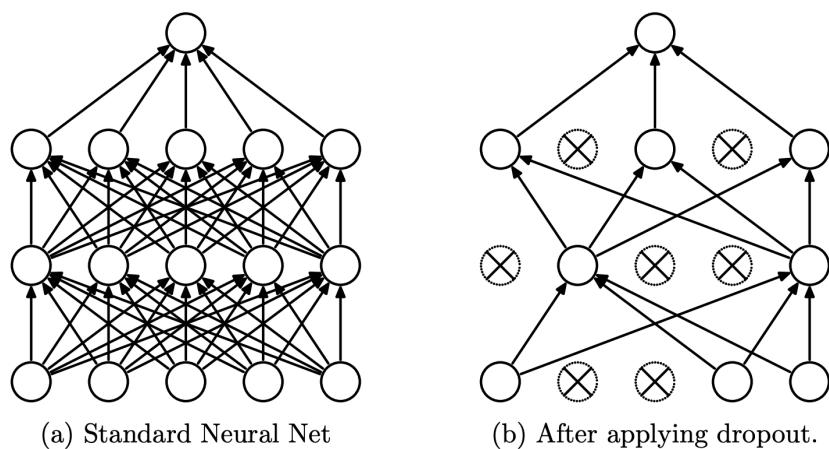


Figure 1: Dropout

2.2.2 Clustering

Theory of clustering algorithms, DBSCAN, HDBSCAN, PCA

2.3 Imaging

2.3.1 Digital imaging

How image is stored in memory, which conventions there are (RGB, BGR (conventions are used in corruptions augmentations)).

2.3.2 Microscopy imaging

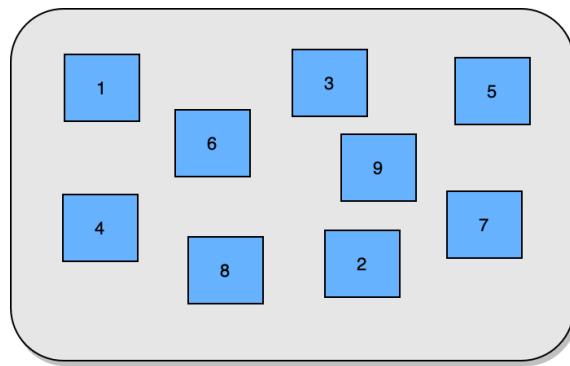


Figure 2: Way in which photos of the well-plate were taken

Which difficulties it may cause (validation loss is lower than train loss)

3 Model training

3.1 Neural network architecture

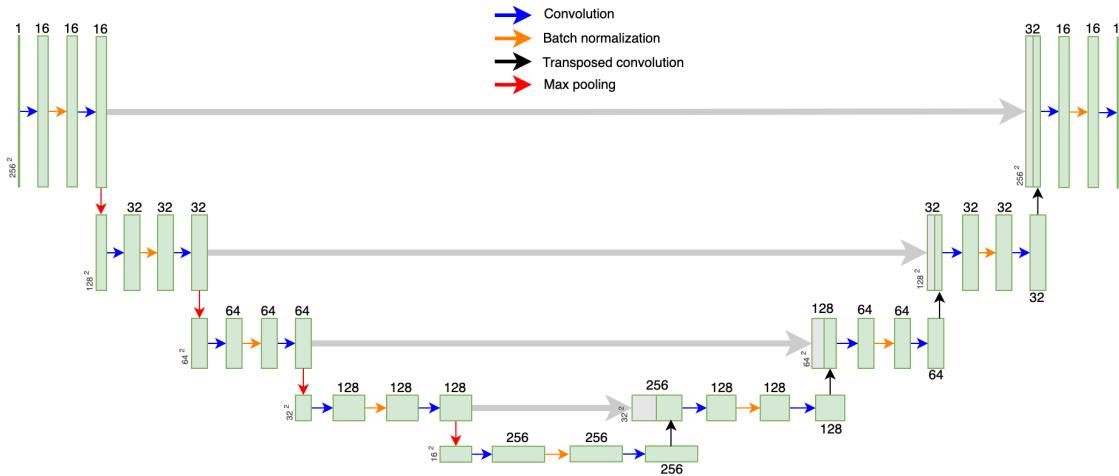


Figure 3: Unet

And information on the embeddings, output sizes, amount of parameters, etc.

3.2 Loss functions

Which loss functions were used, Pearson correlation coefficient explained.

3.3 Available data

Description of the datasets and the amount of images in each category.

3.4 Training costs estimation

Table with the estimation of costs and times for AWS

3.5 Augmentations

Description of all augmentations used

3.5.1 Smart augmentations for rotation and scaling

3.6 Convergence

Images of train and validation loss.

3.7 Model setup

3.7.1 Weight Initialization

Comparison on different weights Initialization. Was the w_i suggested in the paper worth it?

3.7.2 Regularization

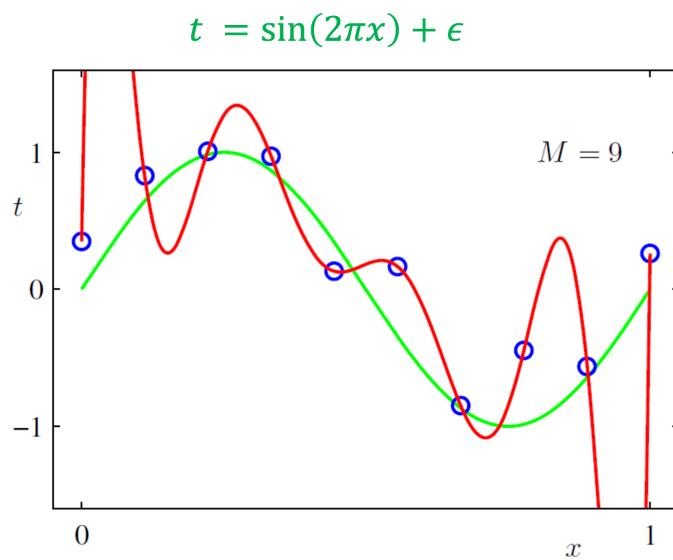


Figure 4: Overfitting

3.7.3 Optimizers

Comparison of different optimizers

4 Nuclei

4.1 Preprocessing

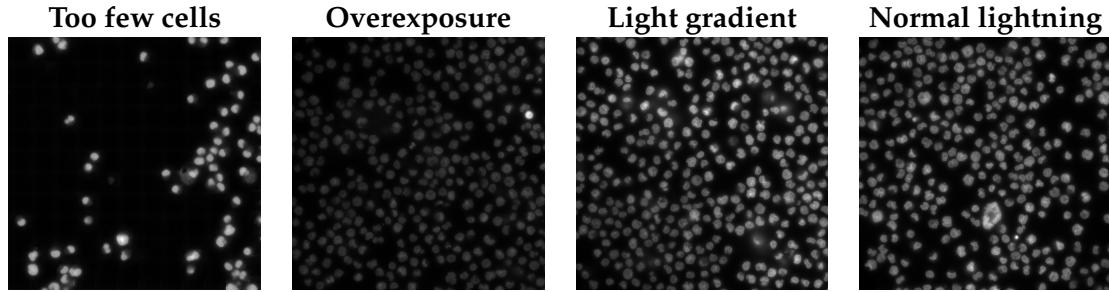


Figure 5: Different lightning conditions

4.1.1 Thresholding algorithms

Global and local thresholding

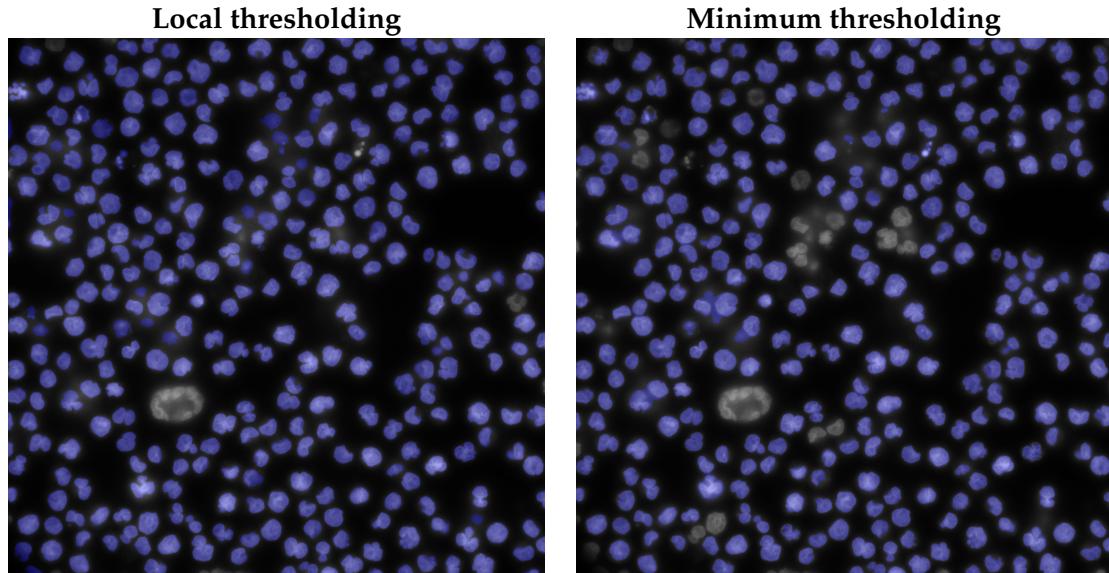


Figure 7: Local vs. Global thresholding (normal conditions)

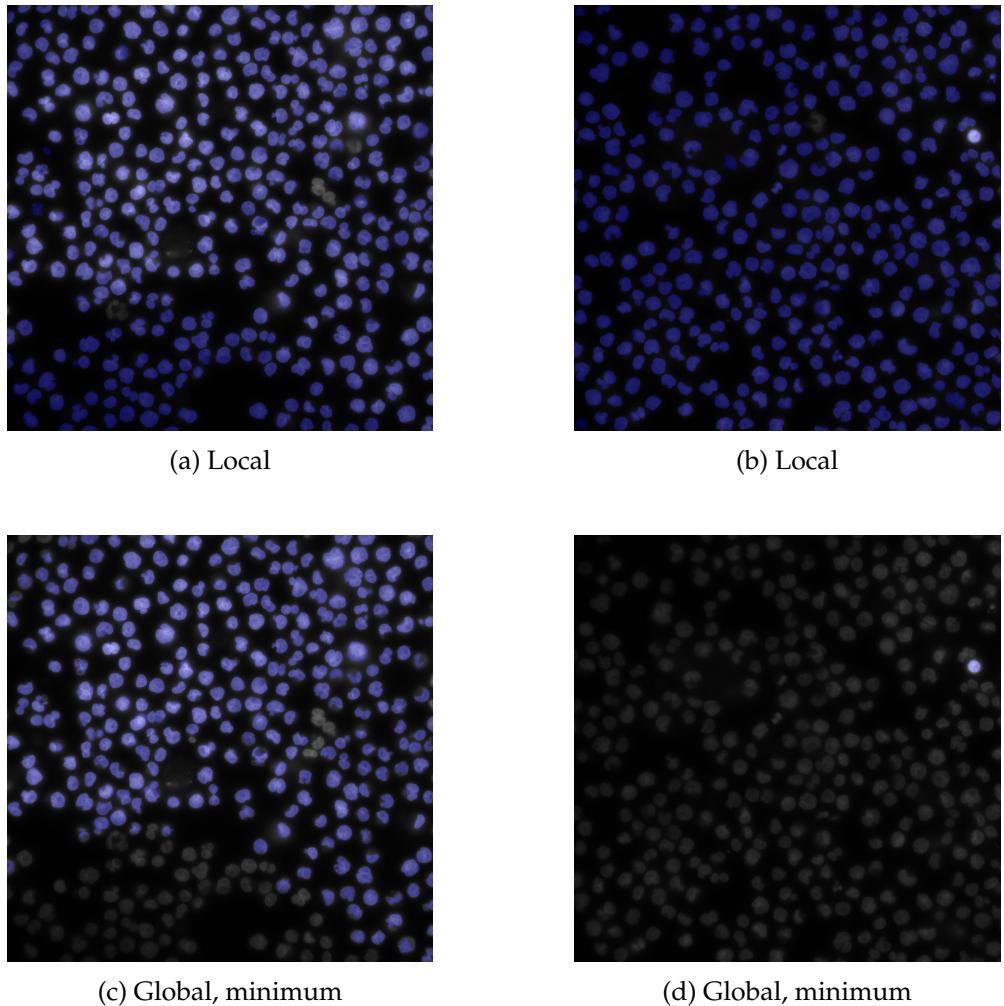


Figure 6: Local vs. Global thresholding

4.2 Training and predictions

4.2.1 Convergence

Has the model converged or not. Will more data help?

4.2.2 Predictions quality

Blurry, boundaries, not enough of details and possible improvements

4.3 Postprocessing for nuclei segmentation

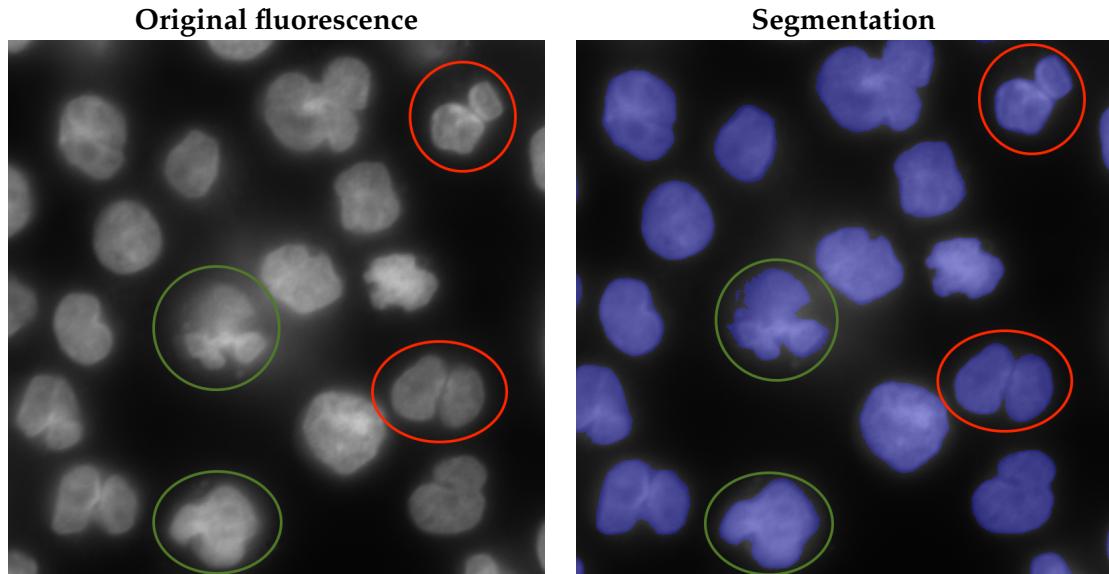


Figure 8: Closely located cells

Overall algorithm

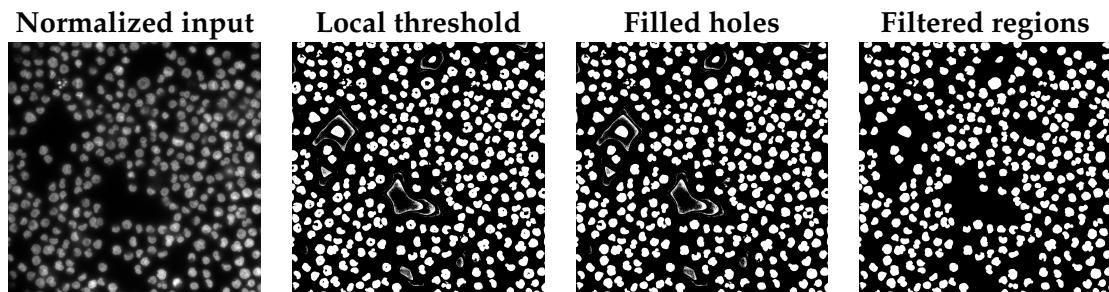


Figure 9: Fluorescence segmentation

4.4 Influence of scaling on predictions quality

Examples of predictions quality with different scales.

5 Actin

5.1 Preprocessing

Algorithm 1 Fluorescence segmentation

1. Normalize image
 2. Apply global *threshold_mean* to receive initial mask.
 3. Zero out pixels outside the mask
 4. Apply local thresholding.
 5. Apply *fill_holes* transformation.
 6. Morphological opening from opencv and Gaussian blur.
 7. Run *findContours* from opencv in order to obtain separate regions and filter out too small regions.
-

Segmentation steps are also illustrated in Figure

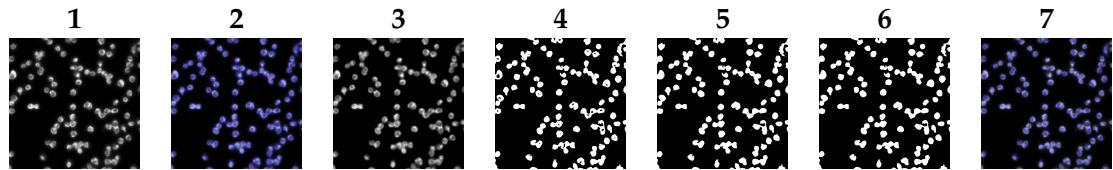


Figure 10: ER prediction

5.2 Training and predictions

5.3 Postprocessing

5.4 Combination of nuclei and actin predictions

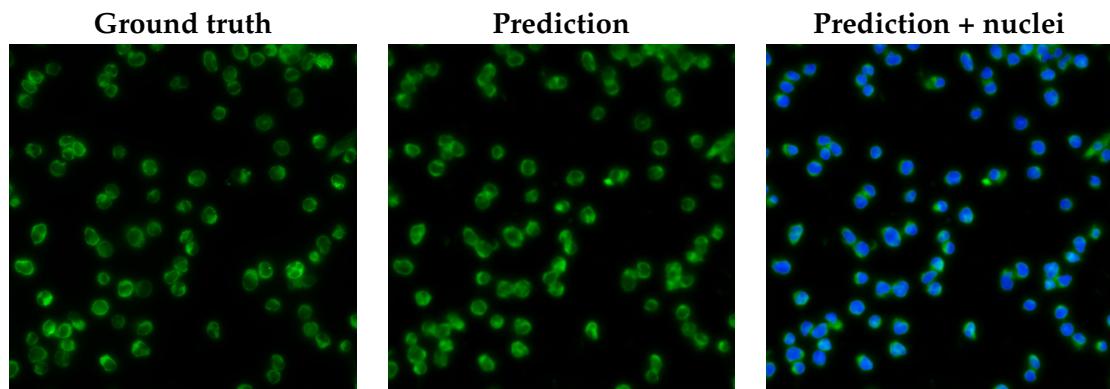


Figure 11: ER prediction

5.5 Prediction quality on the crop's border

5.6 Generalizability across phenotypes

6 Golgi

6.1 Preprocessing

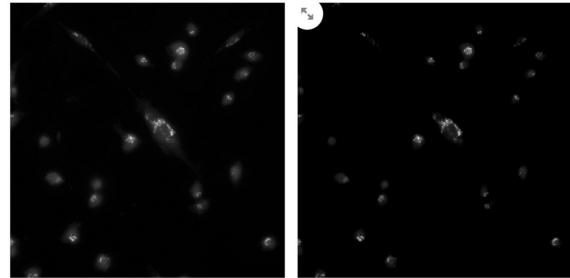


Figure 12: Golgi enhancement

6.1.1 Background removal algorithms

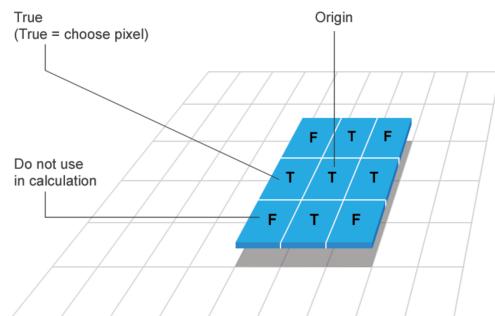


Figure 13: Structuring Element

Rolling ball algorithms

Rolling ball still leaves some noise

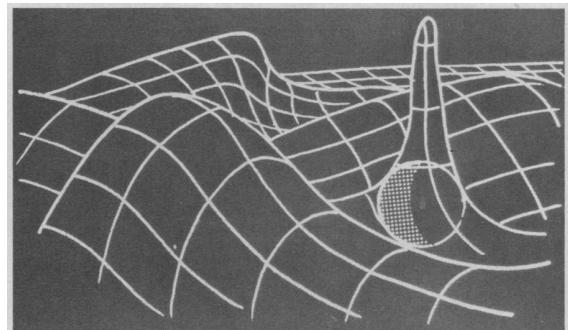


Figure 14: Rolling Ball

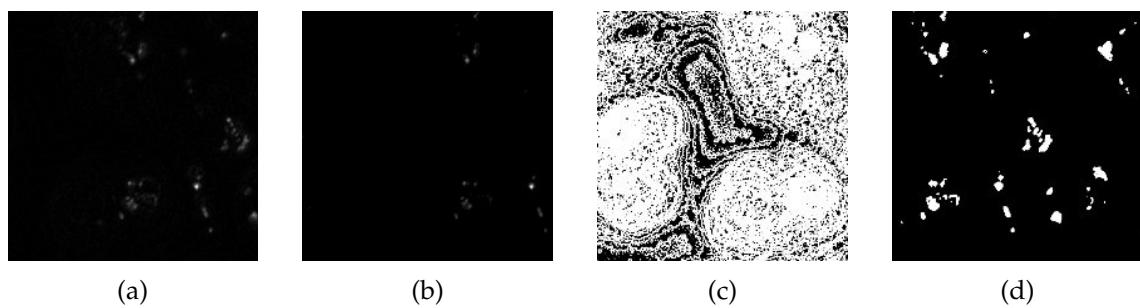


Figure 15: (a) Vanilla pre-processing with automatic background removal algorithm only; (b) Additional clipping of lower intensities after vanilla pre-processing; (c) masked or subfigure (a); (d) mask of subfigure (b)

6.2 Predictions

6.3 Postprocessing

6.4 Alternative ways to improve predictions

6.4.1 Noise reduction methods

6.4.2 Asymmetrical losses

6.4.3 Use of gradient in loss

7 Nucleuolus and full cell prediction

7.1 Preprocessing

7.2 Predictions

7.3 Postprocessing

7.4 Combination of nucleuolus and nuclei

8 Model Evaluation

8.1 Crops combination technique

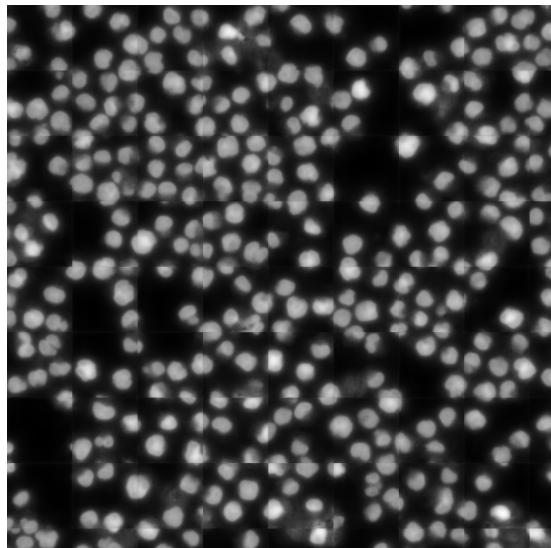


Figure 16: No overlap

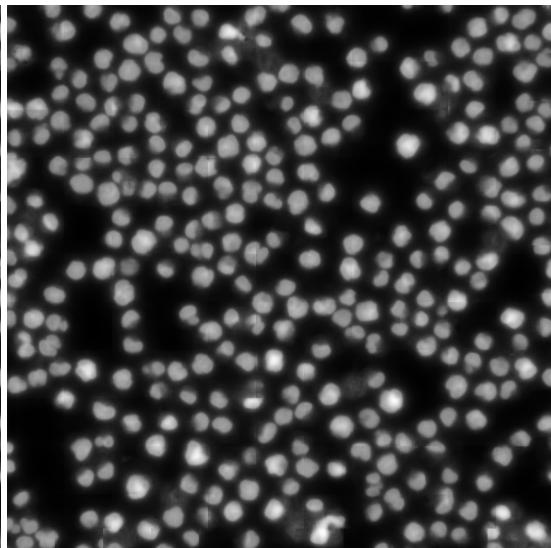


Figure 17: 30 pixels overlap

Improve this plot by showing the visible border explicitly, example of how it can influence a further segmentation perhaps?

8.2 Metrics for downstream tasks

Which downstream tasks are there and how to evaluate them? All violin plots go here

8.3 Influence of different loss functions on metrics for downstream tasks

9 Stability

9.1 Artificial corruptions

Description of artificial corruptions.

Table 1: Hyperparameter for different artificial corruption severities

Severity Corruption \	-5	-4	-3	-2	-1	0	1	2	3	4	5
Defocus blur (radius)						0	0.5	1.0	1.5	2	3
Contrast (gain)	3.5	3.0	2.5	2.0	1.5	1	0.9	0.8	0.7	0.5	0.3
Brightness (bias)	-150	-135	-120	-90	-50	0	50	90	120	135	150

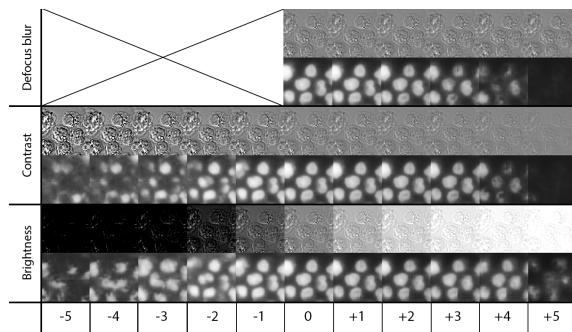


Figure 18: Influence of artificial corruptions on the predictions

9.2 Real corruptions

9.3 Influence of corruptions on metrics for downstream tasks

9.4 Improving predictions with additional corruption augmentations

10 Information in the UNET embeddings

10.1 Dimensionality reduction methods

10.2 UMAP, t-SNE, PCA

10.3 Autoencoder embeddings as an alternative

TODO add the second training loss for the second autoencoder

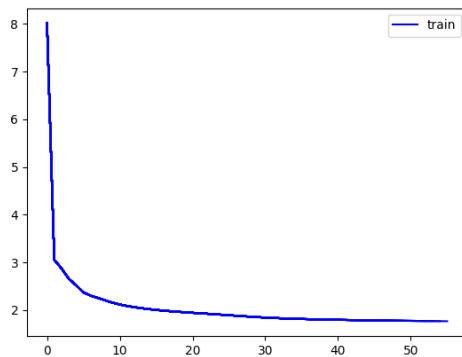


Figure 19: Autoencoder training convergence

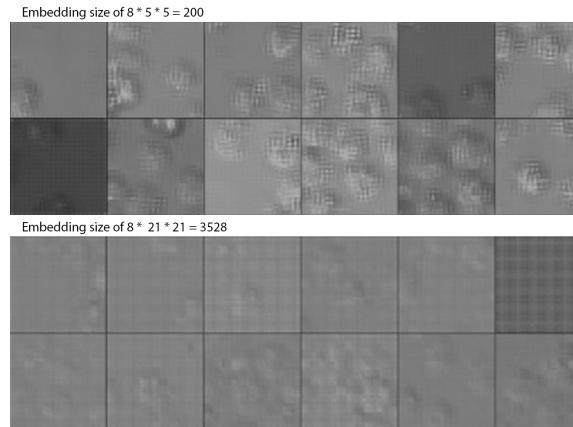


Figure 20: Samples drawn from the trained autoencoder

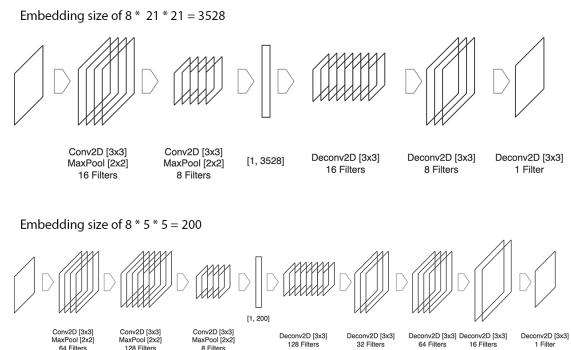


Figure 21: Architectures of two autoencoders

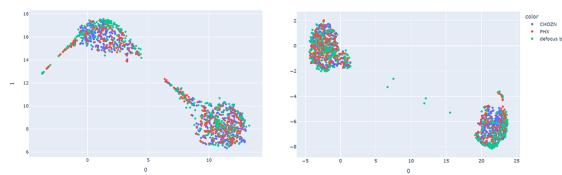


Figure 22: Autoencoder embeddings after applying PCA with 10 components and UMAP afterwards. Earlier epoch VS later epoch

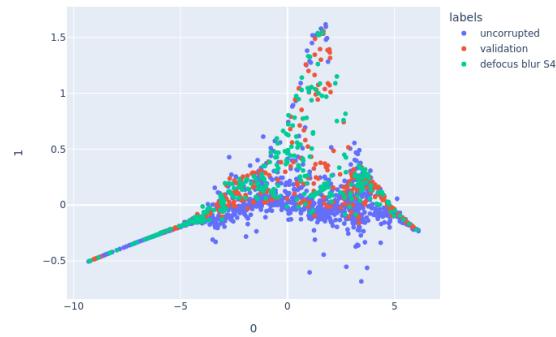


Figure 23: PacMAP does not provide information on the corruption

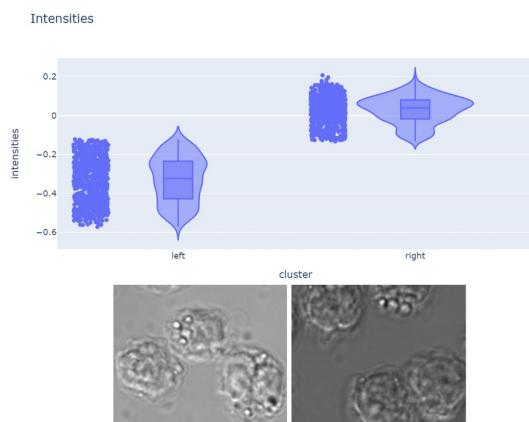


Figure 24: What do two UMAP clusters represent

10.4 Clustering of corrupted samples

11 Drift detection

11.1 A need to detect drift

11.2 MMD

11.2.1 Offline drift detection

11.2.2 Online drift detection

11.3 MMD on UNet embeddings

11.4 Clustering on UNet embeddings

11.5 Clustering on autoencoder embeddings

12 Software Tools

12.1 Foundry, Palantir

12.2 AWS

12.3 Streamlit

12.4 ImageJ, CellProfiler

13 Summary

List of Figures

1	Dropout	3
2	Way in which photos of the well-plate were taken	4
3	Unet	5
4	Overfitting	6
5	Different lightning conditions	8
7	Local vs. Global thresholding (normal conditions)	8
6	Local vs. Global thresholding	9
8	Closely located cells	10
9	Fluorescence segmentation	10
10	ER prediction	11
11	ER prediction	12
12	Golgi enhancement	13
13	Structuring Element	13
14	Rolling Ball	14
15	(a) Vanilla pre-processing with automatic background removal algorithm only; (b) Additional clipping of lower intensities after vanilla pre-processing; (c) masked or subfigure (a); (d) mask of subfigure (b)	14
16	No overlap	16
17	30 pixels overlap	16
18	Influence of artificial corruptions on the predictions	17
19	Autoencoder training convergence	18
20	Samples drawn from the trained autoencoder	18
21	Architectures of two autoencoders	19
22	Autoencoder embeddings after applying PCA with 10 components and UMAP afterwards. Earlier epoch VS later epoch	19
23	PacMAP does not provide information on the corruption	19
24	What do two UMAP clusters represent	20

List of Tables

1	Hyperparameter for different artificial corruption severities	17
---	---	----