

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	6
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	11
4.3	Postprocessing for nuclei segmentation	12
4.4	Influence of scaling on predictions quality	12
5	ER	13
5.1	Preprocessing	13
5.2	Training and predictions	13
5.3	Postprocessing	14
5.4	Combination of nuclei and actin predictions	14
5.5	Prediction quality on the crop's border	14
5.6	Generalizability across phenotypes	14
6	Golgi	15
6.1	Preprocessing	15
6.1.1	Background removal algorithms	15
6.2	Training and predictions	16
6.3	Alternative ways to improve predictions	18
6.3.1	Noise reduction methods	18
6.3.2	Asymmetrical losses	18
6.3.3	Use of gradient in loss	18
7	GFP prediction	19
7.1	Preprocessing	19
7.2	Predictions	19
7.3	Postprocessing	20
7.4	Combination of GFP, nuclei and ER	20
8	Model Evaluation	21
8.1	Crops combination technique	21
8.2	Metrics for downstream tasks	22
8.3	Influence of different loss functions on metrics for downstream tasks	22
9	Stability	23
9.1	Artificial corruptions	23

CONTENTS	iii
9.2 Real corruptions	24
9.3 Influence of corruptions on metrics for downstream tasks	24
9.4 Improving predictions with additional corruption augmentations	24
10 Information in the UNET embeddings	25
10.1 Dimensionality reduction methods	25
10.2 UMAP, t-SNE, PCA	25
10.3 Autoencoder embeddings as an alternative	25
10.4 Clustering	27
10.4.1 Clustering methods (HDBSCAN, DBSCAN, K-means)	27
10.4.2 Clustering on UNet embeddings	27
11 Drift detection	28
11.1 A need to detect drift	28
11.2 MMD	28
11.2.1 Drift detection	28
11.2.2 Online drift detection	28
12 Software Tools	30
12.1 Foundry. Palantir	30
12.2 AWS	30
12.3 Streamlit	30
12.4 ImageJ, CellProfiler	30
13 Summary	31
List of Figures	32
List of Tables	33

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.

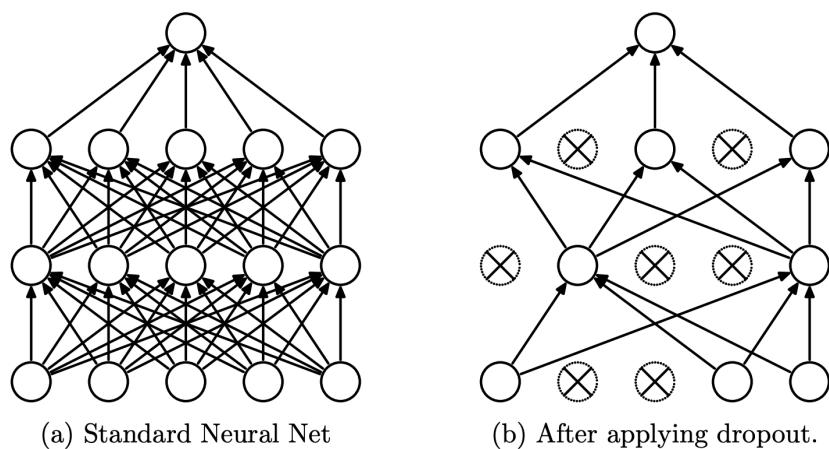


Figure 1: Dropout

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

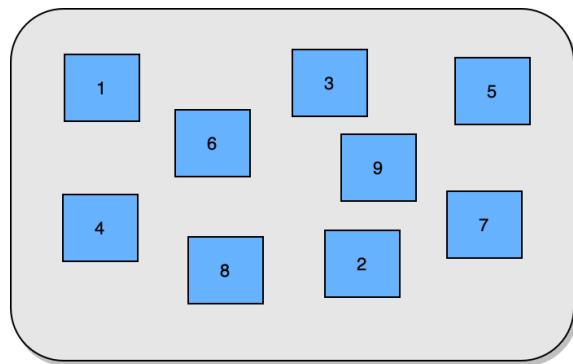


Figure 2: Way in which photos of the well-plate were taken

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

3 Model training

3.1 Neural network architecture

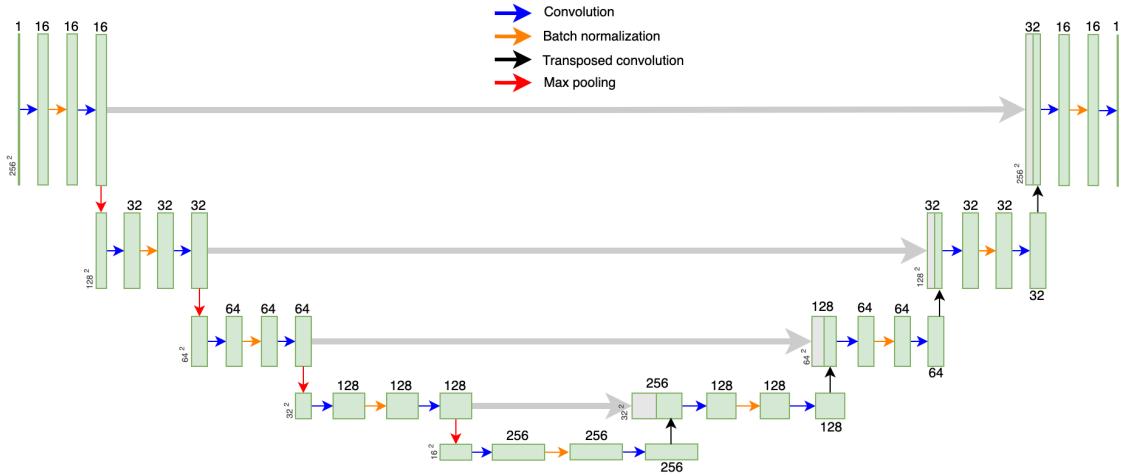


Figure 3: Unet

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.

Table 1: Available data for each fo the organelles

	Total images	Training crops	Validation crops	Test crops
Nuclei	595	27,264	3,008	7,616
Actin	400	18,432	2,048	5120
Golgi	761	23,036	2,336	6,347
H19	400	?	?	?
Nucleolei	?	?	?	?

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

These plots represent MSE

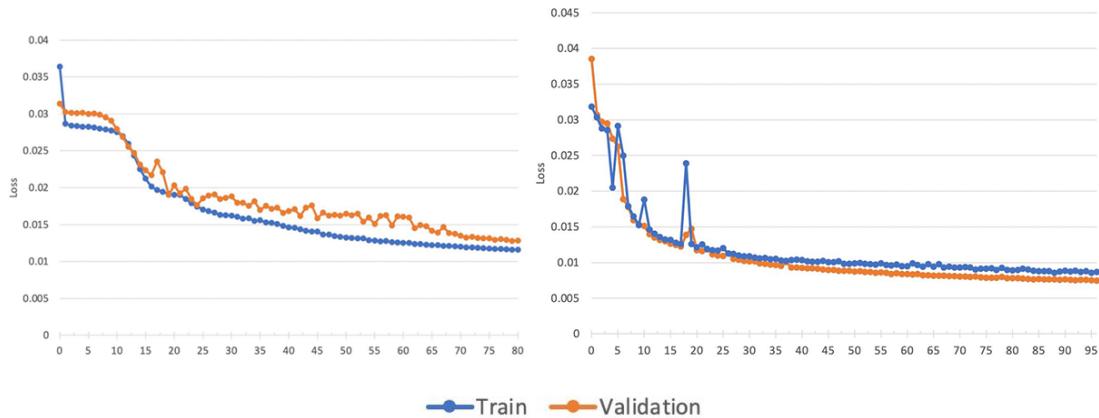


Figure 4: Nuclei training without and with custom weight initialization

3.7.2 Regularization

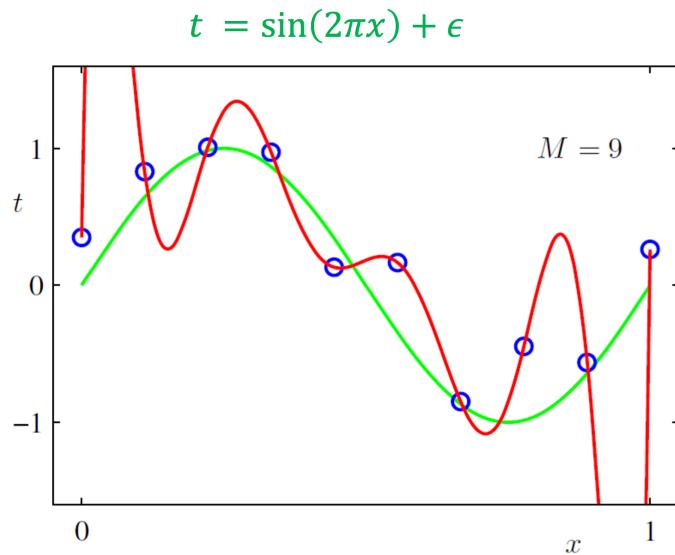


Figure 5: Overfitting

3.7.3 Optimizers

Comparison of different optimizers

4 Nuclei

4.1 Preprocessing

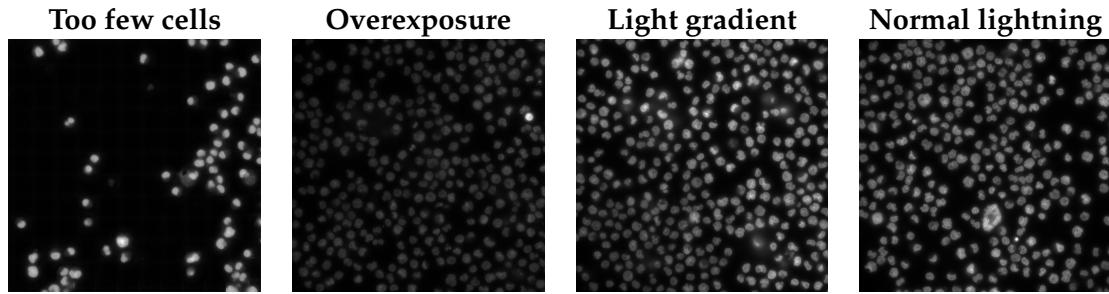


Figure 6: Different lightning conditions

4.1.1 Thresholding algorithms

Global and local thresholding

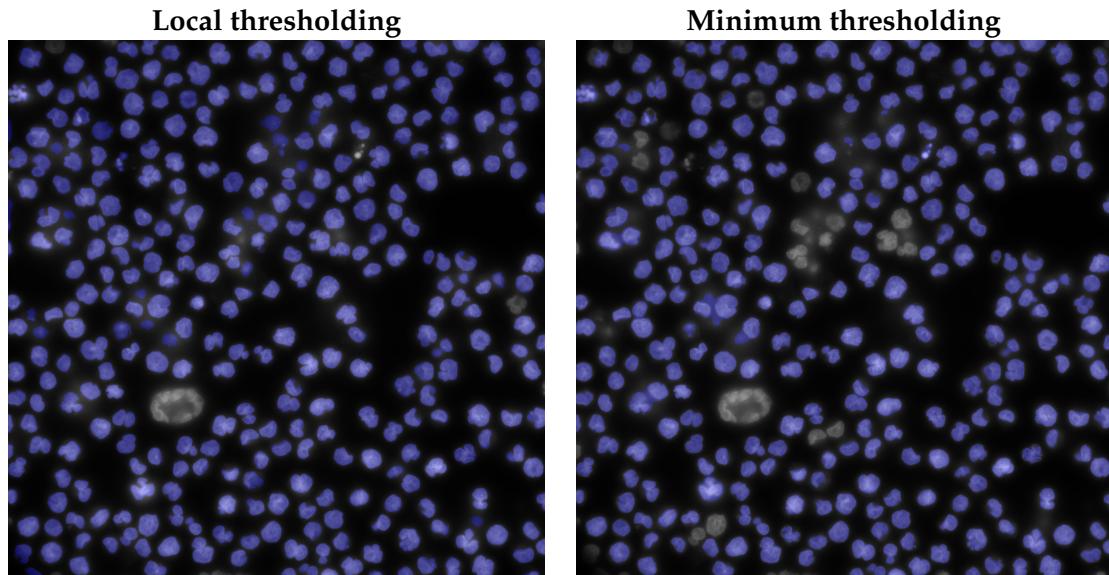


Figure 8: Local vs. Global thresholding (normal conditions)

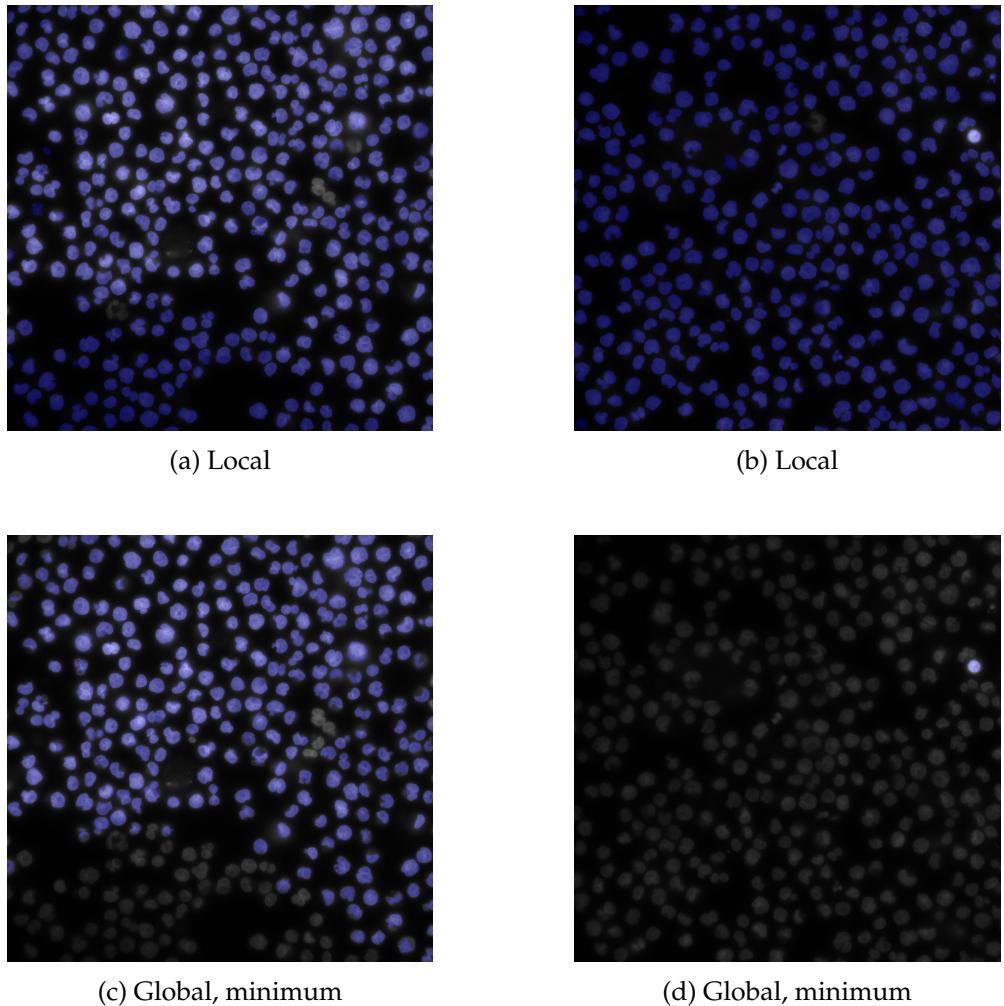


Figure 7: Local vs. Global thresholding

4.2 Training and predictions

4.2.1 Convergence

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

Not regularized yet, full dataset training with PCC

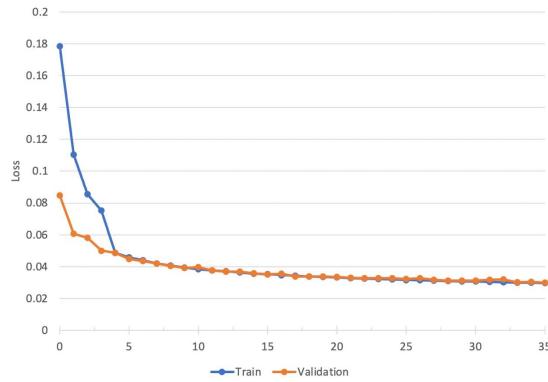


Figure 9: Having more data makes training more stable

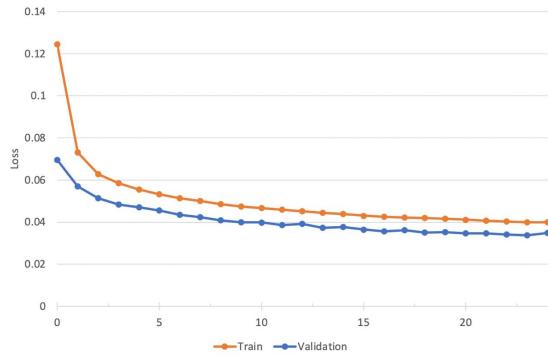


Figure 10: With regularization and augmentations PCC

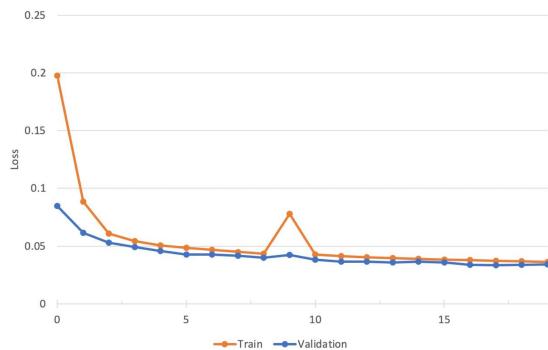


Figure 11: No regularization but augmentations

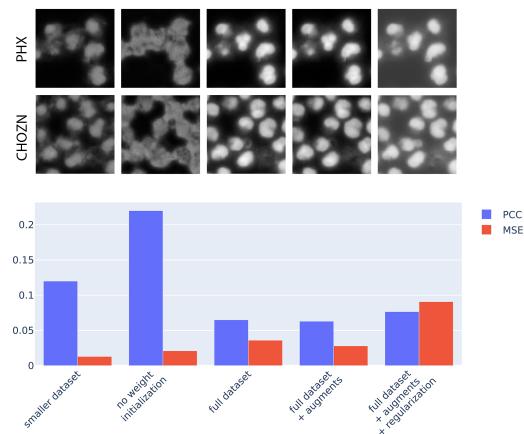


Figure 12: Difefrent models predictions and scores comparison

4.2.2 Predictions quality

Blurry, boundaries, not enough of details and possible improvements

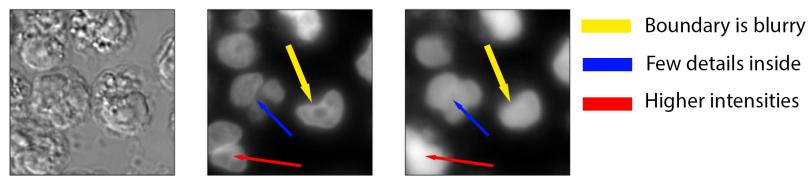


Figure 13: Some troubles in predictions

4.3 Postprocessing for nuclei segmentation

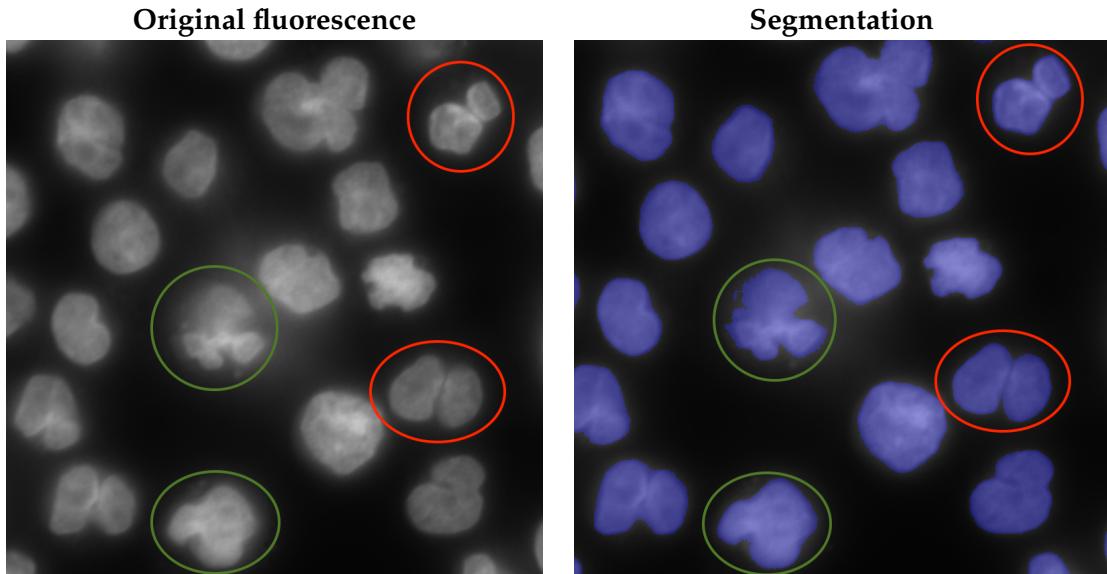


Figure 14: Closely located cells

Overall algorithm

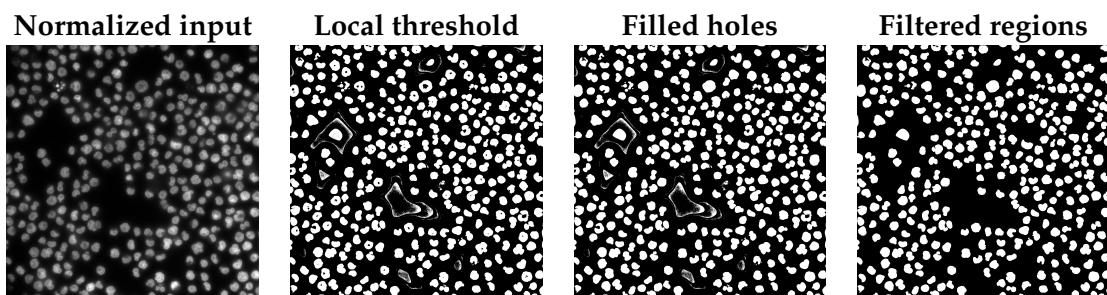


Figure 15: Fluorescence segmentation

4.4 Influence of scaling on predictions quality

Examples of predictions quality with different scales.

5 ER

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

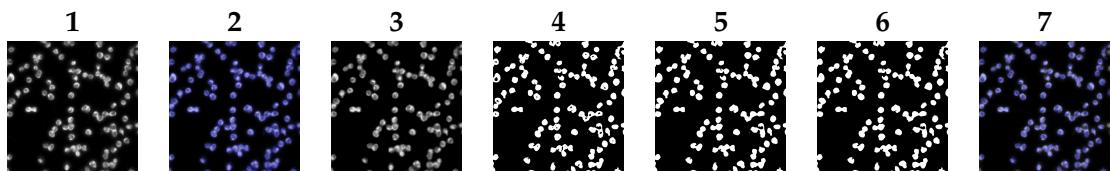


Figure 16: ER prediction

5.2 Training and predictions

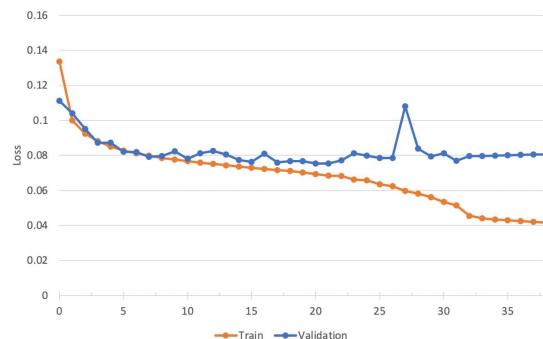


Figure 17: Overfit

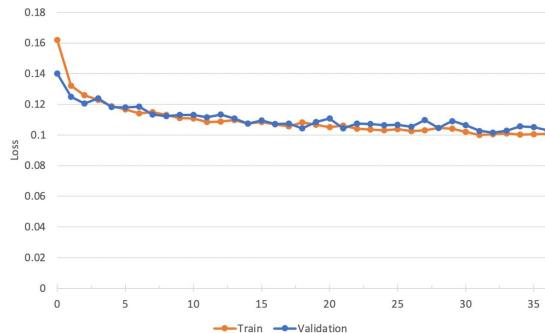


Figure 18: No overfit with augmentations

5.3 Postprocessing

5.4 Combination of nuclei and actin predictions

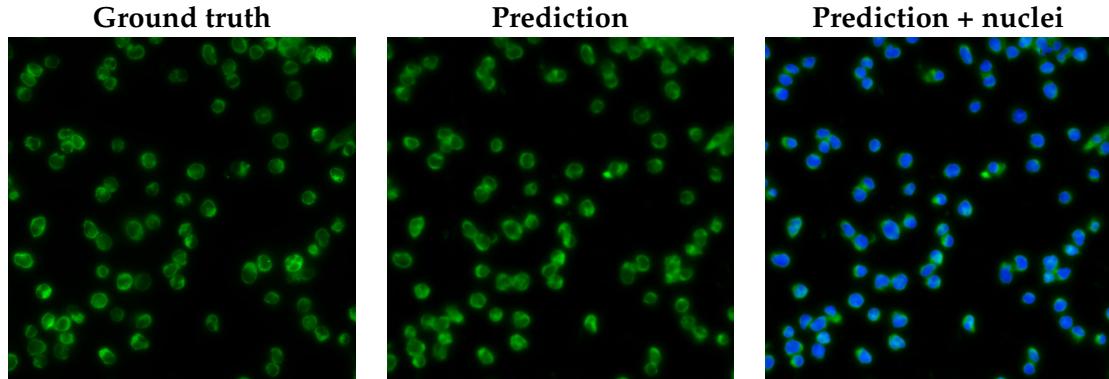


Figure 19: ER prediction

5.5 Prediction quality on the crop's border

5.6 Generalizability across phenotypes

6 Golgi

6.1 Preprocessing

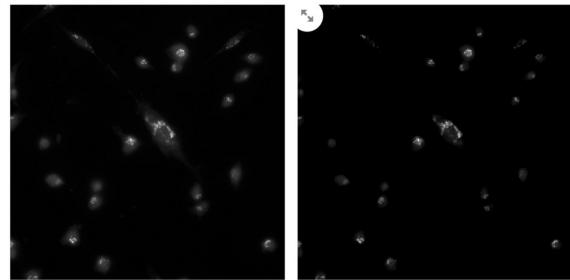


Figure 20: Golgi enhancement

6.1.1 Background removal algorithms

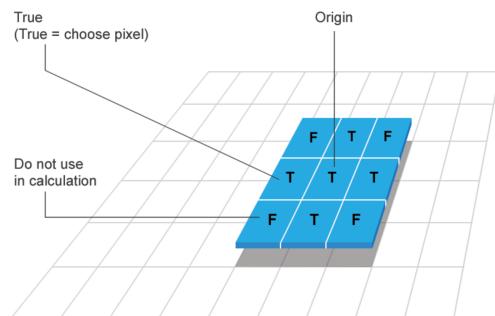


Figure 21: Structuring Element

Rolling ball algorithms

Rolling ball still leaves some noise

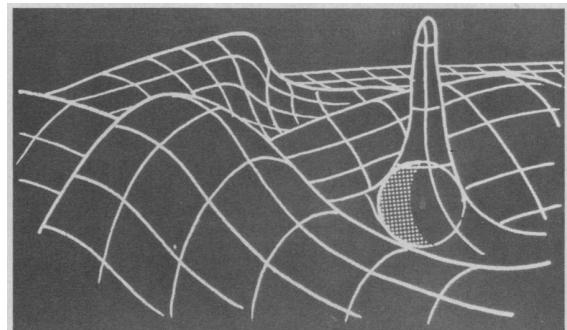


Figure 22: Rolling Ball

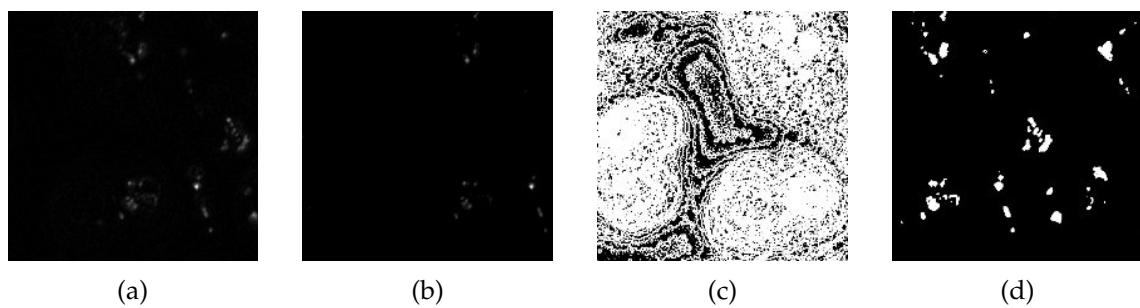


Figure 23: (a) Vanilla pre-processing with automatic background removal algorithm only; (b) Additional clipping of lower intensities after vanilla pre-processing; (c) masked or subfigure (a); (d) mask of subfigure (b)

6.2 Training and predictions

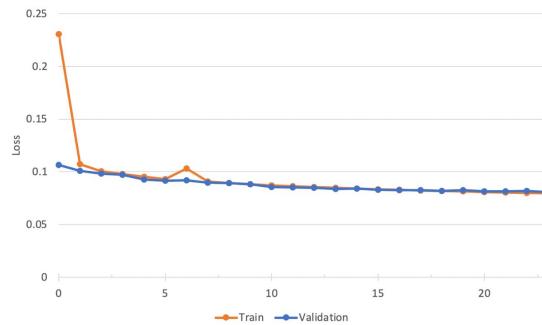


Figure 24: Straightforward training doesn't work

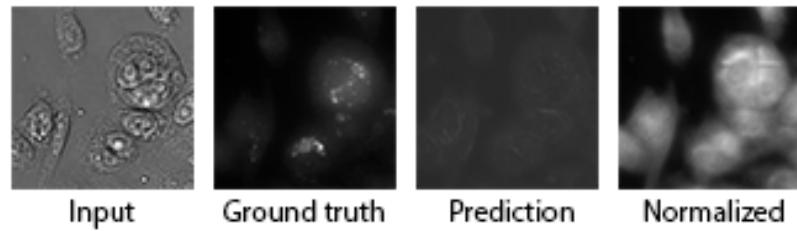


Figure 25: Training on original data

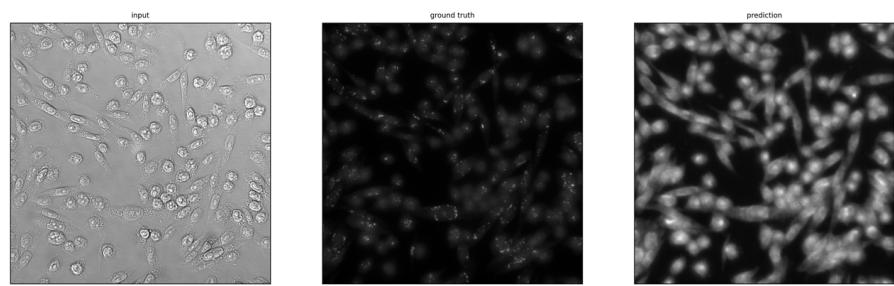


Figure 26: Full size predictions

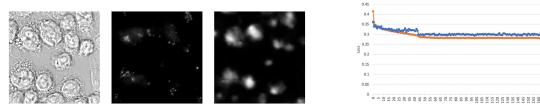


Figure 27: Training on the enhanced data

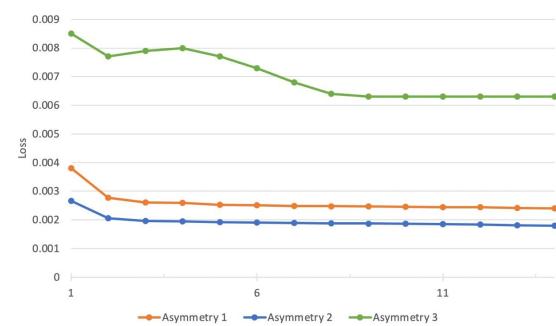


Figure 28: Asymmetrical training

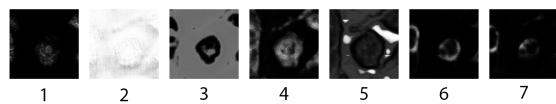


Figure 29: Asymmetrical training predictions

6.3 Alternative ways to improve predictions

6.3.1 Noise reduction methods

6.3.2 Asymmetrical losses

6.3.3 Use of gradient in loss

7 GFP prediction

7.1 Preprocessing

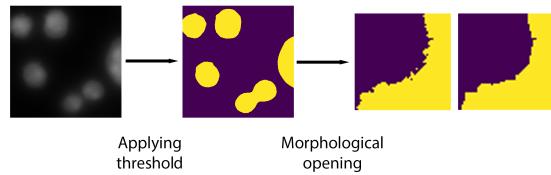


Figure 30: Converting gfp to a binary mask

7.2 Predictions

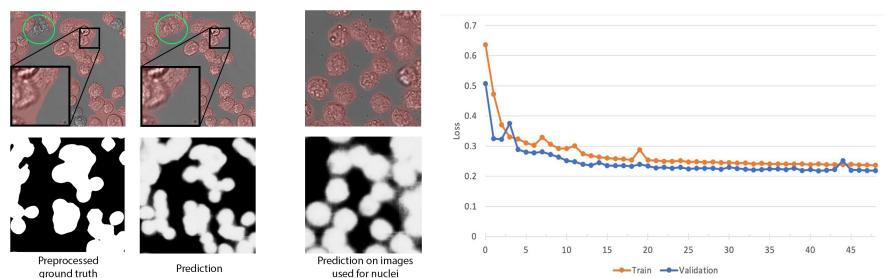


Figure 31: Binary training with BCE

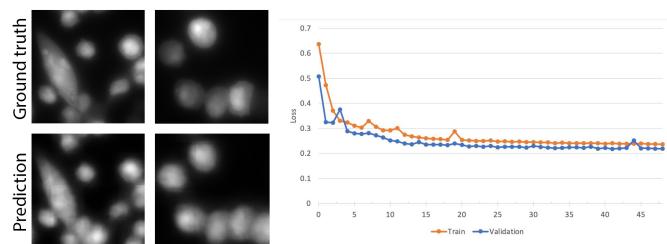


Figure 32: Training with Pearson correlation loss

Table 2: Correlation coefficients for downstream tasks

Binary training	Pearson	Spearman
Number of ER	0.67	0.64
Area	0.82	0.75
Continuos training	Pearson	Spearman
Number of ER	0.57	0.55
Area	0.26	0.64

7.3 Postprocessing

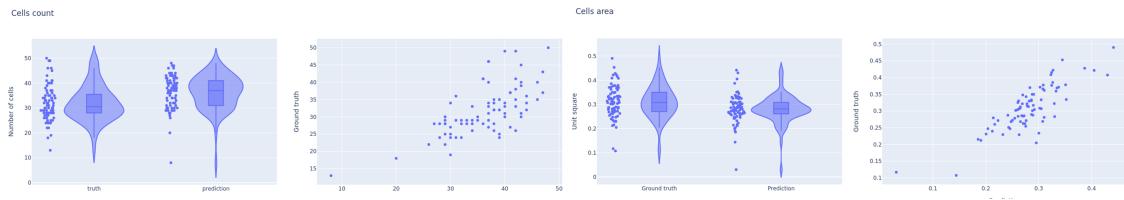


Figure 33: Downstream metrics

7.4 Combination of GFP, nuclei and ER

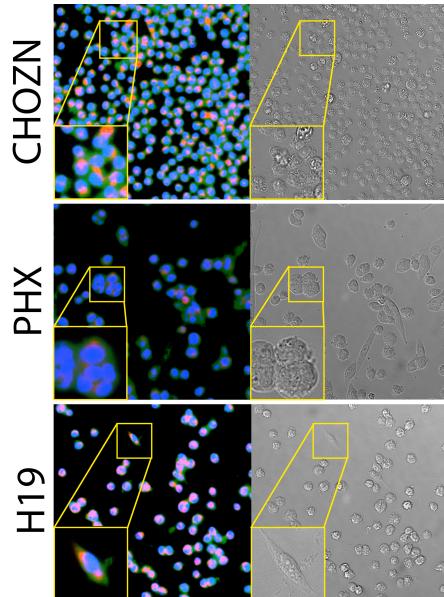


Figure 34: GFP, Nuclei and ER combined

8 Model Evaluation

8.1 Crops combination technique

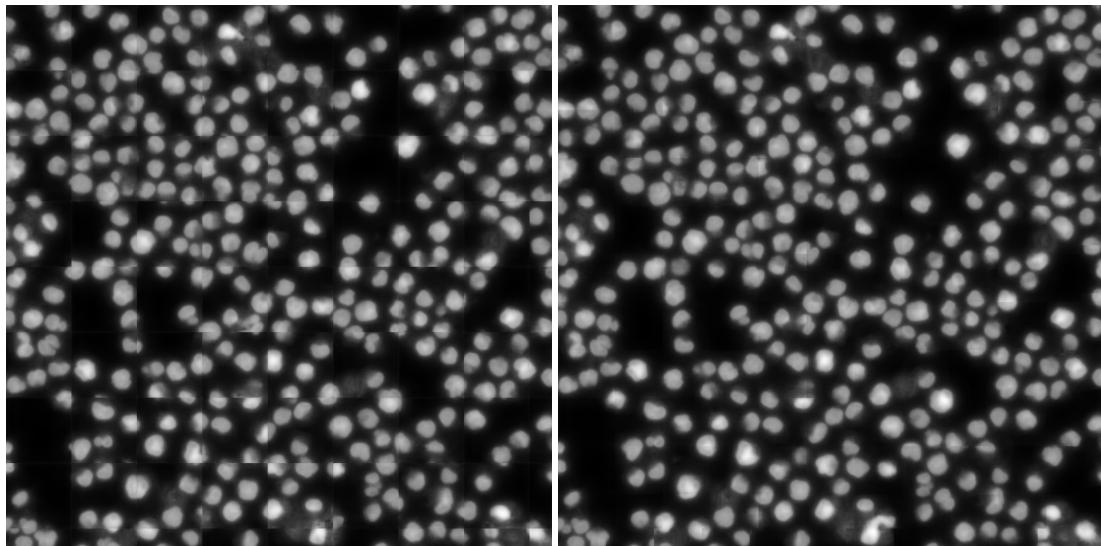


Figure 35: No overlap

Figure 36: 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

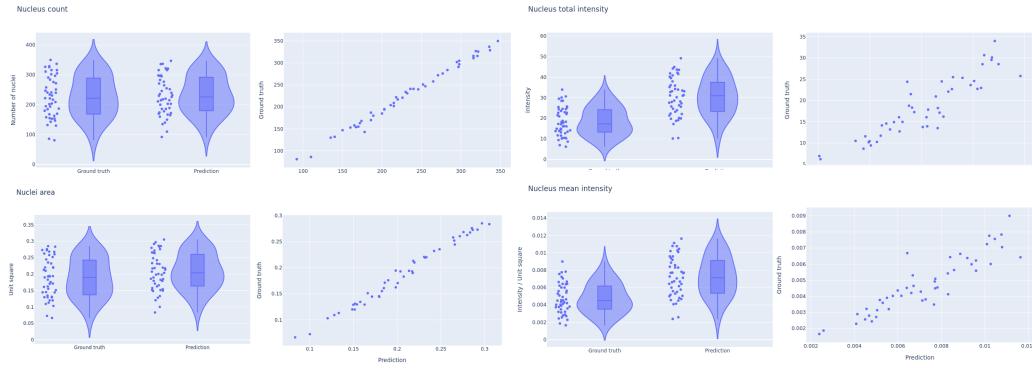


Figure 37: Metrics for downstream tasks on nuclei

Table 3: Correlation coefficients for downstream tasks

	Pearson	Spearman
Number of nuclei	0.995	0.994
Total intensity	0.902	.911
Mean intensity	0.907	0.904
Area	0.992	0.990

8.3 Influence of different loss functions on metrics for downstream tasks

9 Stability

9.1 Artificial corruptions

Description of artificial corruptions.

Table 4: 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

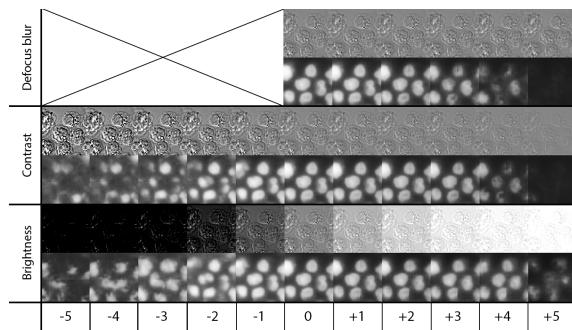


Figure 38: Influence of artificial corruptions on the predictions

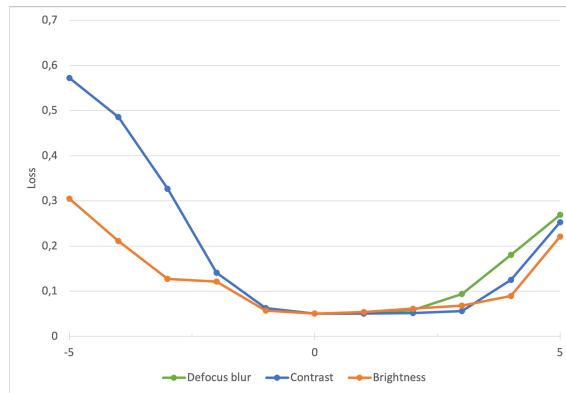


Figure 39: Change of loss for artificial corruptions

9.2 Real corruptions

9.3 Influence of corruptions on metrics for downstream tasks

9.4 Improving predictions with additional corruption augmentations

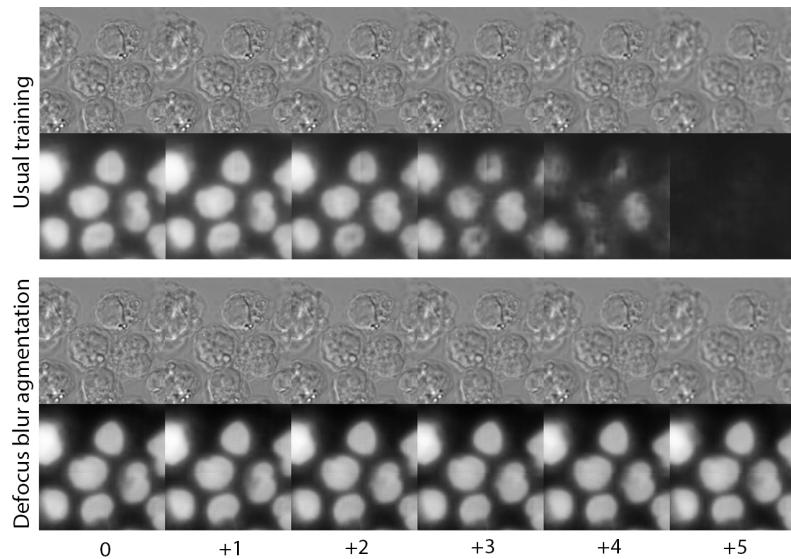


Figure 40: Using corruptions as augmentations improves predictions

10 Information in the UNET embeddings

10.1 Dimensionality reduction methods

10.2 UMAP, t-SNE, PCA

10.3 Autoencoder embeddings as an alternative

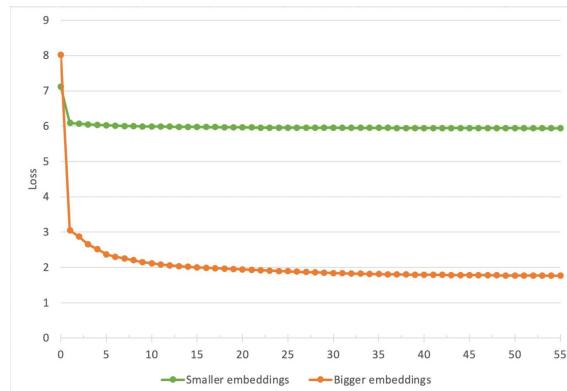


Figure 41: Autoencoders training convergence

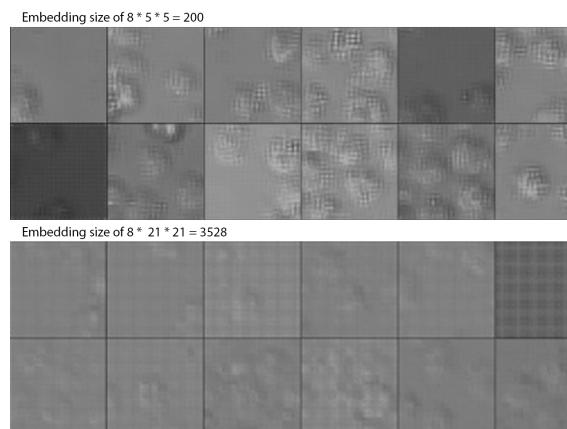


Figure 42: Samples drawn from the trained autoencoder

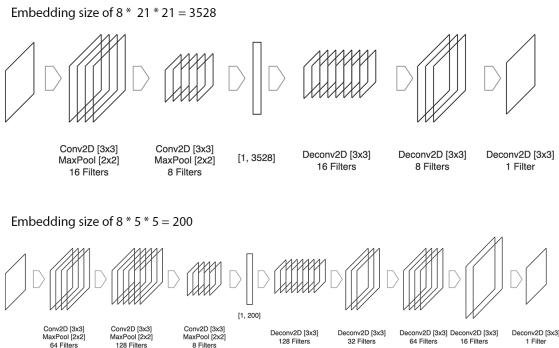


Figure 43: Architectures of two autoencoders

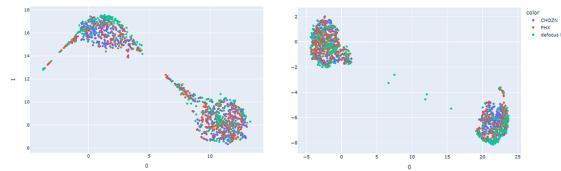


Figure 44: Autoencoder embeddings after applying PCA with 10 components and UMAP afterwards. Earlier epoch VS later epoch

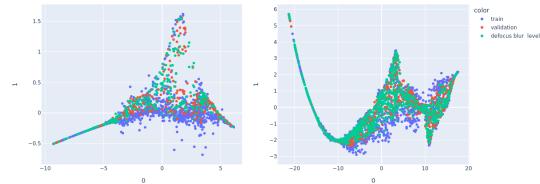


Figure 45: PacMAP does not provide information on the corruption

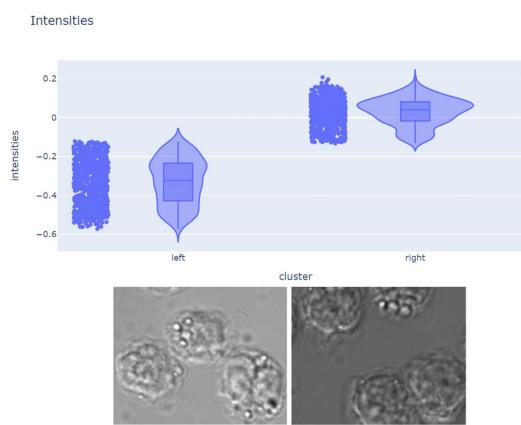


Figure 46: What do two UMAP clusters represent

10.4 Clustering

10.4.1 Clustering methods (HDBSCAN, DBSCAN, K-means)

10.4.2 Clustering on UNet embeddings

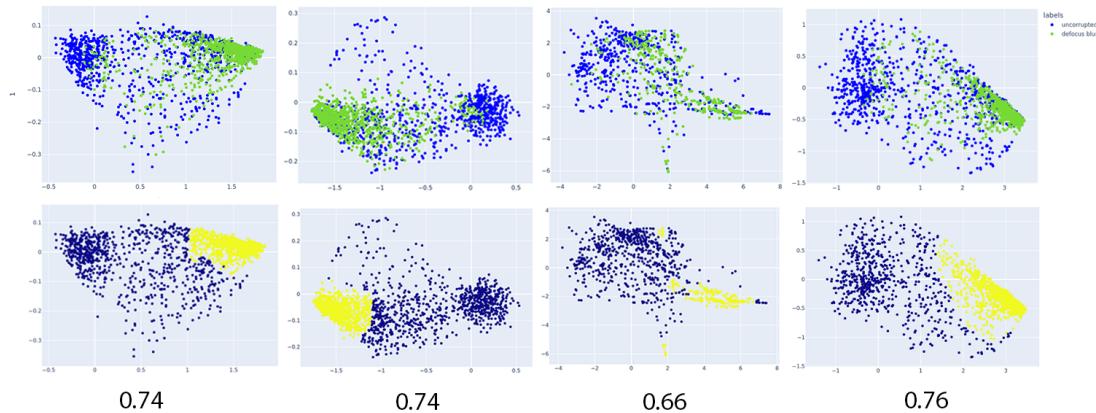


Figure 47: Clustering of UNet Embeddings

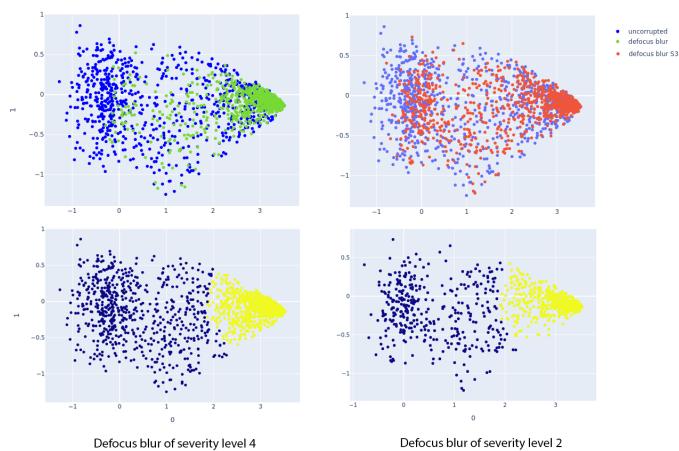


Figure 48: Clustering for different severities levels

TABLE with F1-score: 0.76 VS 0.64

11 Drift detection

11.1 A need to detect drift

11.2 MMD

11.2.1 Drift detection

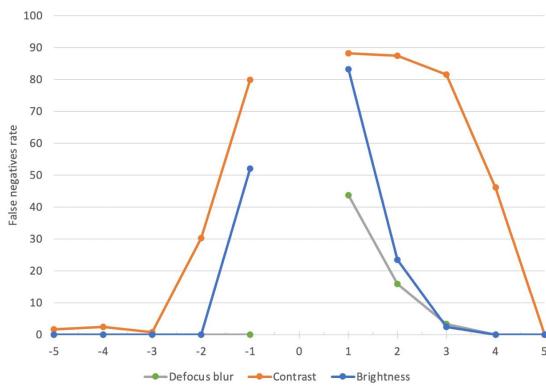


Figure 49: False negatives rate for drift detection on artificial corruptions

11.2.2 Online drift detection

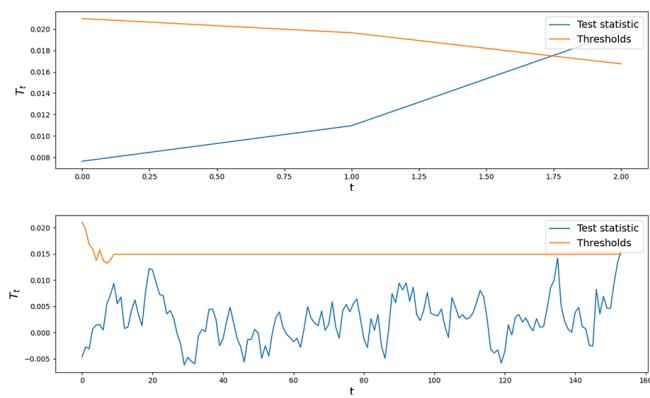


Figure 50: Expected runtime (ERT) for corrupted and in-distribution data

Table 5: Test window size influence on separability

W	2	5	10	15	20
Auc-Roc	0.85	0.92	0.98	0.90	0.88

Table 6: ERT influence on separability

W	32	64	128	256
Auc-Roc	0.90	0.95	0.98	0.98

Table 7: Severity of corruptions on separability

W	Level 2	Level 3	Level 4
Auc-Roc	0.84	0.92	0.98

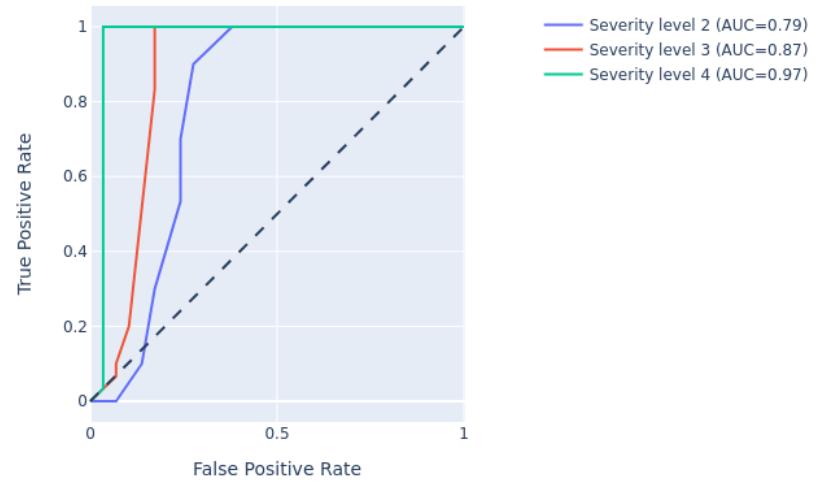


Figure 51: AUC ROC scores for various defocus corruptions severities

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	Nuclei training without and with custom weight initialization	6
5	Overfitting	7
6	Different lightning conditions	8
8	Local vs. Global thresholding (normal conditions)	8
7	Local vs. Global thresholding	9
9	Having more data makes training more stable	10
10	With regularization and augmentations PCC	10
11	No regularization but augmentations	10
12	Difefrent models predictions and scores comparison	11
13	Some troubles in predictions	11
14	Closely located cells	12
15	Fluorescence segmentation	12
16	ER prediction	13
17	Overfit	13
18	No overfit with augmentations	14
19	ER prediction	14
20	Golgi enhancement	15
21	Structuring Element	15
22	Rolling Ball	16
23	(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)	16
24	Straightforward training doesn't work	16
25	Training on original data	17
26	Full size predictions	17
27	Training on the enhanced data	17
28	Asymmetrical training	17
29	Asymmetrical training predictions	18
30	Converting gfp to a binary mask	19

<i>LIST OF TABLES</i>	33
-----------------------	----

31	Binary training with BCE	19
32	Training with Pearson correlation loss	19
33	Downstream metrics	20
34	GFP, Nuclei and ER combined	20
35	No overlap	21
36	30 pixels overlap	21
37	Metrics for downstream tasks on nuclei	22
38	Influence of artificial corruptions on the predictions	23
39	Change of loss for artificial corruptions	23
40	Using corruptions as augmentations improves predictions	24
41	Autoencoders training convergence	25
42	Samples drawn from the trained autoencoder	25
43	Architectures of two autoencoders	26
44	Autoencoder embeddings after applying PCA with 10 components and UMAP afterwards. Earlier epoch VS later epoch	26
45	PacMAP does not provide information on the corruption	26
46	What do two UMAP clusters represent	26
47	Clustering of UNet Embeddings	27
48	Clustering for different severities levels	27
49	False negatives rate for drift detection on artificial corruptions	28
50	Expected runtime (ERT) for corrupted and in-distribution data	28
51	AUC ROC scores for various defocus corruptions severities	29

List of Tables

1	Available data for each fo the organelles	5
2	Correlation coefficients for downstream tasks	20
3	Correlation coefficients for downstream tasks	22
4	Hyperparameter for different artificial corruption severities	23
5	Test window size influence on separability	29
6	ERT influence on separability	29
7	Severity of corruptions on separability	29