QM Course 2024

Charles University, Prague

Machine Learning with napari-plugins

(+ extra flim plugin)

Marcelo Leomil Zoccoler

Re-using materials from:

Robert Haase, ScaDS.AI Leipzig

Johannes Soltwedel, PoL, TU Dresden

Cornelia Wetzker, TU Dresden and NFDI4BIOIMAGE

Till Korten, Helmholtz AI

Ryan Savill, MPI CBG

Laura Zigutyte, EKFZdigital

Mara Lampert, TU Dresden

Deborah Schmidt, Jug Lab, MPI CBG

Uwe Schmidt, Myers Lab, MPI CBG

Martin Weigert, EPFL

Ignacio Arganda-Carreras, Universidad del Pais Vasco

Carsen Stringer, HHMI Janelia

Wei Ouyang, KTH Royal Institute of Technology, Stockholm and

The Scikit-Learn community

@zoccolermarcelo

Before we start…

- Let’s practice some conda environment configuration!
  - more into how environments work:
    - https://jni.github.io/using-python-for-science/intro-to-environments.html
    - https://hackmd.io/@talley/SJB_lObBi#Python-environments-workshop
- We actually need to create 2 environments to run some of the exercises today. Please follow the instructions here:
  - https://zoccoler.github.io/QM_Course_Bio_Image_Analysis_with_napari_2024/Machine_Learning_with_napari/00_intro.html

First environemnt: devbio-napari

Open a terminal and type (or copy the line below):

```
mamba create --name devbio-napari-env python=3.9 devbio-napari pyqt -c conda-forge -c pytorch
```

Hit `ENTER` to answer yes when asked to install packages. Activate your environment with:

```
mamba activate devbio-napari
```

Second environment: napari-flim-phasor-env

Open a terminal and type (or copy the line below):

```
mamba create -n napari-flim-phasor-env python=3.9 napari==0.4.17 napari-clusters-plotter git pyqt devbio-napari
```

Hit `ENTER` to answer yes when asked to install packages.

Activate your environment with:

```
mamba activate napari-flim-phasor-env
```

Then install napari-flim-phasor-plotter with pip:

```
pip install napari-flim-phasor-plotter
```

Image segmentation using thresholding

- Recap: Finding the right workflow towards a good segmentation takes time

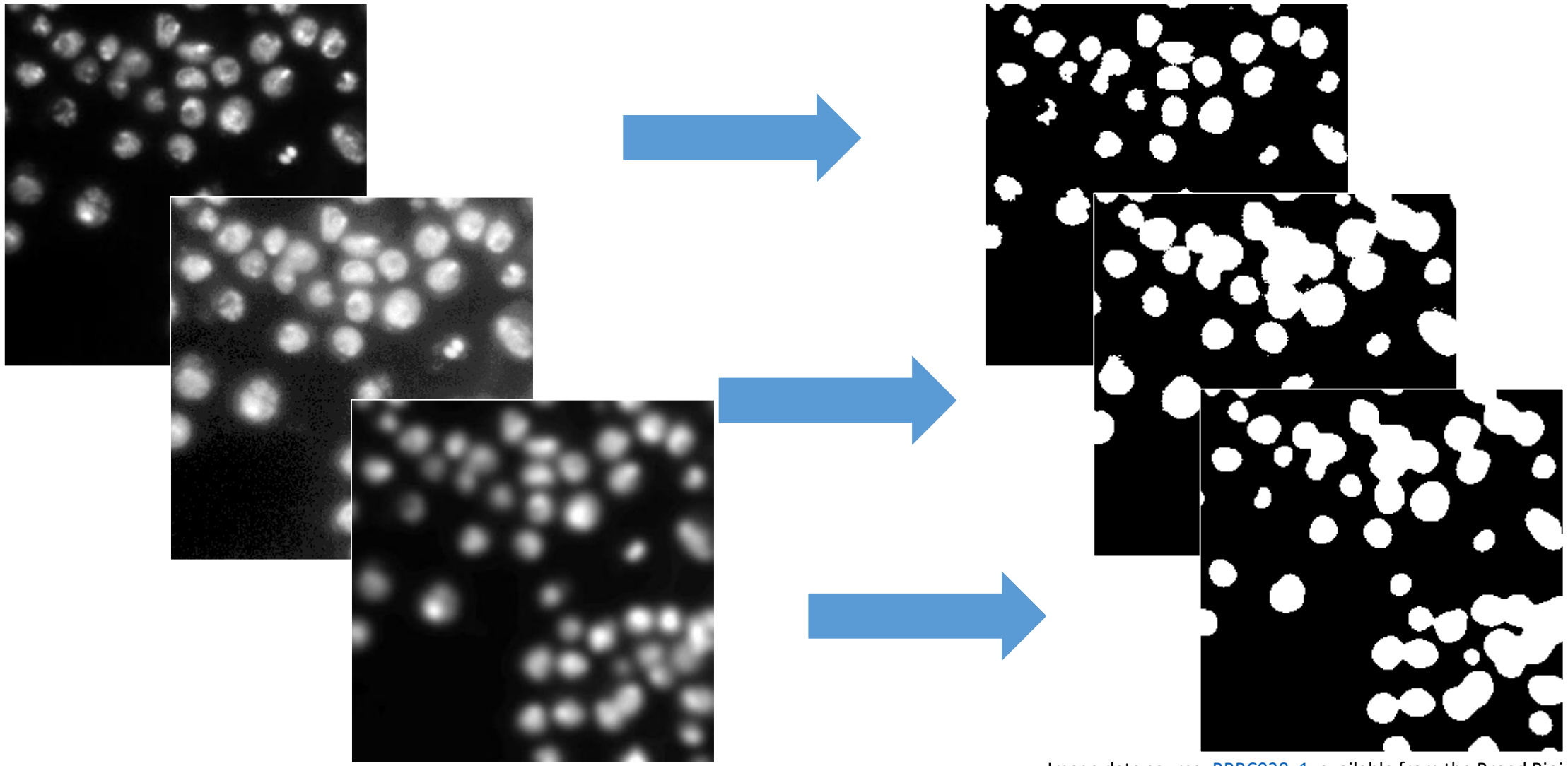
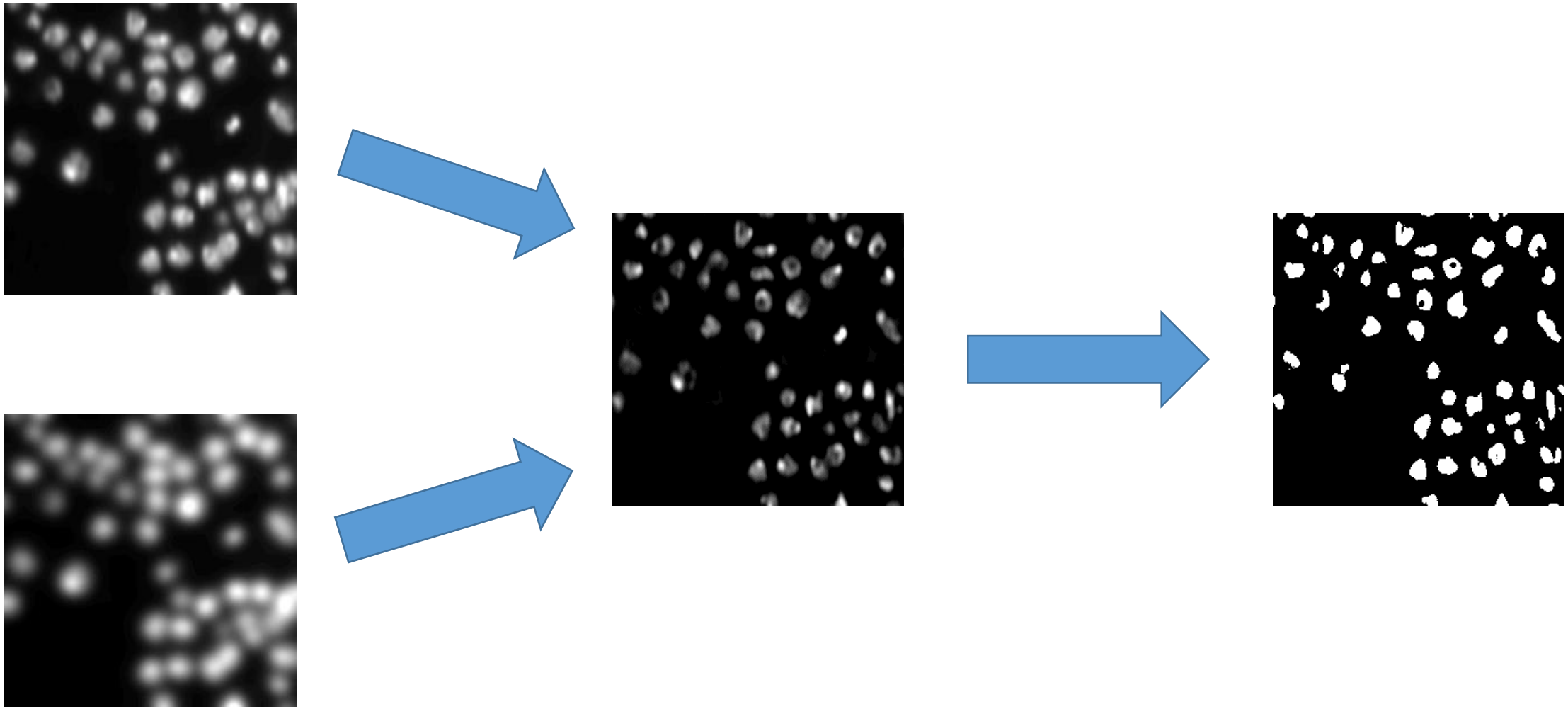


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

Image segmentation using thresholding

- Recap: Combining images, e.g. using Difference of Gaussian (DoG)



@zoccolermarcelo

Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

Image segmentation using thresholding

- Might there be a technology for optimization which combination of images can be used to get the best segmentation result?

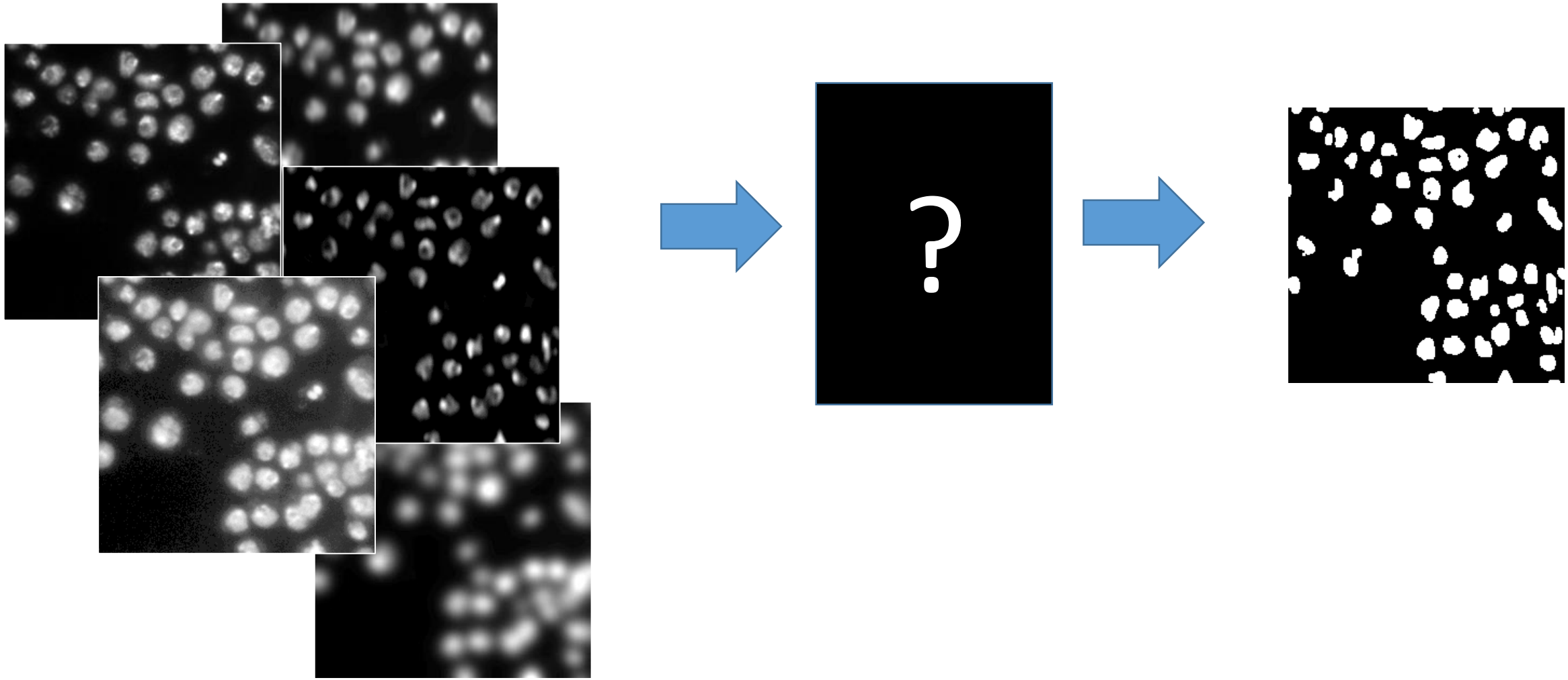
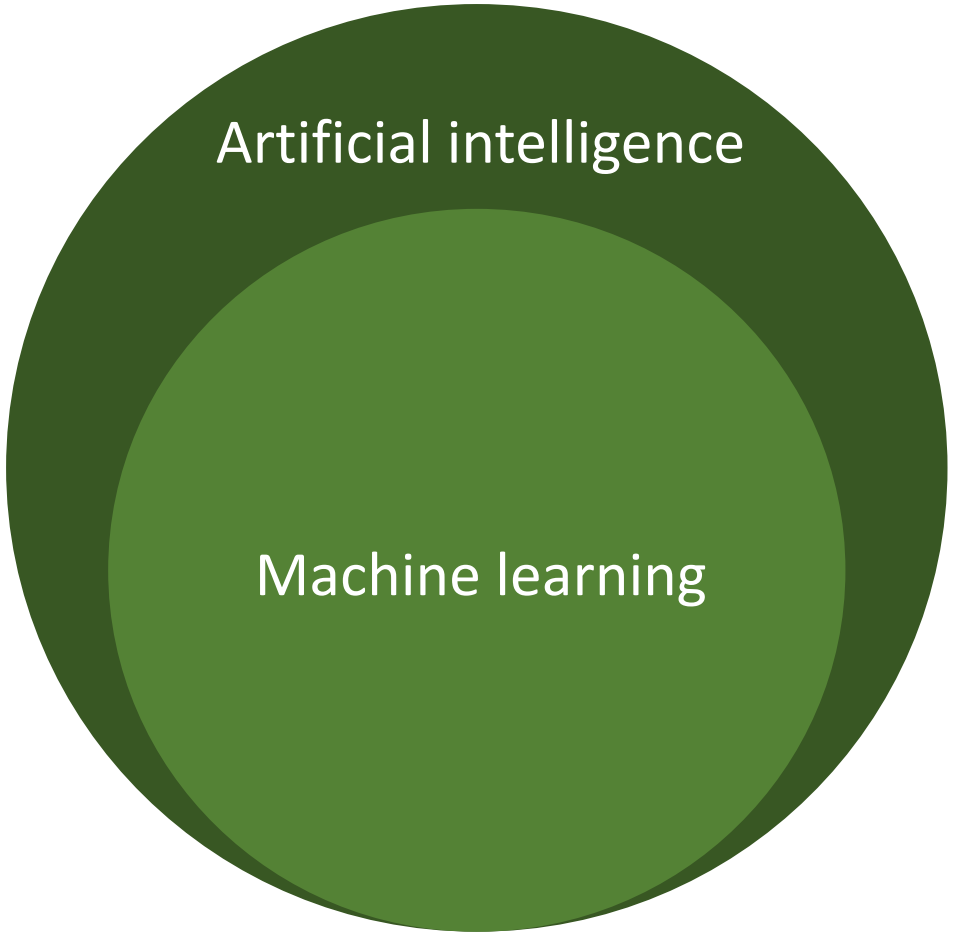
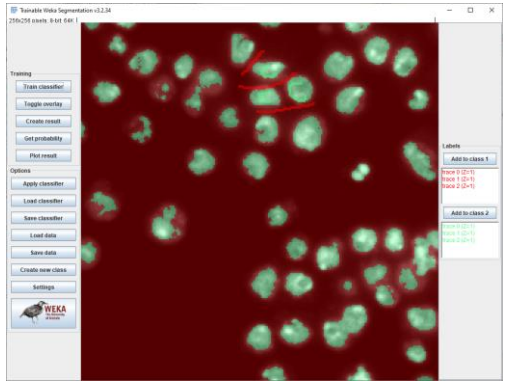


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

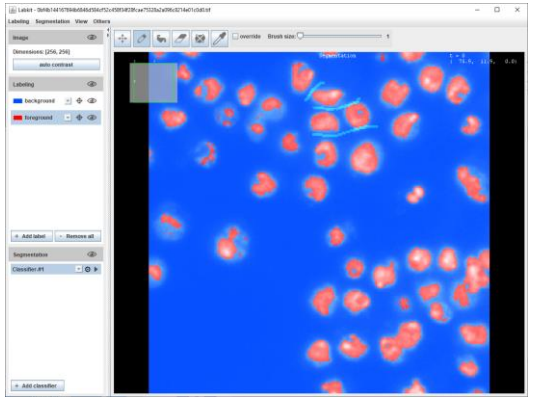
Machine learning

- A research field in computer science
- Finds more and more applications, also in life sciences.





Trainable Weka Segmentation
https://imagej.net/plugins/tws/



LabKit
https://imagej.net/plugins/labkit/

Python /
scikit-learn /
napari /
apoc

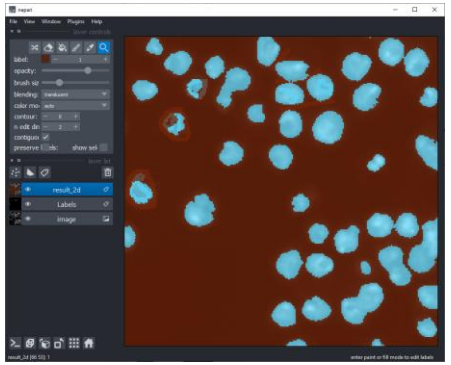
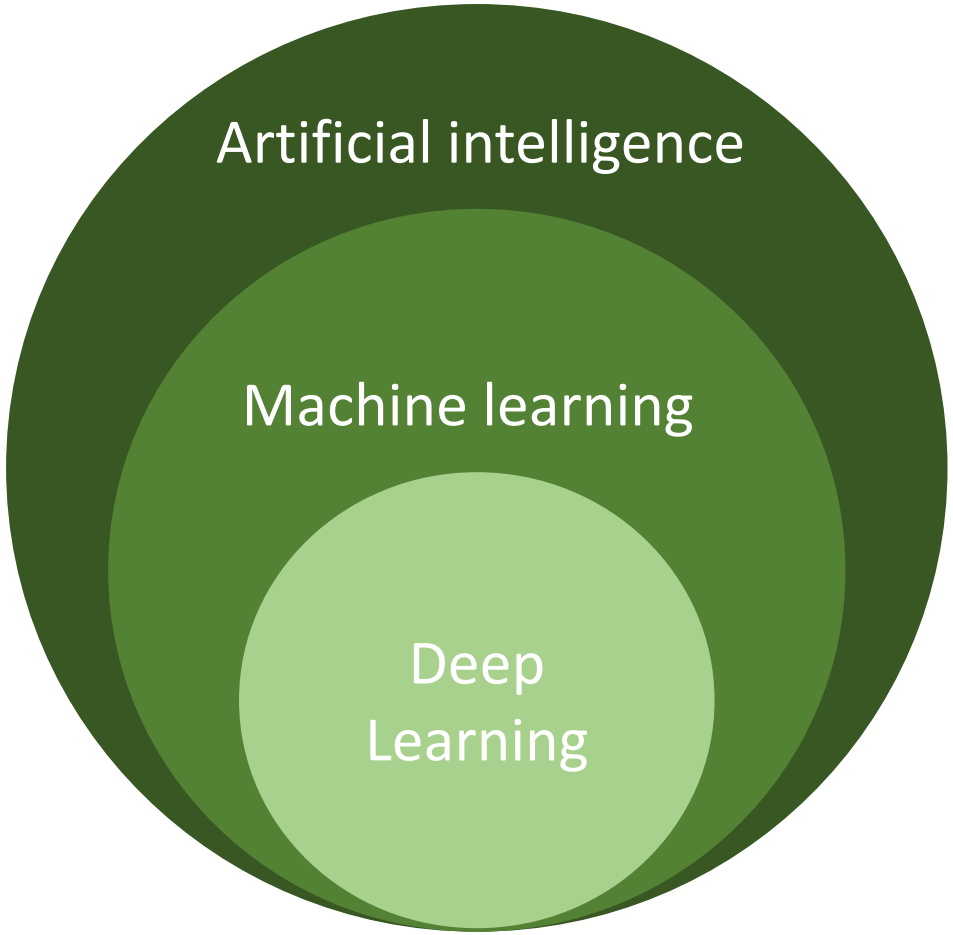
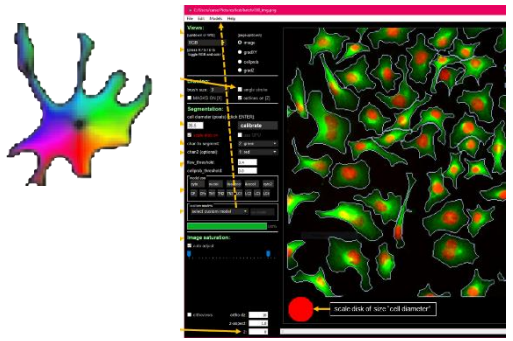


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

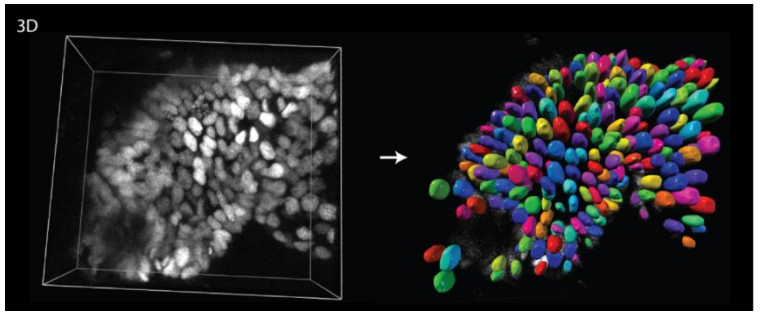
Machine learning

- A research field in computer science
- Finds more and more applications, also in life sciences.

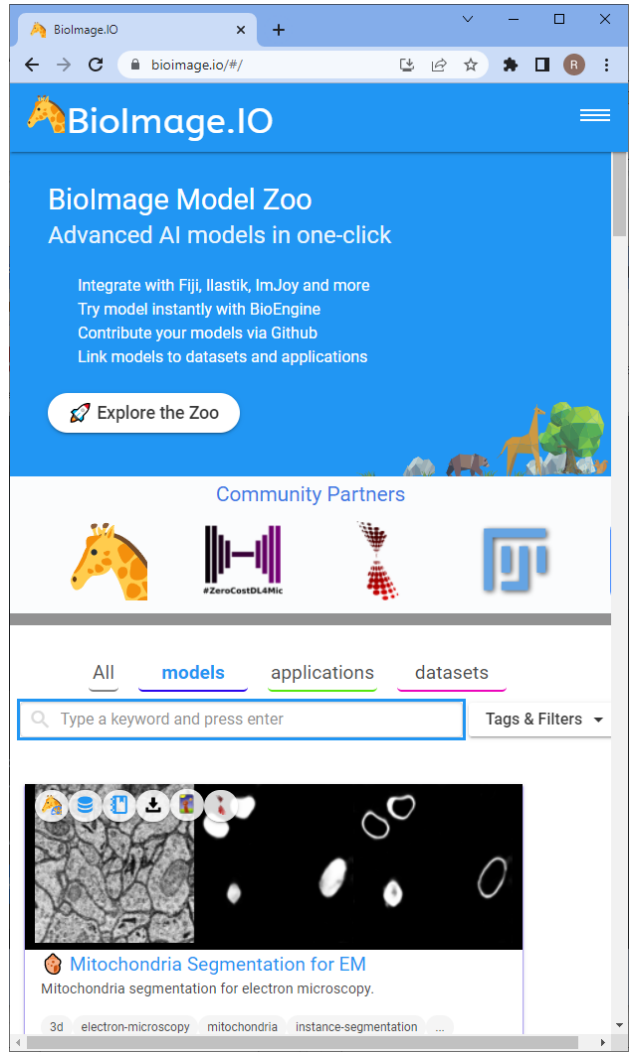




www.cellpose.org/



https://github.com/stardist/stardist



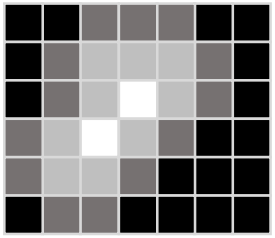
https://bioimage.io/

@zoccolermarcelo

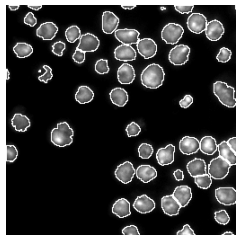
Machine learning

- Automatic construction of predictive models from given data

Pixels,

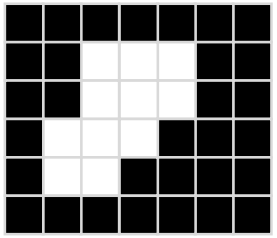


Objects,



Images, Audio, Text, Measurements, …

Dense Segmentation / Binarization



Object classification

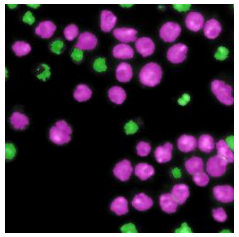
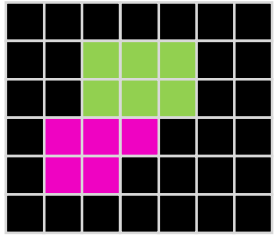


Image classification

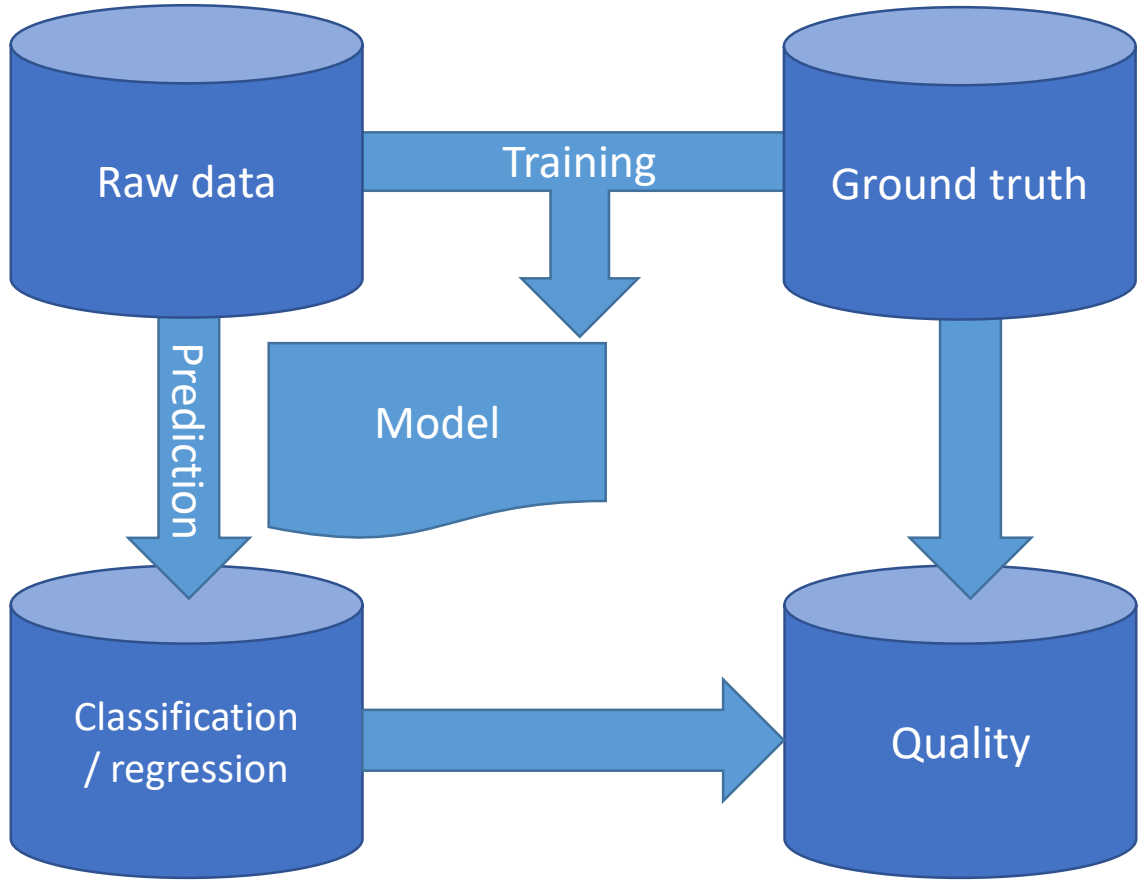
"Cat"

Instance segmentation



Cont. quantity

P_{Cat}= 0.5
$P_{Microscope}$= 0.4
Height = 80 cm



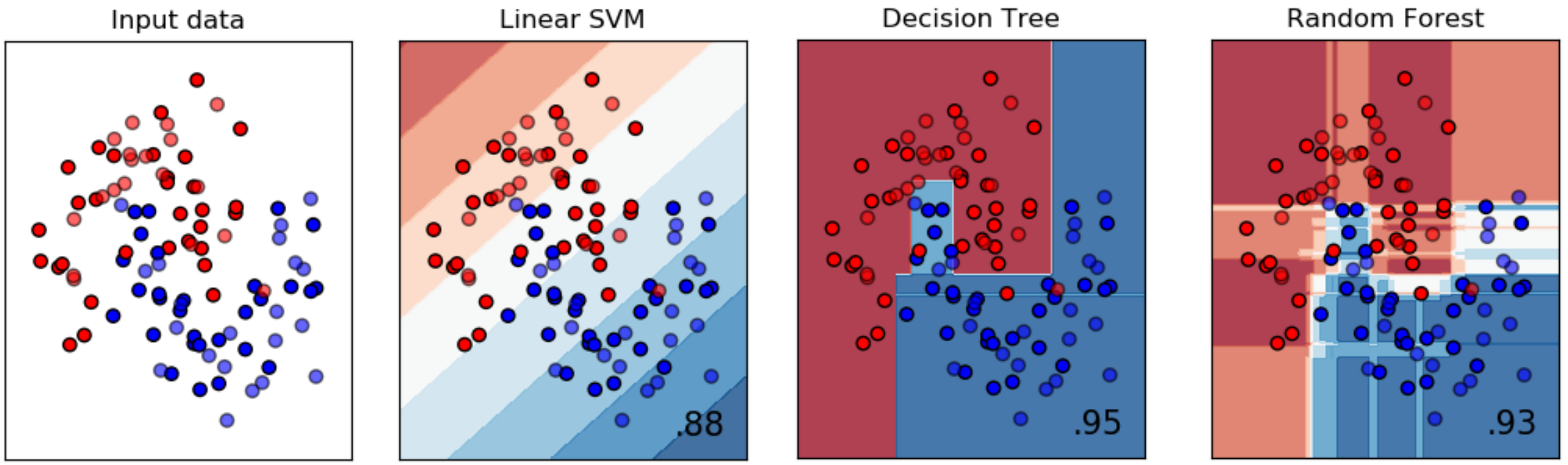
Annotated raw data, usually generated by humans

Precision, Recall

@zoccolermarcelo

Goal

- Guess classification (color) from position of a sample in parameter space.

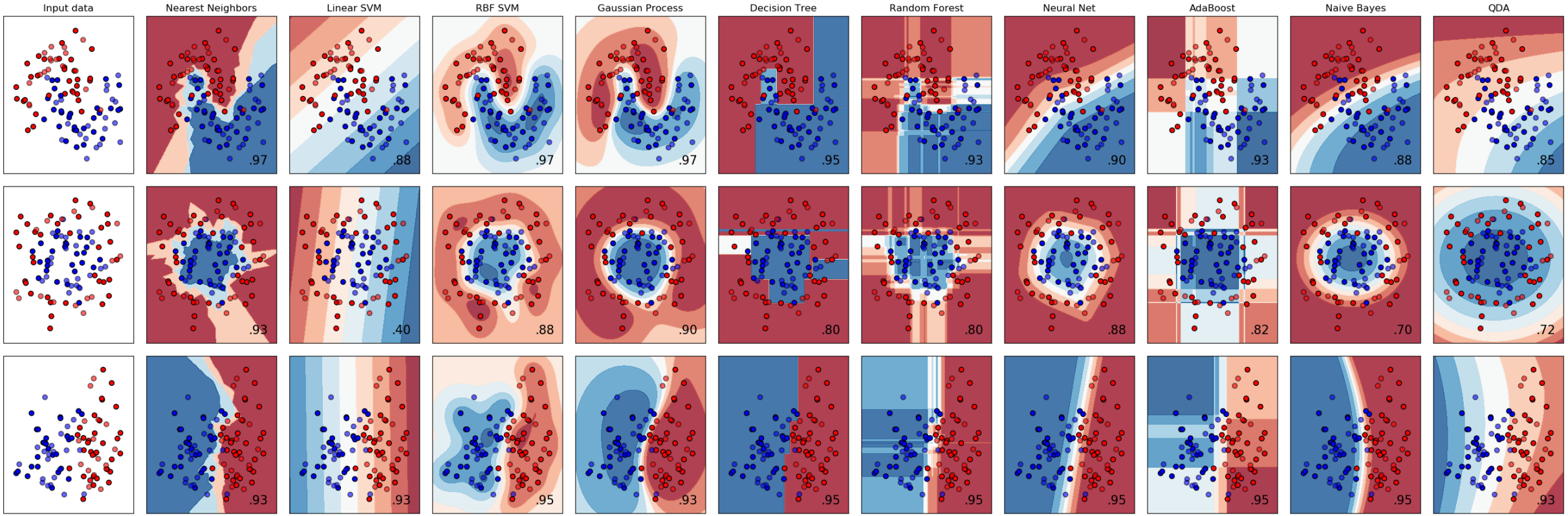


Adapted from https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

Approaches

PoL Physics of Life TU Dresden

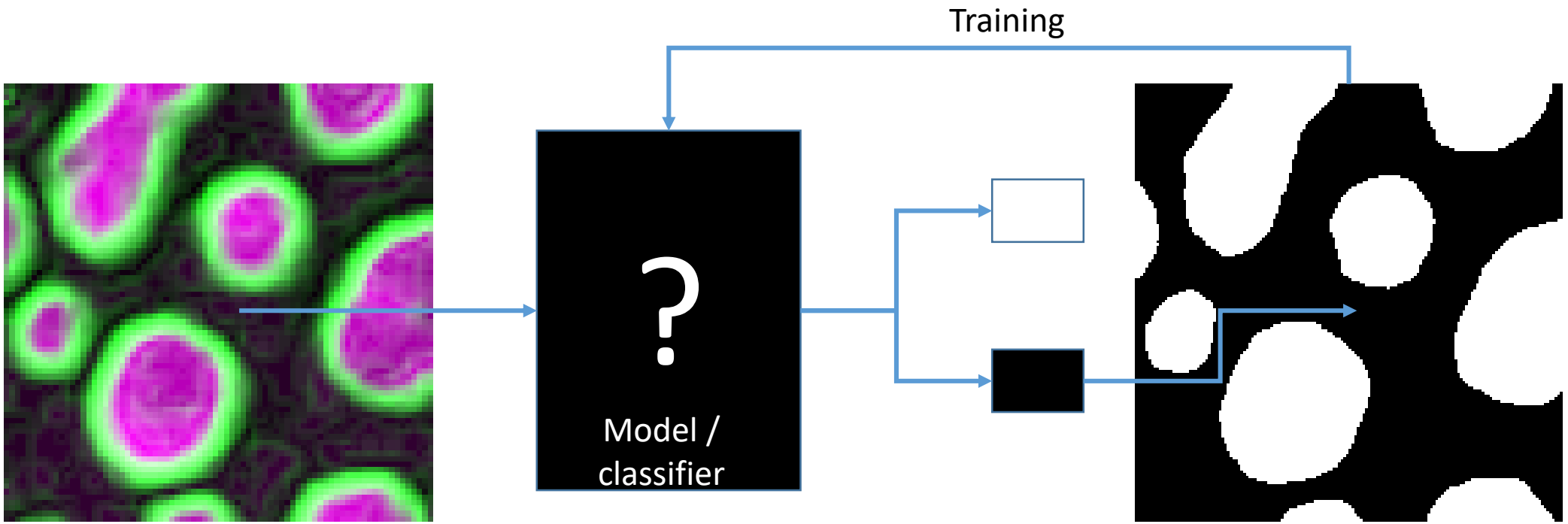
- The right approach depends on data, computational resources and desired quality



Adapted from https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

Machine learning for image segmentation

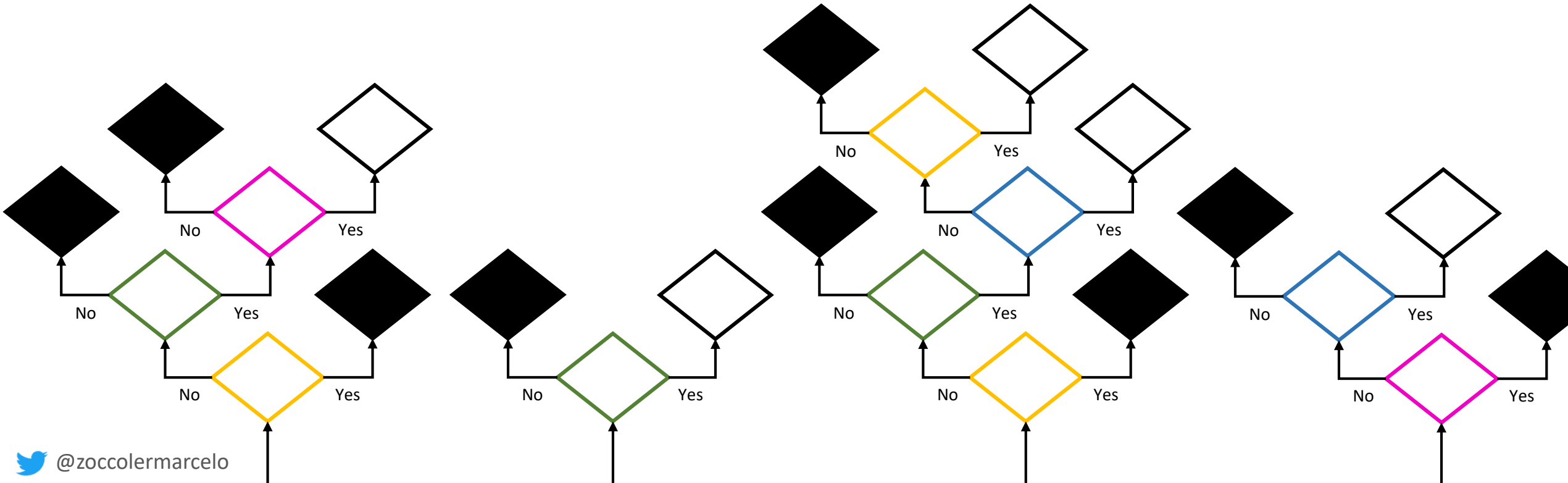
- *Supervised* machine learning: We give the computer some ground truth to learn from
- The computer derives a *model* or a *classifier* which can judge if a pixel should be foreground (white) or background (black)
- Example: Binary classifier



Random forest based image segmentation

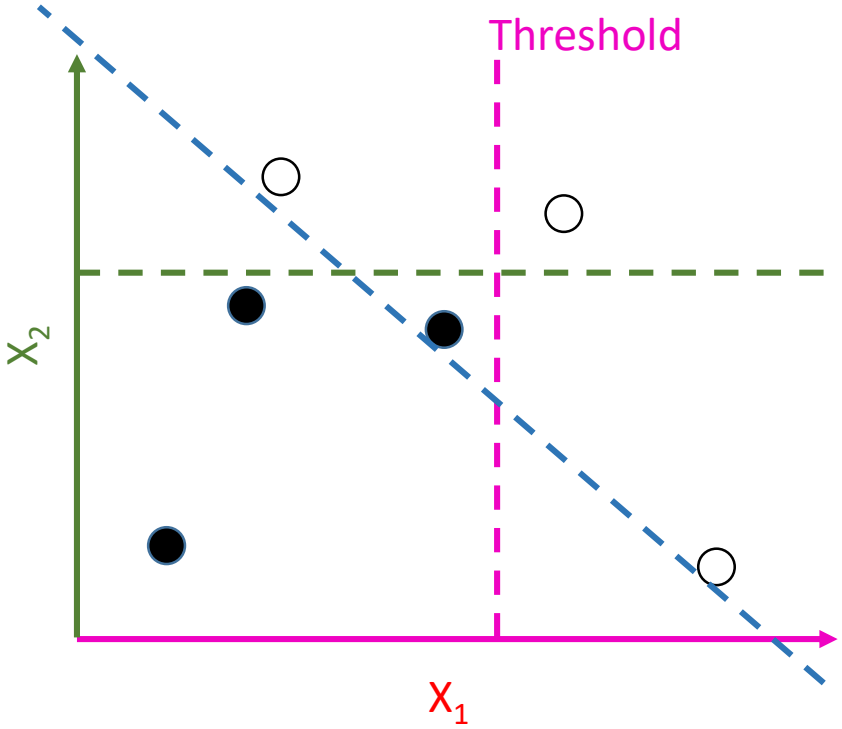
- Decision trees are classifiers, they decide if a pixel should be white or black
- Random decision trees are randomly initialized, afterwards evaluated and selected
- Random forests consist of many random decision trees

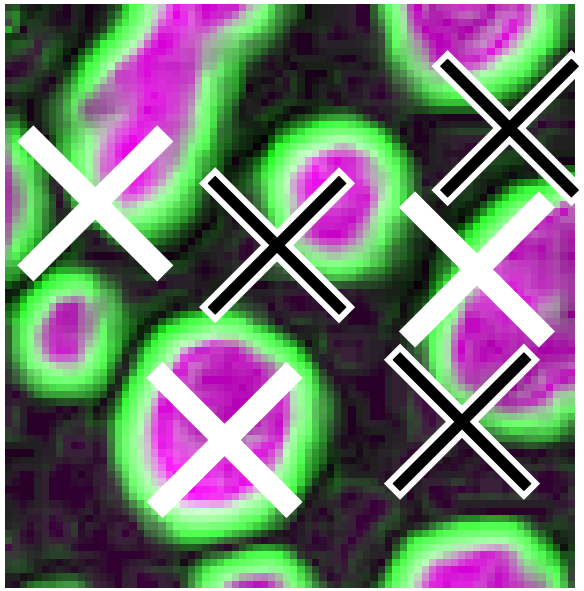
- Example: Random forest of binary decision trees

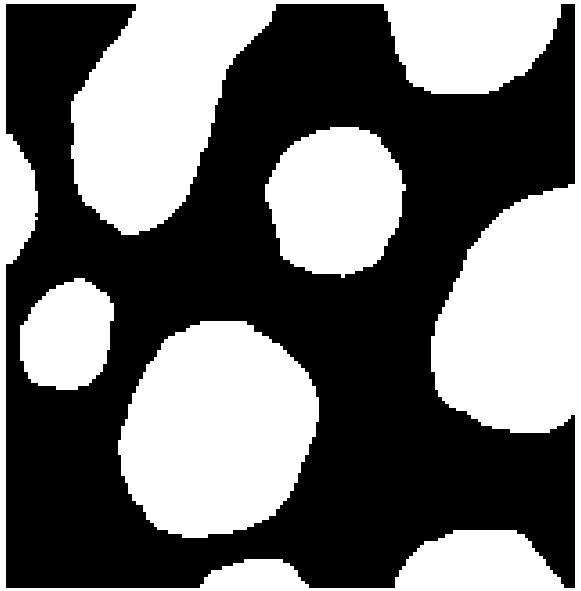


Deriving random decision trees

- For efficient processing, we randomly *sample* our data set
  - Individual pixels, their intensity and their classification



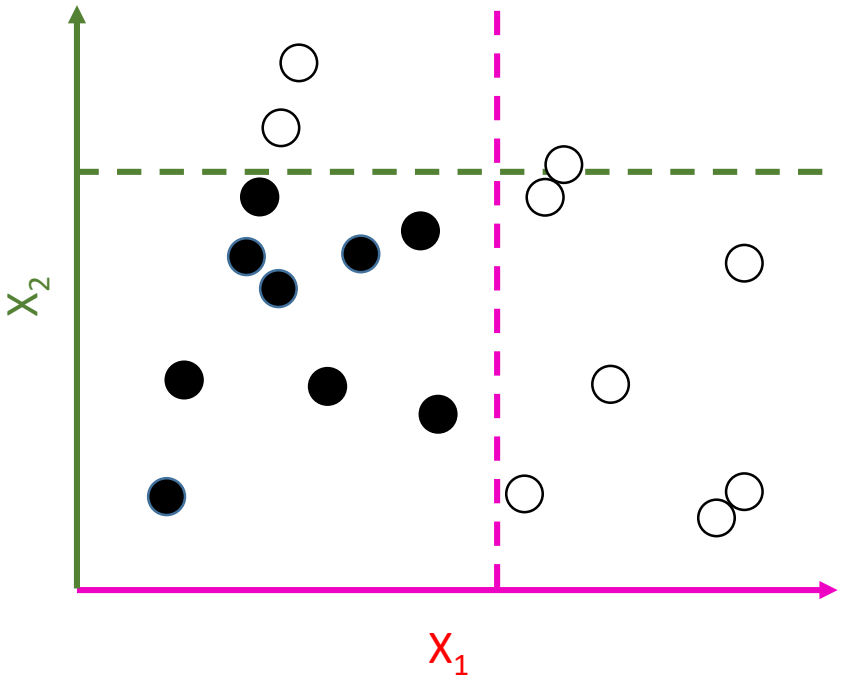


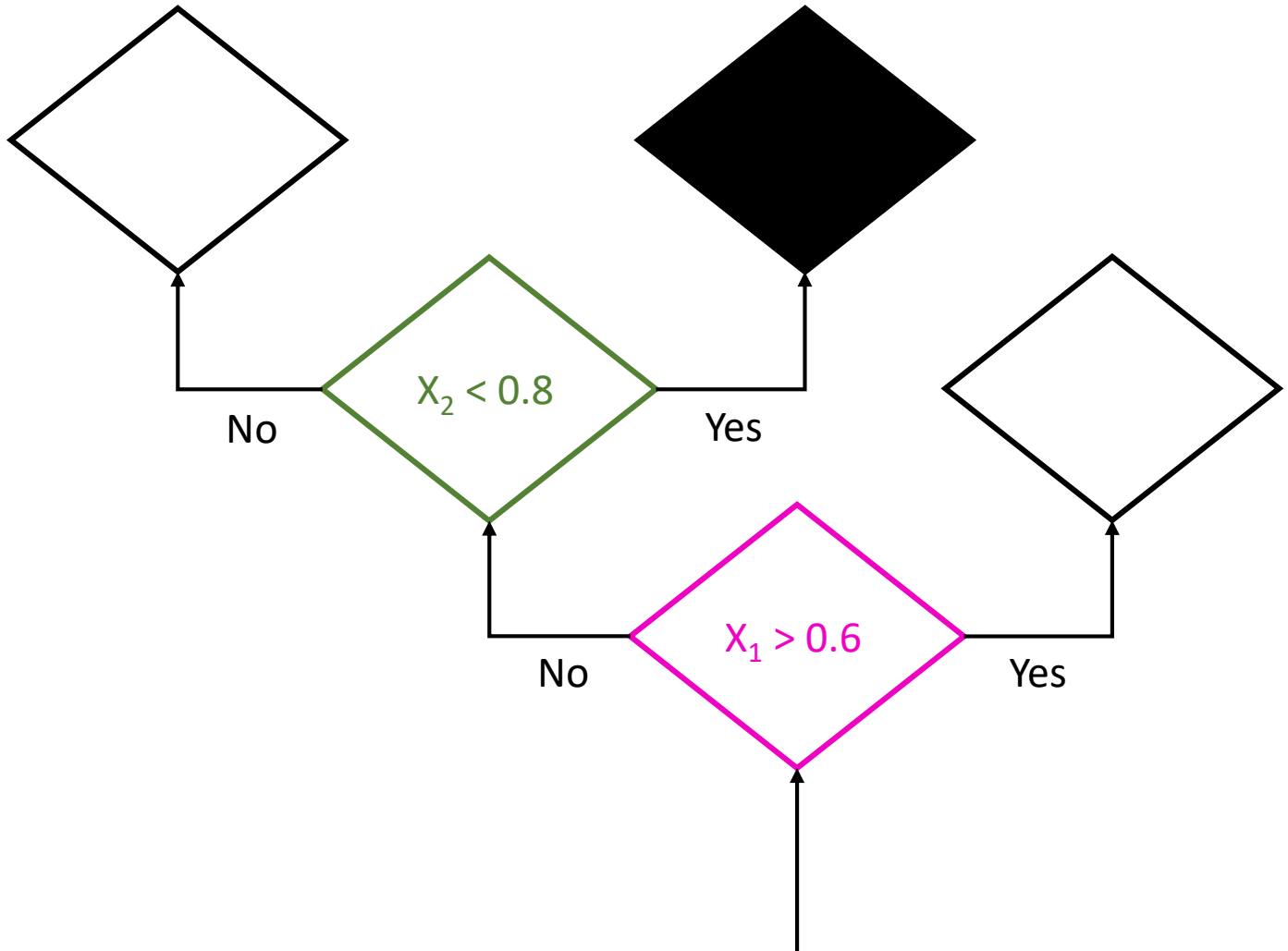


Note: You cannot use a single threshold to make the decision correctly

Deriving random decision trees

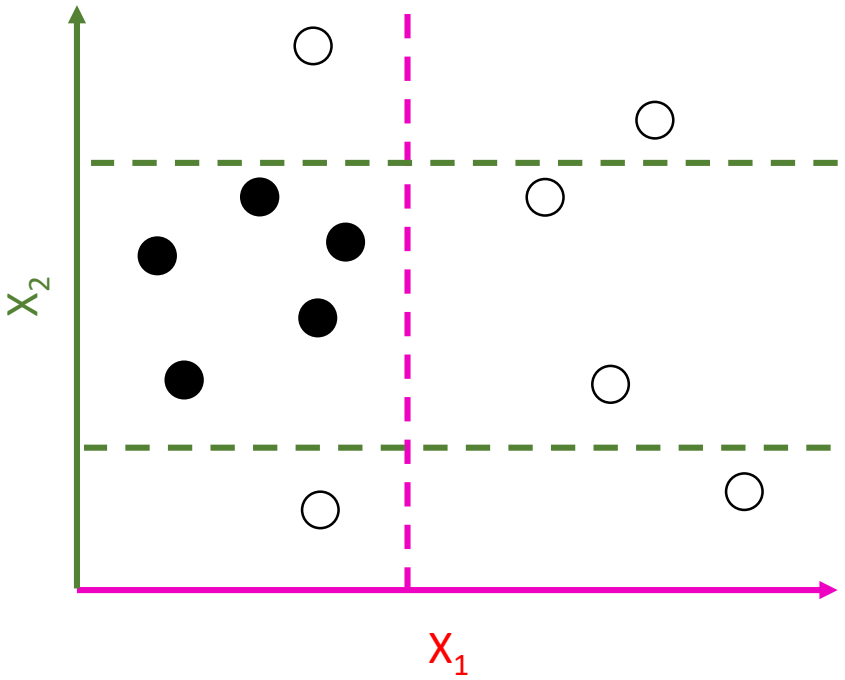
- Decision trees combine several thresholds on several parameters

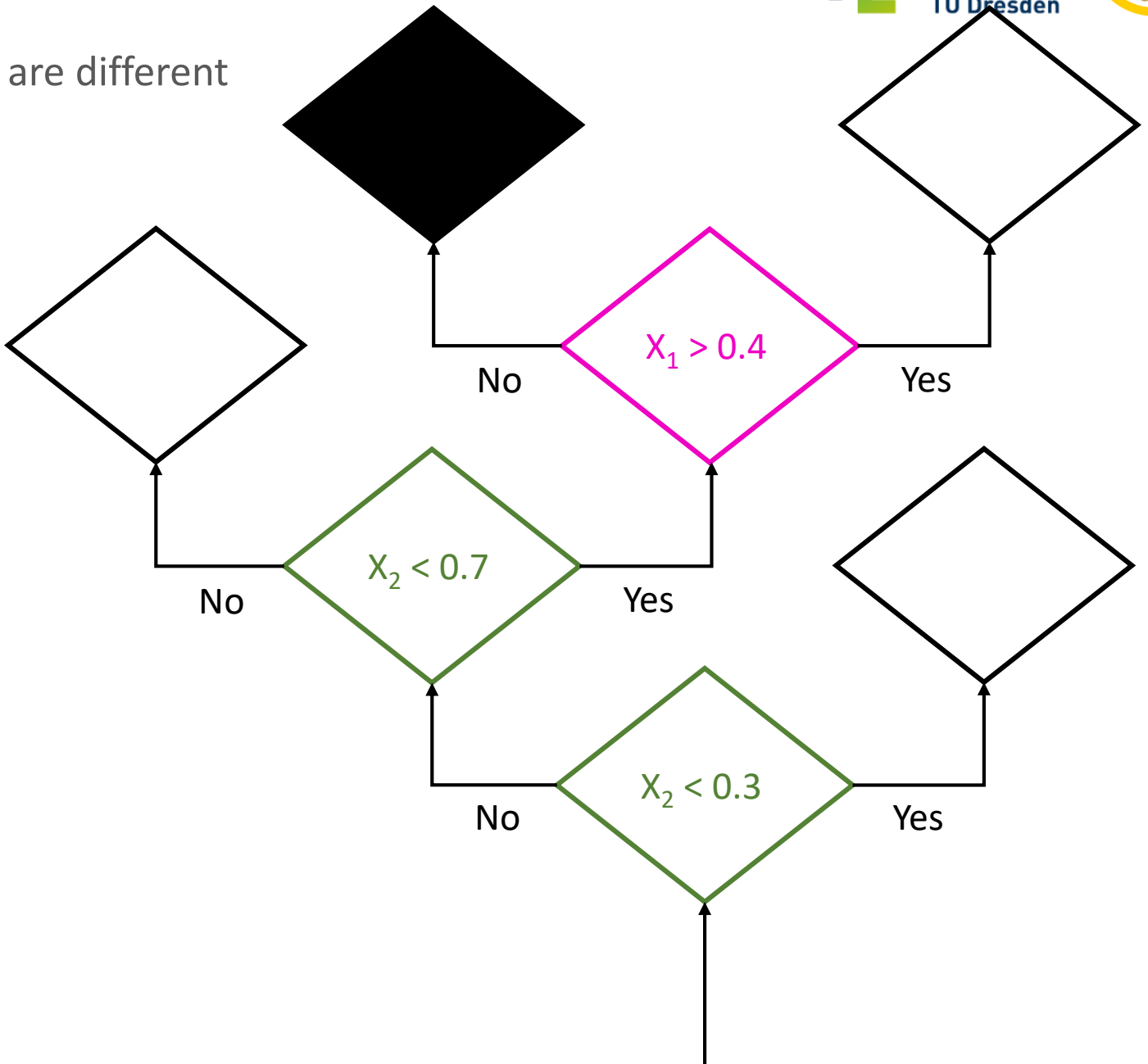




Deriving random decision trees

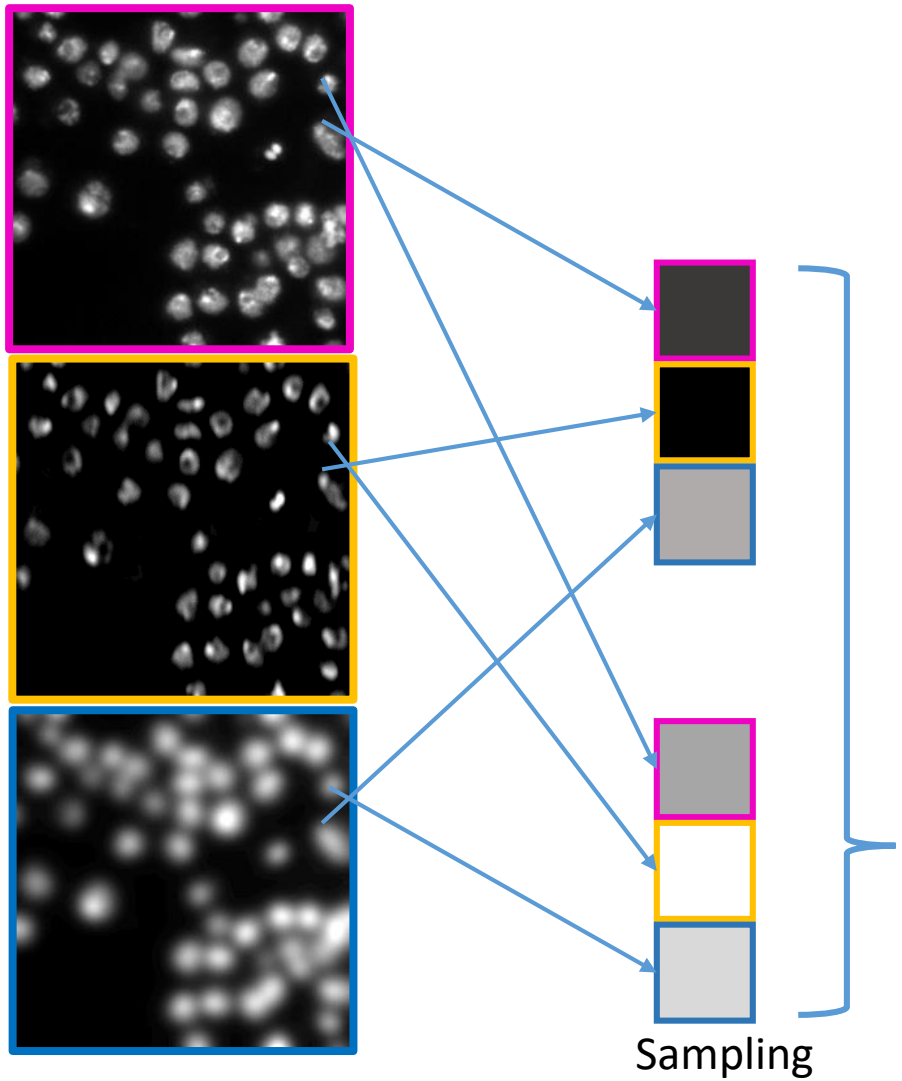
- Depending on sampling, the decision trees are different



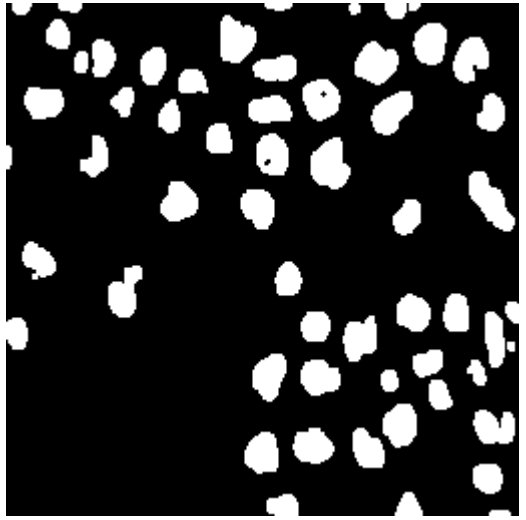


Random Forest Pixel Classifiers

- By comparing performance of individual decision trees, good ones can be selected, bad ones excluded.



Sampling



Selection

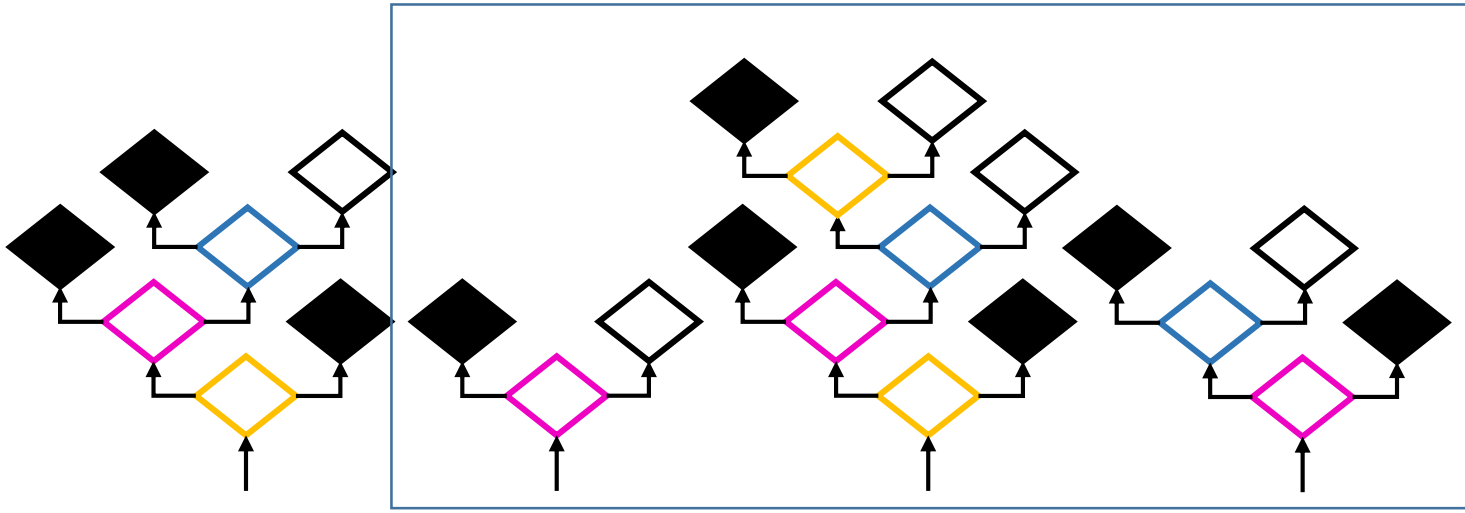
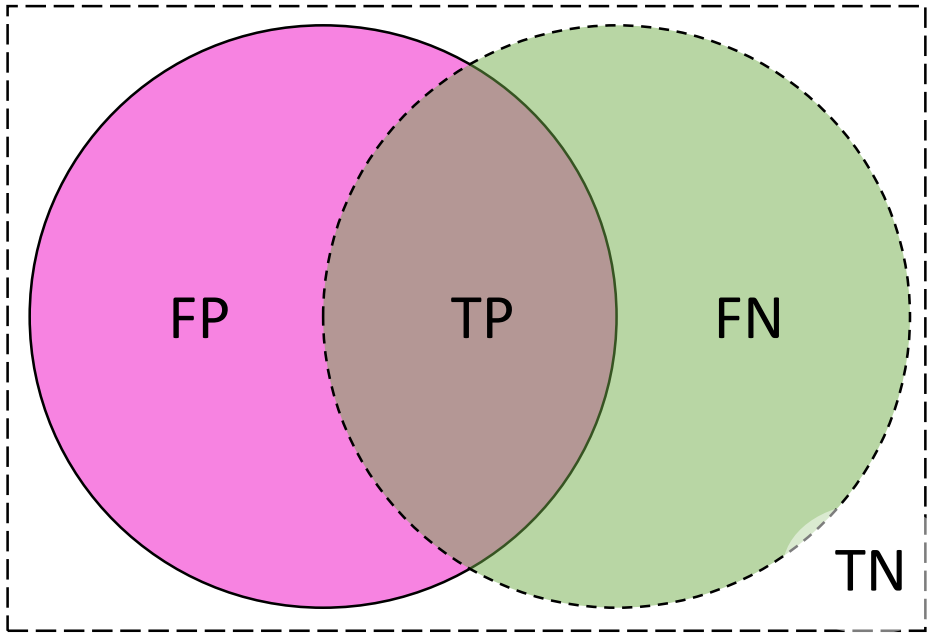


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019).

Segmentation quality estimation

- In general
  - Define what's positive and what's negative.
  - Compare with a reference to figure out what was true and false



A — Prediction A

B — Reference B (ground truth)

ROI — Region of interest

TP — True-positive

FN — False-negative

FP — False-positive

TN — True-negative

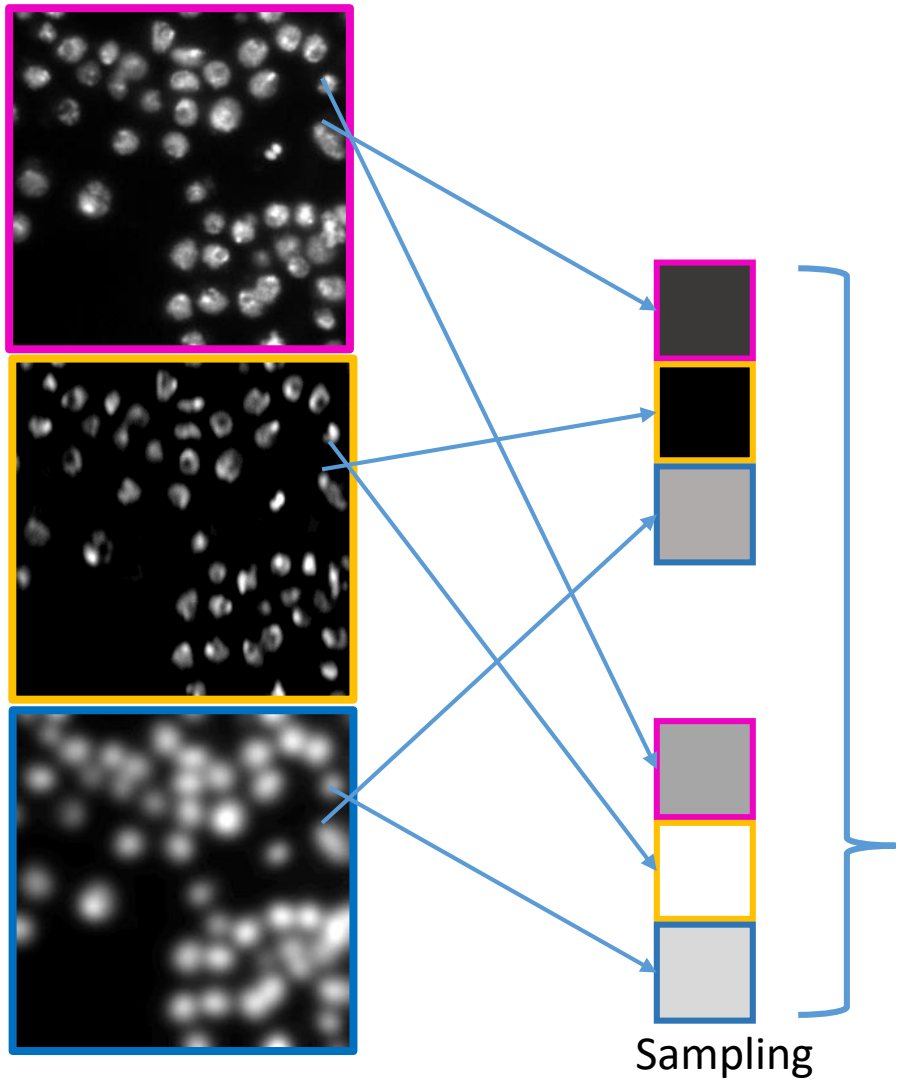
Overlap (a.k.a. Jaccard index) $\frac{TP}{TP+FN+FP}$ How much do A and B overlap?

Precision $\frac{TP}{TP+FP}$ What fraction of points that were predicted as positives were really positive?

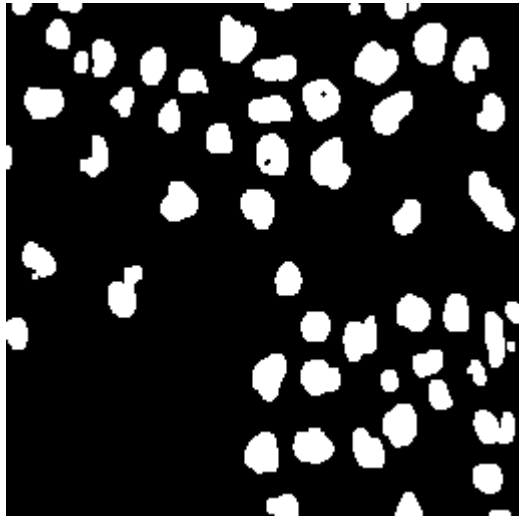
Recall (a.k.a. sensitivity) $\frac{TP}{TP+FN}$ What fraction of positives points were predicted as positives?

Random Forest Pixel Classifiers

- By comparing performance of individual decision trees, good ones can be selected, bad ones excluded.



Sampling



Selection

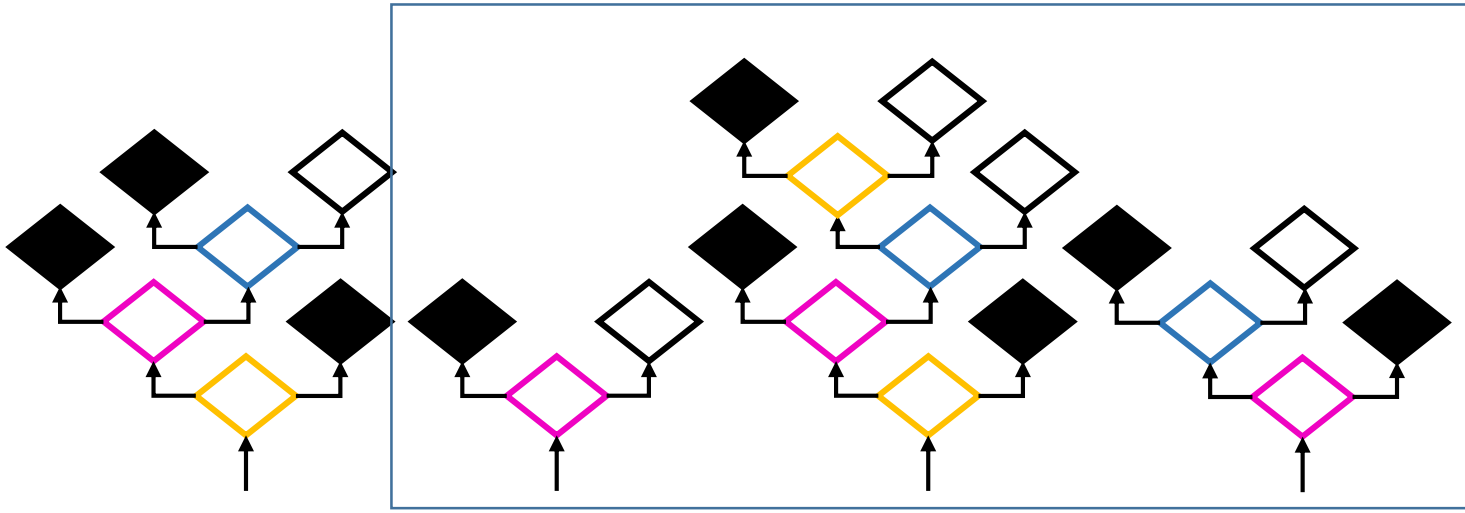
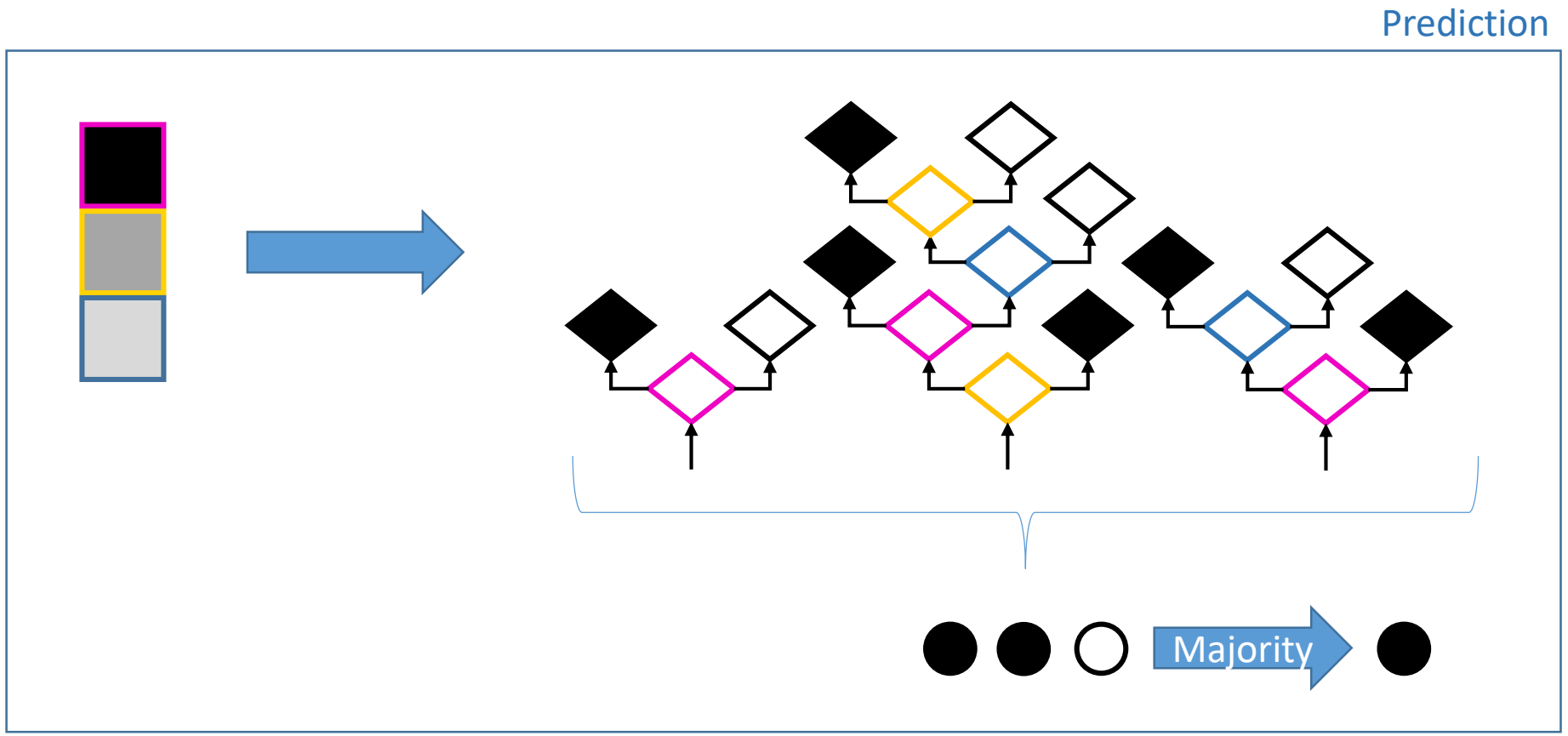


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019).

Random Forest Pixel Classifiers

- Combination of individual tree decisions by voting or max / mean

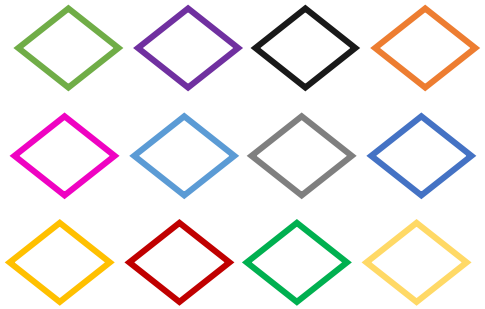


Random Forest Pixel Classifiers

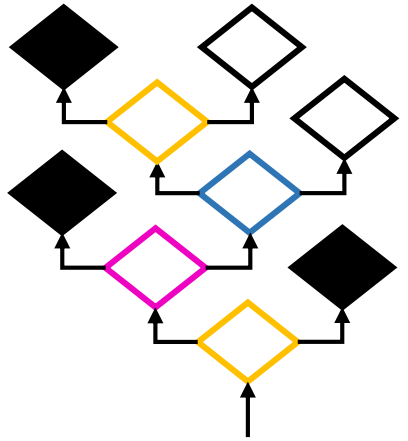
PoL Physics of Life TU Dresden

- Typical numbers for pixel classifiers in microscopy

Available features: > 20



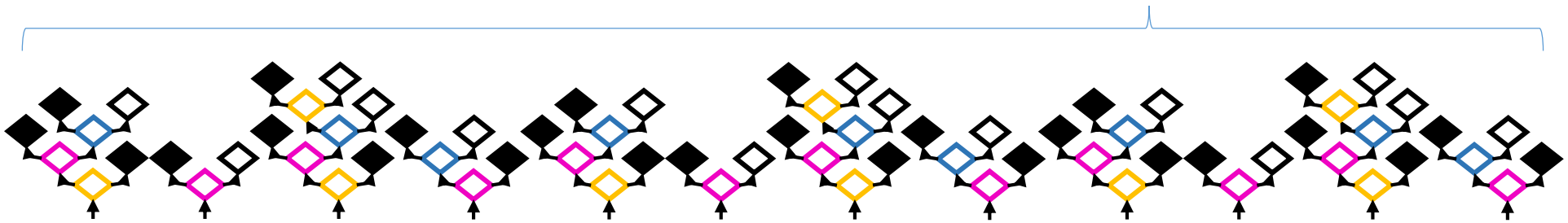
- Gaussian blur image
- DoG image
- LoG image
- Hessian
- ….



Depth: 4

Selected features: <= depth

Number of trees: > 100



Model validation

- A good classifier is trained on a hand full of datasets and works on thousands similarly well.
- In order to assess that, we split the ground truth into two set
  - Training set (50%-90% of the available data)
  - Test set (10%-50% of the available data)

Typically done with hundreds or thousands of cells / images / objects / whatever.

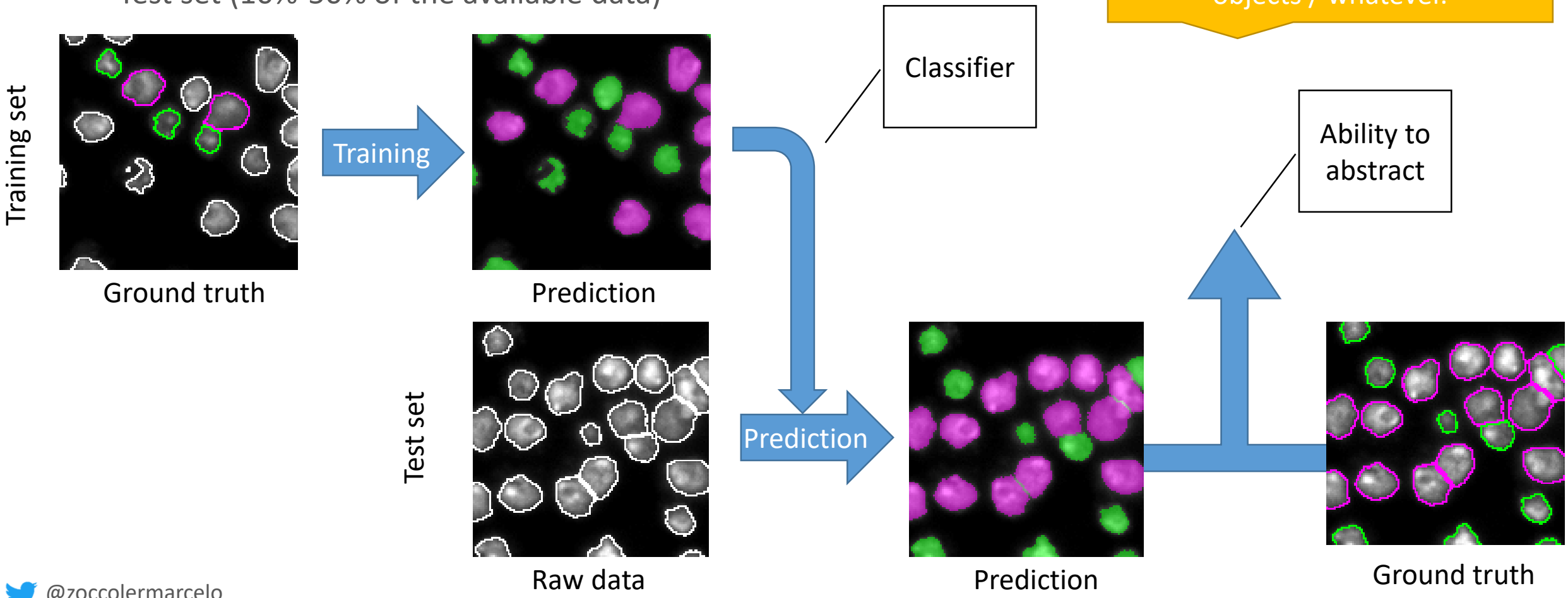
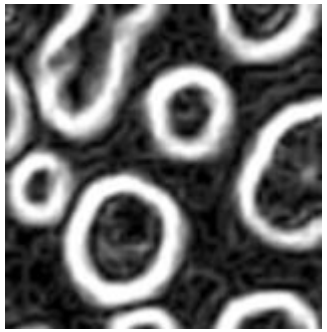


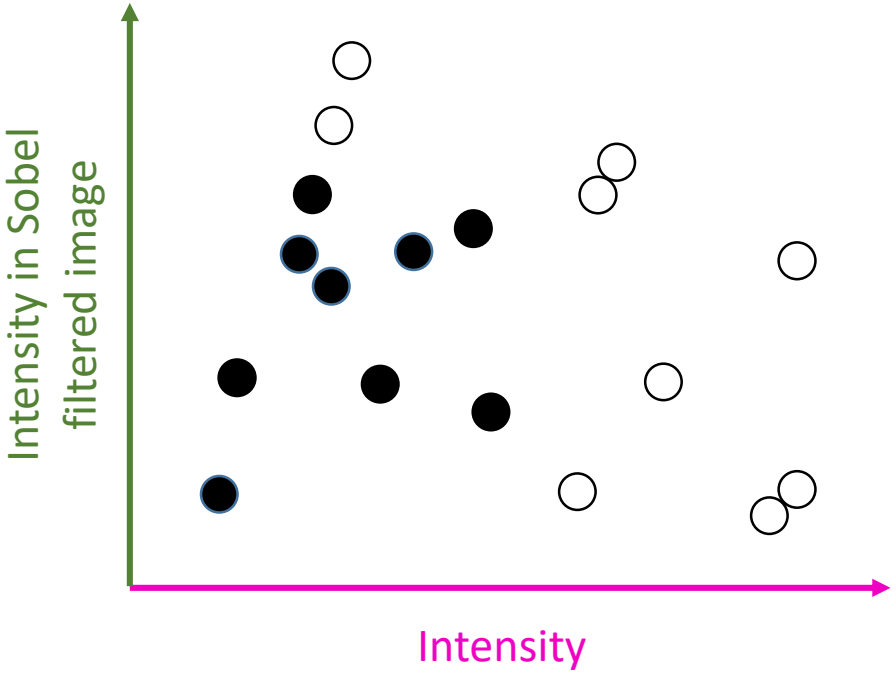
Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019).

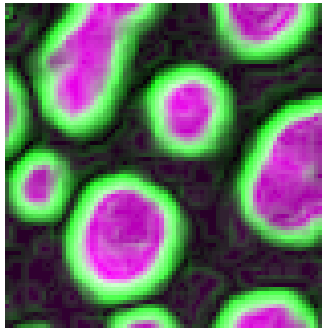
Object classification

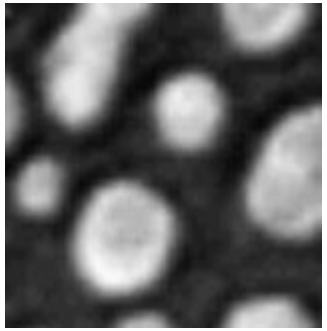
- What if we exchange pixel features with object features?

Pixel classification

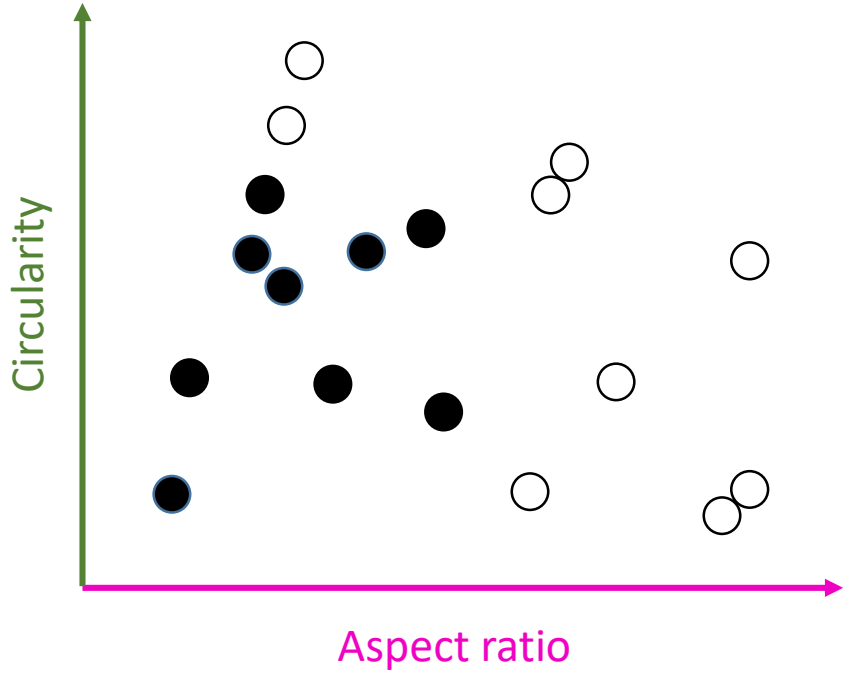








Object classification



- The algorithms work the same but with different
  - Features
  - Number of features
  - Tree / forest parameters
  - Selection criteria

Feature extraction

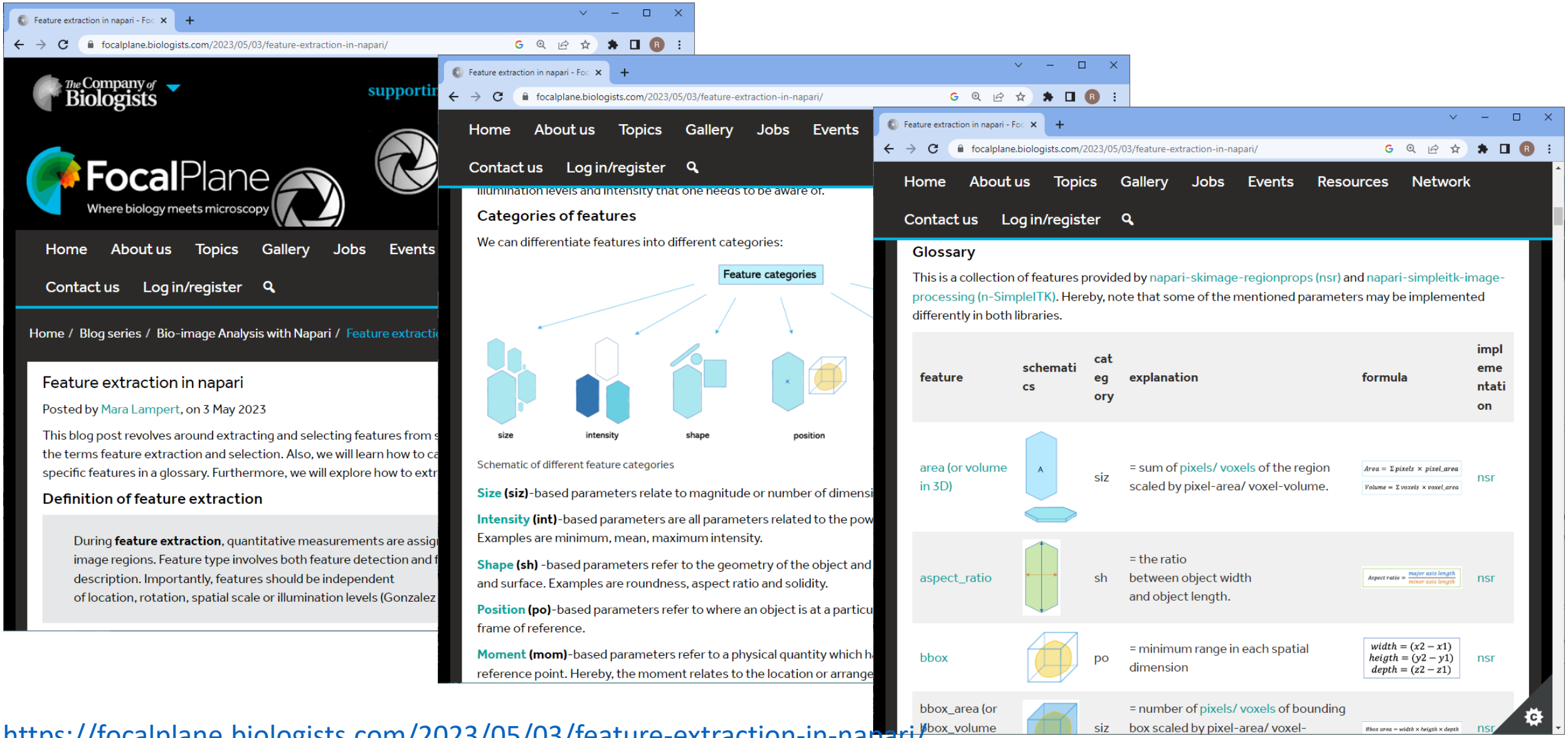
PoL Physics of Life TU Dresden

- A *feature* is a countable or measurable property of an image or object.
- Goal of feature extraction is finding a minimal set of features to describe an object well enough to differentiate it from other objects.

- **Intensity based**
  - Mean intensity
  - Standard deviation
  - Total intensity
  - Textures
- **Shape based /spatial**
  - Area / Volume
  - Roundness
  - Solidity
  - Circularity / Sphericity
  - Elongation
  - Centroid
  - Bounding box
- **Spatio-temporal**
  - Displacement,
  - Speed,
  - Acceleration
- **Others**
  - Overlap
  - Colocalization
  - Neighborhood
- **Mixed features**
  - Center of mass
  - Local minima / maxima

Further reading

PoL Physics of Life TU Dresden



https://focalplane.biologists.com/2023/05/03/feature-extraction-in-napari/

@zoccolermarcelo

PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

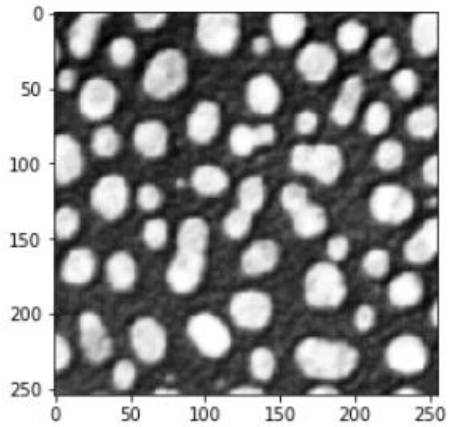
Accelerated pixel and object classification (APOC)

@zoccolermarcelo

Accelerated pixel and object classification

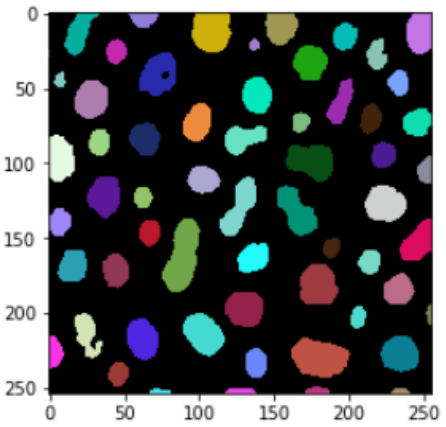
PoL Physics of Life TU Dresden

- APOC is a python library that makes use of OpenCL-compatible Graphics Cards to accelerate pixel and object classification



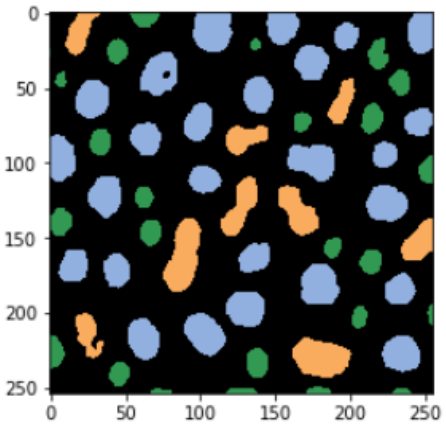
Raw image

Object segmentation

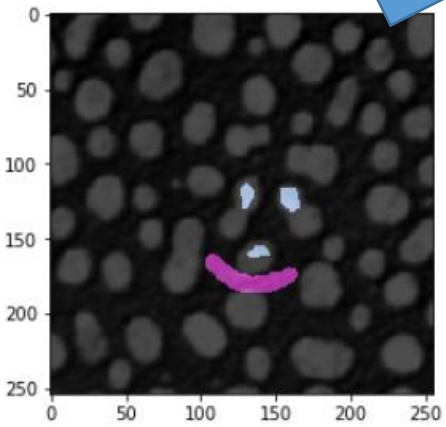


Object label image

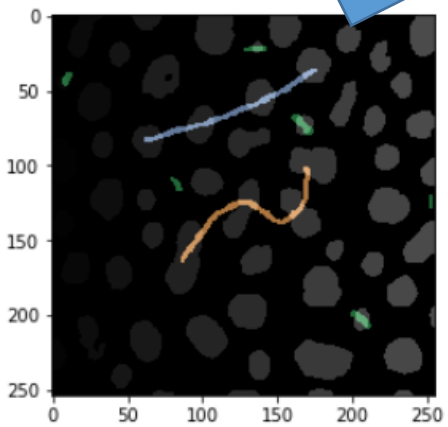
Object classification



Class label image



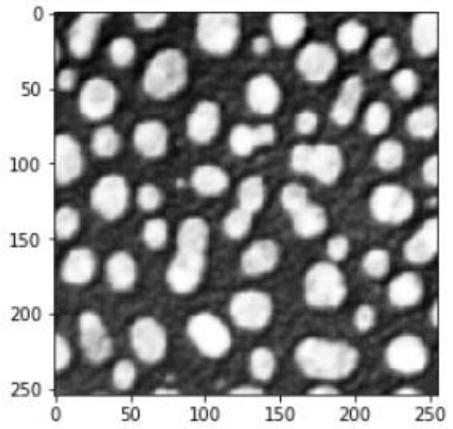
Pixel annotation



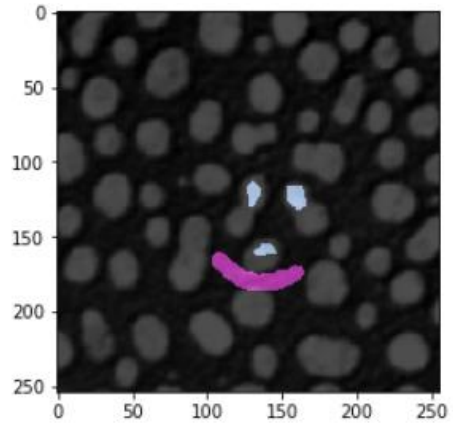
Object annotation

Object segmentation

- Pixel classification + connected component labeling



Raw image



Pixel annotation

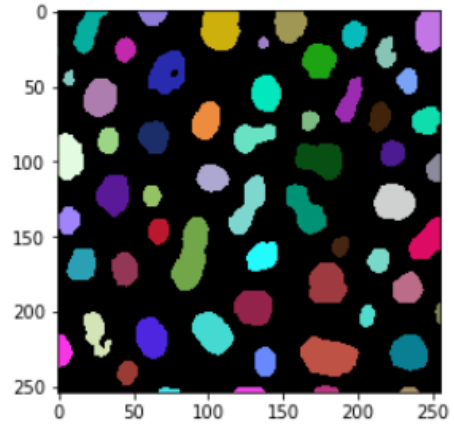
```
# define features
features = "gaussian_blur=1 gaussian_blur=5 sobel_of_gaussian_blur=1"

# this is where the model will be saved
cl_filename = 'my_object_segmenter.cl'

# delete classifier in case the file exists already
apoc.erase_classifier(cl_filename)

# train classifier
clf = apoc.ObjectSegmenter(opencl_filename=cl_filename, positive_class_identifier=2)
clf.train(features, manual_annotations, image)

segmentation_result = clf.predict(features=features, image=image)
cle.imshow(segmentation_result, labels=True)
```



Object label image

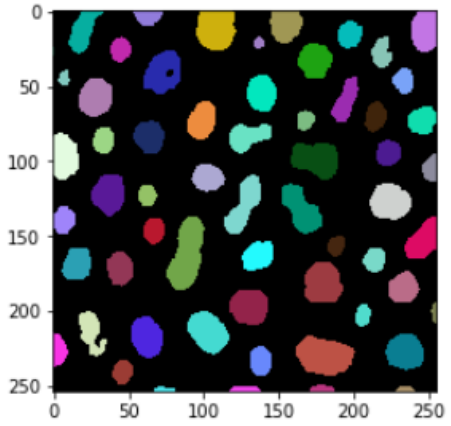
Object segmentation

@zoccolermarcelo

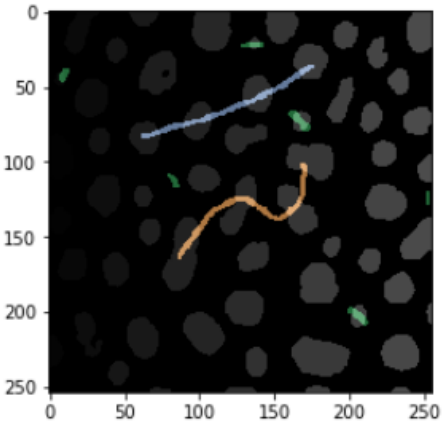
https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/03_apoc_object_segmenter.ipynb

Object classification

- Feature extraction + tabular classification



Object label image



Object annotation

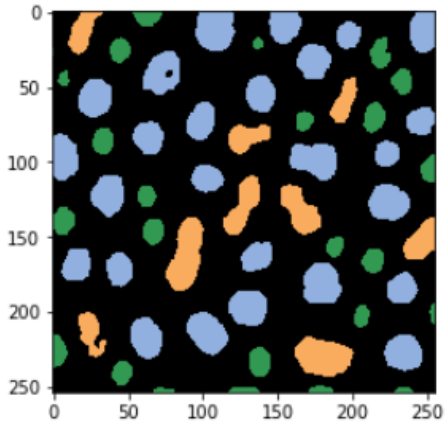
```
# for the classification we define size and shape as criteria
features = 'area mean_max_distance_to_centroid_ratio'

# This is where the model will be saved
cl_filename_object_classifier = "my_object_classifier.cl"

# delete classifier in case the file exists already
apoc.erase_classifier(cl_filename_object_classifier)

# train the classifier
classifier = apoc.ObjectClassifier(cl_filename_object_classifier)
classifier.train(features, segmentation_result, annotation, image)

# determine object classification
classification_result = classifier.predict(segmentation_result, image)
cle.imshow(classification_result, labels=True)
```



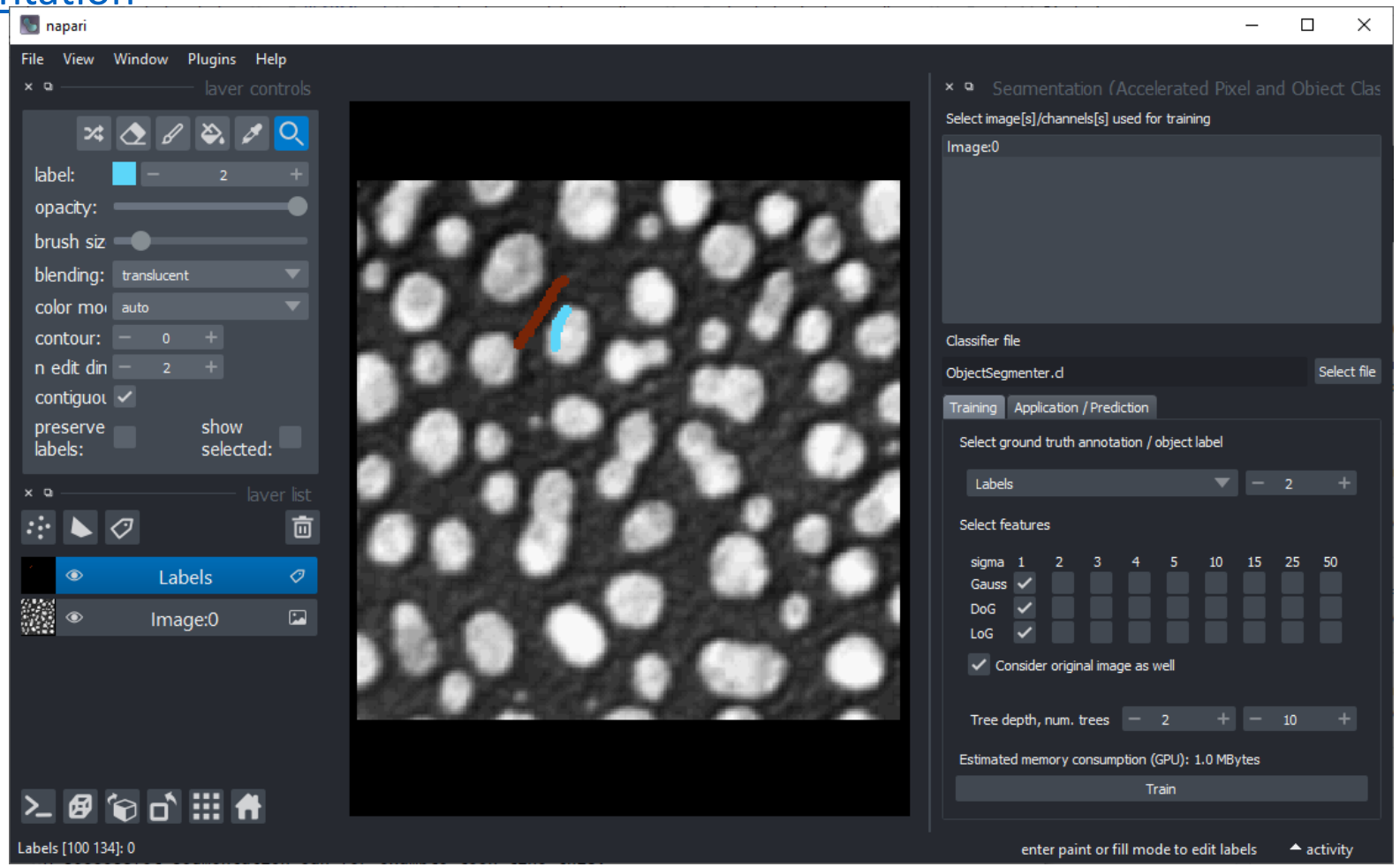
Class label image

Object classification

https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/03_apoc_object_segmenter.ipynb

Graphical user interface

- Object segmentation
- https://github.com/haesleinhuepf/napari-accelerated-pixel-and-object-classification#object-and-semantic-segmentation



Exercises

Pixel classification / object segmentation

- Use Napari to segment objects

Interactive pixel classification and object segmentation in Napari

In this exercise we will train a Random Forest Classifier for pixel classification and convert the result in an instance segmentation. We will use the napari plugin napari-accelerated-pixel-and-object-classification.

Getting started

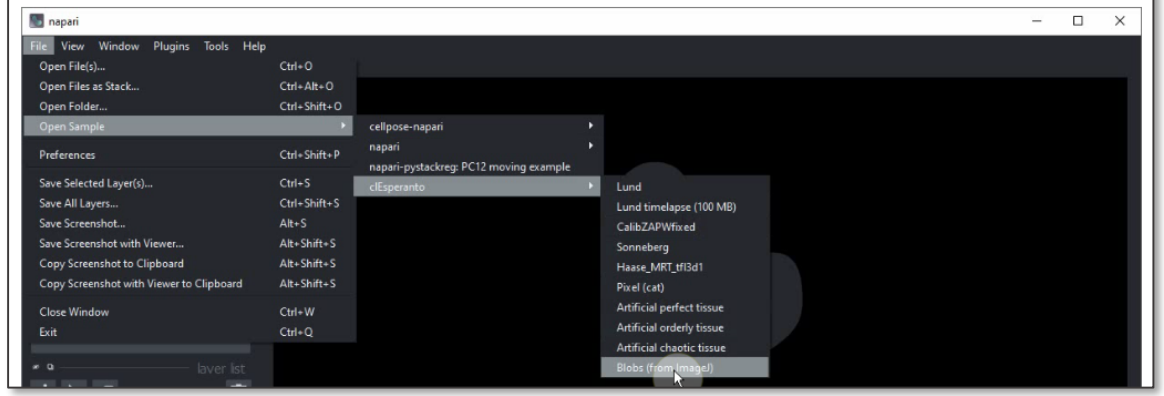
Open a terminal window and activate your conda environment:

```
conda activate devbio-napari-env
```

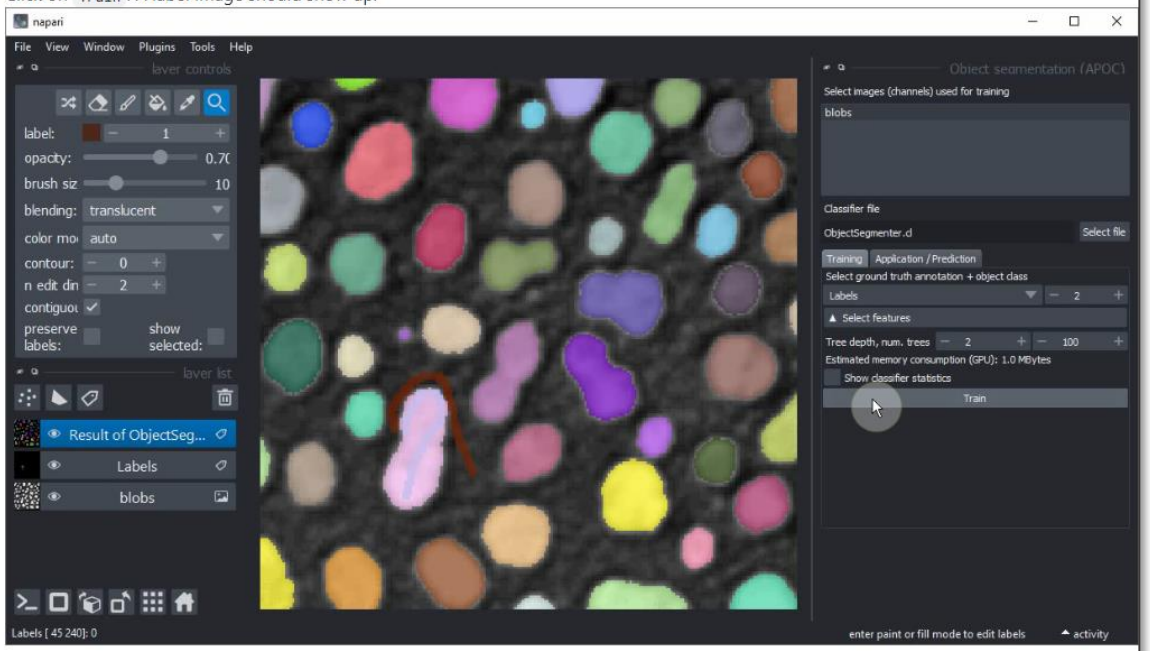
Afterwards, start up Napari:

```
napari
```

Load the "Blobs" example dataset from the menu `File > Open Sample > clEsperanto > Blobs (from ImageJ)`



Click on `Train`. A label image should show up.



If the segmentation works well, consider backing up the `ObjectSegmenter.cl` file that has been saved. If you didn't change the file location before training, it will be located in the folder from where you started napari on the command line.

https://zoccoler.github.io/QM_Course_Bio_Image_Analysis_with_napari_2024/Machine_Learning_with_napari/interactive_pixel_classification/readme.html

Object classification

- Use Napari to group round and elongated objects

Interactive object classification in Napari

In this exercise we will train a Random Forest Classifiers for classifying segmented objects. We will use the napari plugin napari-accelerated-pixel-and-object-classification.

Getting started

Open a terminal window and activate your conda environment:

```
conda activate devbio-napari-env
```

Afterwards, start up Napari:

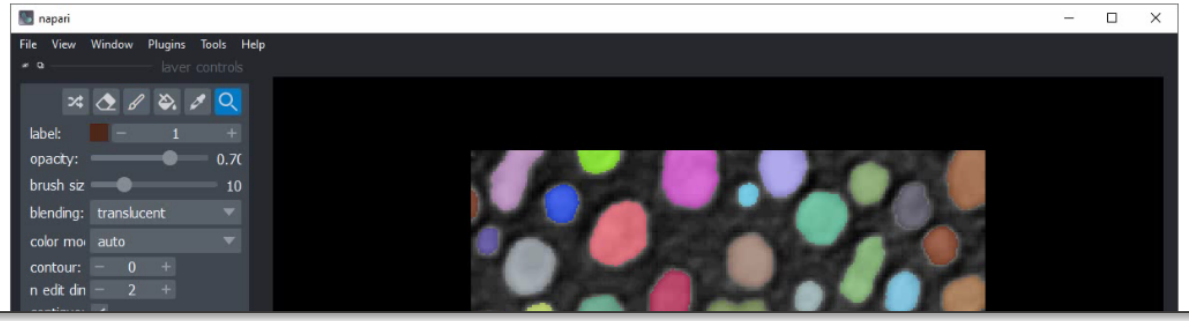
```
napari
```

Load the "Blobs" example dataset from the menu `File > Open Sample > clEsperanto > Blobs (from ImageJ)`

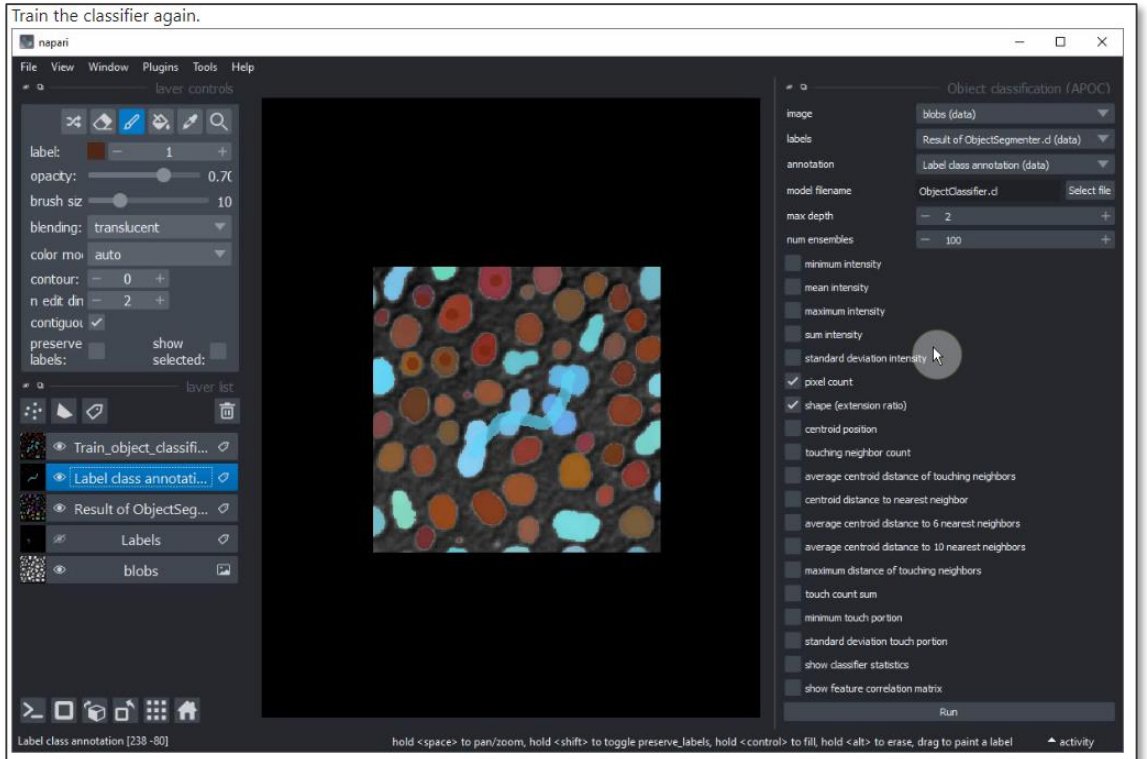
We furthermore need a label image. You can create it using the pixel classifier trained earlier or using the menu `Tools > Segmentation / labeling > Gauss-Otsu Labeling (clesperanto)`.

Object classification

Our starting point is a loaded image and a label image with segmented objects. The following procedure is also shown in this video.



Train the classifier again.



If you are happy with the trained classifier, copy the file to a safe place. When training the next classifier this one might be overwritten.

Extra exercise

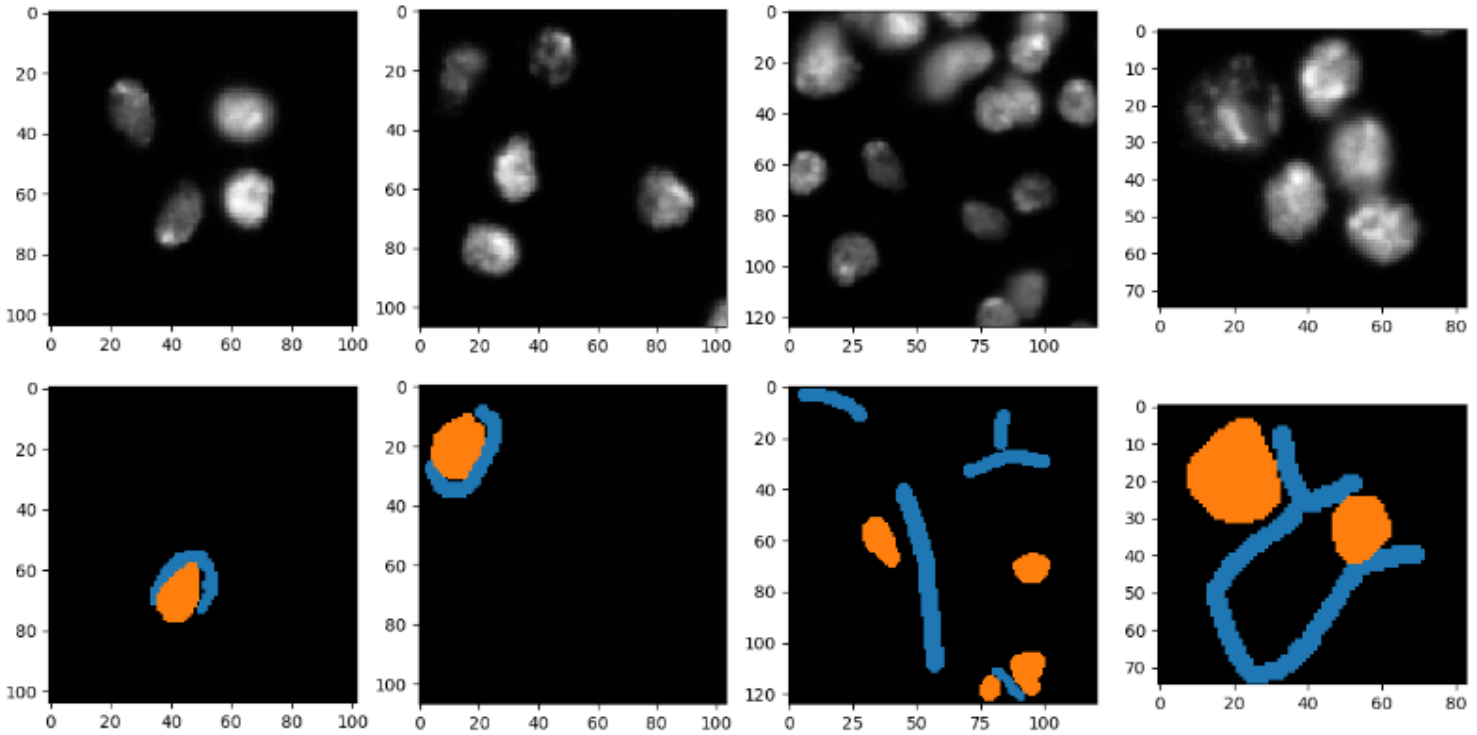
Retrain the classifier so that it can differentiate three different classes:

- Small round objects
- Large round objects
- Large elongated objects

https://zoccoler.github.io/QM_Course_Bio_Image_Analysis_with_napari_2024/Machine_Learning_with_napari/interactive_object_classification/readme.html

Pixel classifier on multiple images

- Use apoc to train on multiple images in a folder
- Run jupyter notebook below with Jupyter Lab



https://github.com/zoccoler/QM_Course_Bio_Image_Analysis_with_napari_2024/blob/main/docs/Machine_Learning_with_napari/apoc_train_on_folders.ipynb (original source: apoc repository - https://github.com/haesleinhuepf/apoc/blob/main/demo/demp_pixel_classifier_continue_training.ipynb)

PoL
Physics of Life
TU Dresden

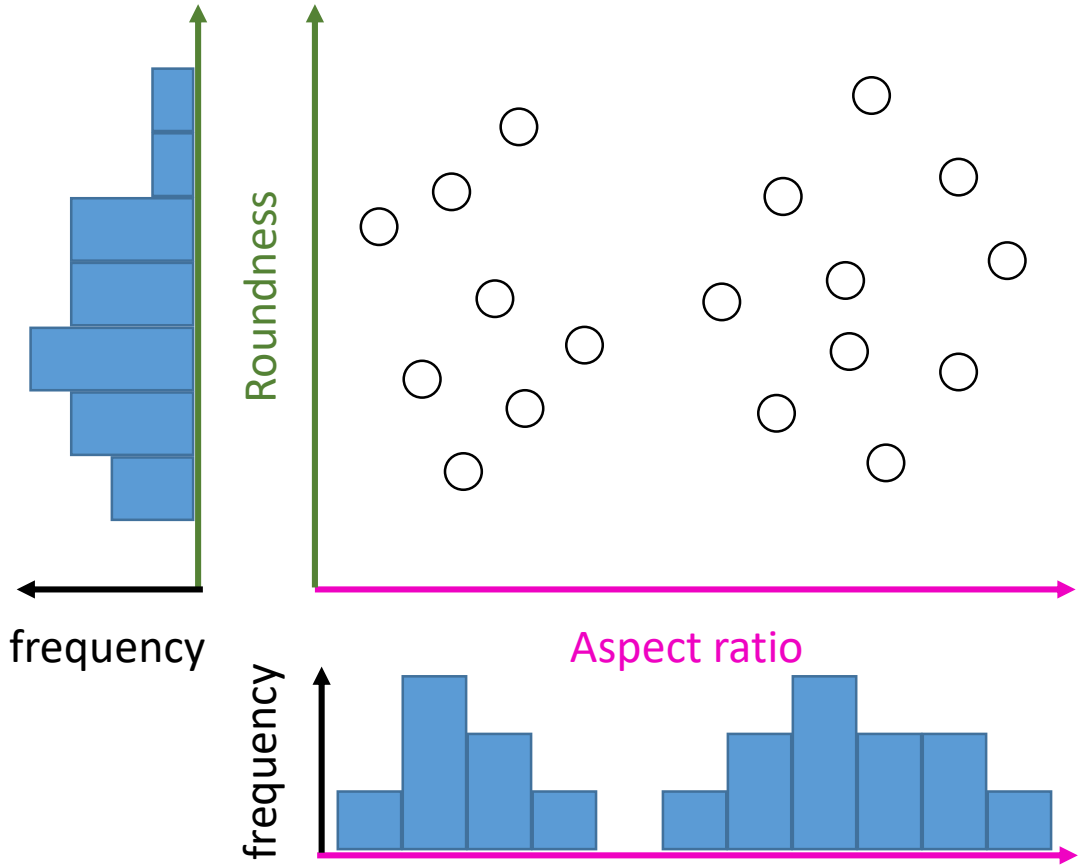
CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

Unsupervised Machine Learning for Object Characterization

@zoccolermarcelo

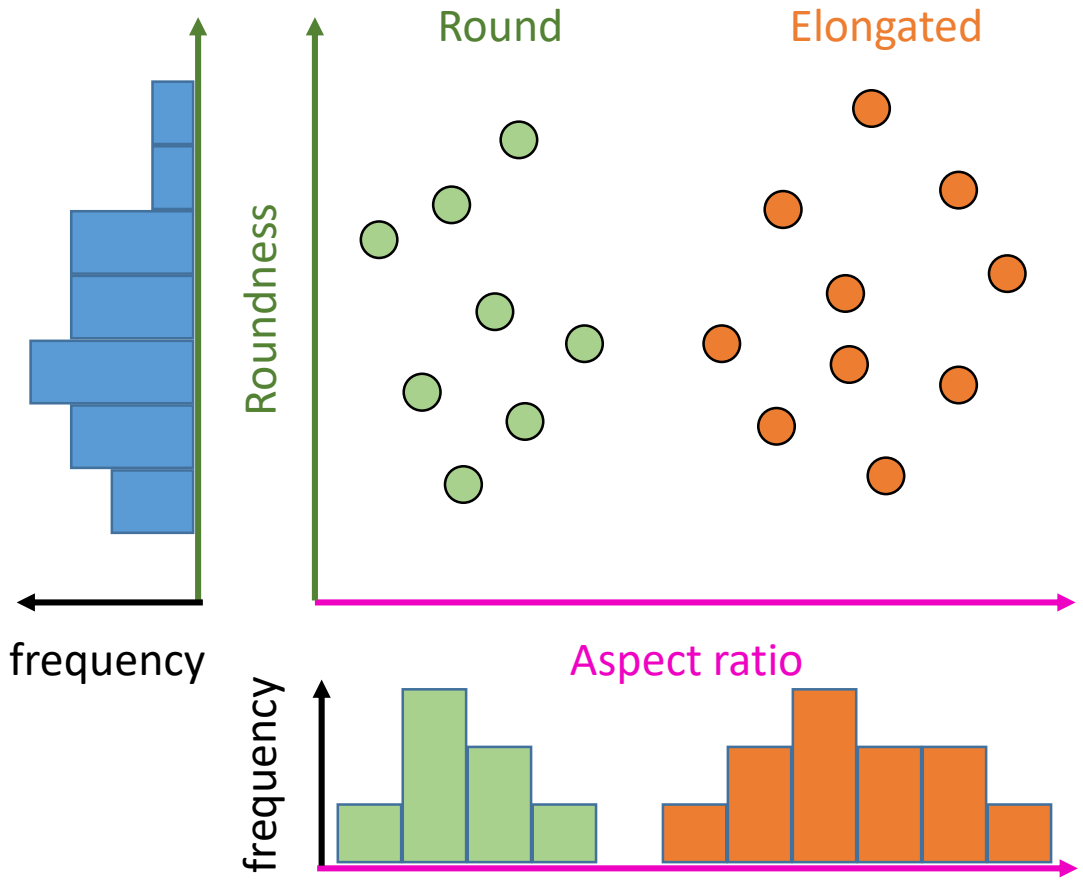
Unsupervised machine learning

- If you don't provide ground truth, the algorithm is *unsupervised*.



Unsupervised machine learning

- If you don’t provide ground truth, the algorithm is unsupervised.
- Nevertheless, algorithms can tell us something about the data



Hypothesis-driven quantitative biology

PoL Physics of Life TU Dresden

- Hypothesis: Cell shape can be influenced by modifying X.
- Null-Hypothesis: Circularity of modified cells is similar to cells in the control group.

- Sample preparation
- Imaging
- Cell segmentation
- Circularity measurement
- Statistics

Shall we use a different microscope?

Should we use a different segmentation algorithm?

Is circularity the right parameter to measure?

Hypothesis *generating* quantitative biology

PoL Physics of Life TU Dresden

- ~~Hypothesis: Cell shape can be influenced by modifying X.~~
- Question: Which image-derived parameter is influenced when modifying X?
  - Sample preparation
  - Imaging
  - Cell segmentation algorithm A, algorithm B, algorithm C
  - Measurement of circularity, solidity, elongation, extend, texture, intensity, topology …
  - Statistics

Which segmentation algorithms allow measurements that show a relationship with X?

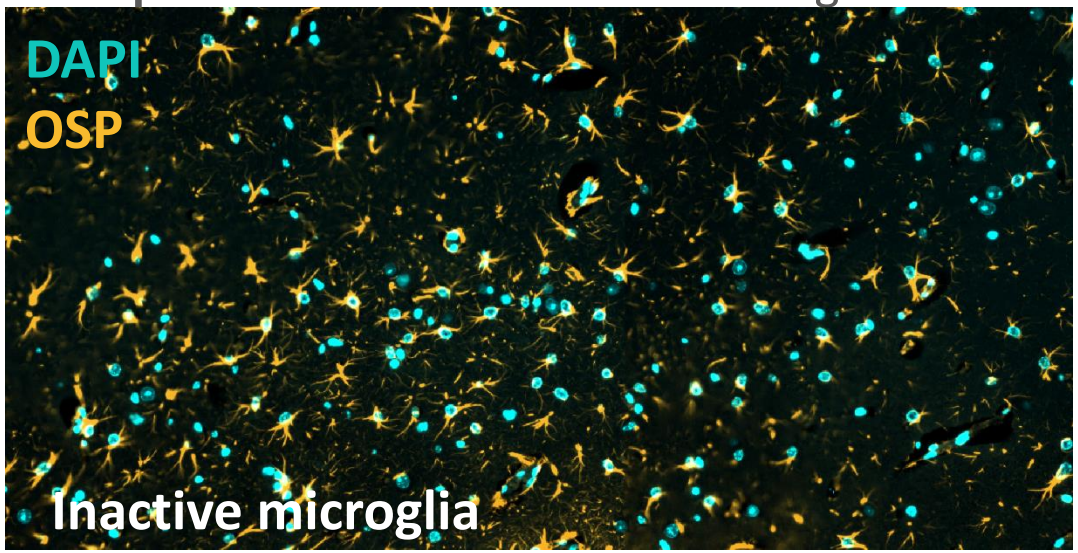
Why?

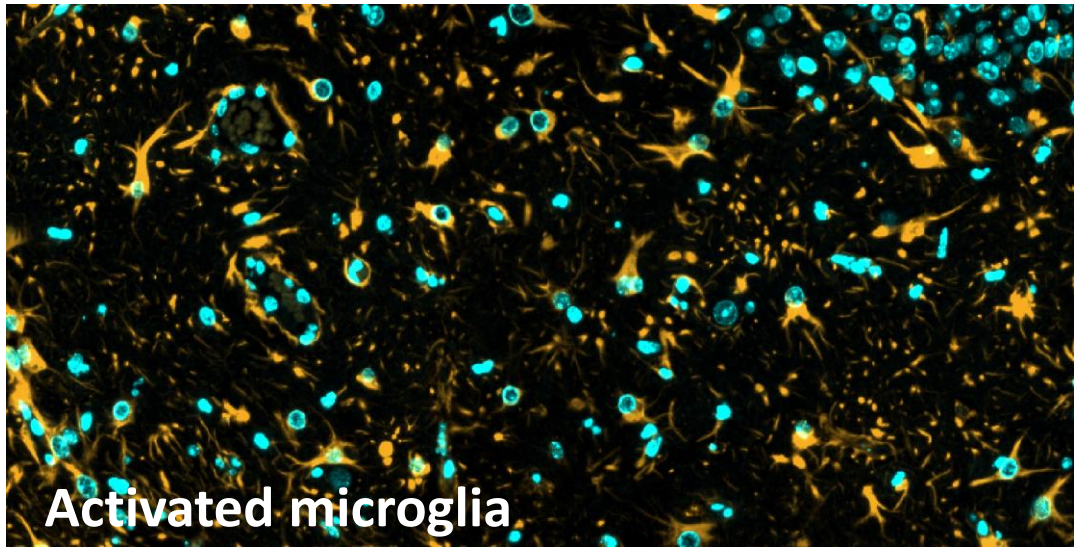
Which parameter shows any relationship with X?

Identifying features to measure

PoL Physics of Life TU Dresden

Example: Inactive vs. activated microglia in mouse brain





Challenge: Which of these features reflect biological propeties?

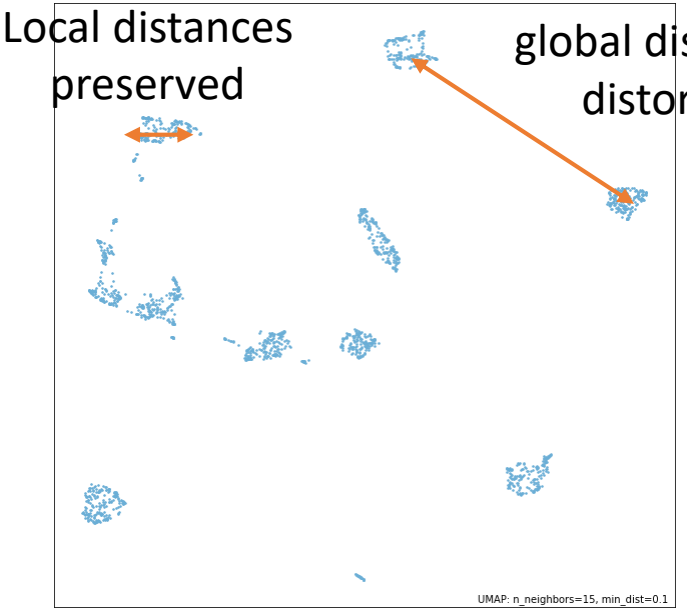
	label	area	bbox_area	convex_area	quivalent_diamete	max_intensity	mean_intensity	min_intensity	solidity	extent	eret_diameter_ma	local_centroid-0	l
1	1	3379	13949	5120	18.61786412639...	613.0	345.6717963894...	259.0	0.6599609375	0....	37.3496987939662	15.77952056821...	18
2	2	2319	7448	3491	16.42230229224...	421.0	297.8434670116...	240.0	0....	0....	38.65229618017...	4....	17
3	3	2304	14415	4281	16.38681751812...	456.0	300.8298611111...	245.0	0....	0....	34.19064199455...	17.73828125	13
4	4	3278	13804	5139	18.43048549951...	467.0	316.1446003660...	249.0	0....	0....	34.84250278036...	15.52287980475...	10
5	5	1501	3315	1681	14.20563625190...	458.0	302.147235176549	236.0	0....	0....	17.97220075561...	6....	6.
6	6	2341	6061	2714	16.47407088948...	594.0	355.4446817599...	261.0	0....	0....	30.67572330035...	16.54250320375...	6.
7	7	1725	3584	1940	14.87979081163...	568.0	343.7866666666...	257.0	0....	0....	17.72004514666...	7.80463768115942	7.
8	8	1502	3840	1753	14.20879025650...	431.0	290.0659121171...	235.0	0....	0....	18.57417562100...	8....	6.

Dimensionality reduction

- Challenge: Find a representation (embedding) of your data that represents the data in fewer dimensions
- Preserve local distances at the expense of global distortions

	label	area	bbox_area	convex_area	quivalent_diamete	max_intensity	mean_intensity	min_intensity	solidity	extent	eret_diameter_ma	local_centroid-0	l
1	1	3379	13949	5120	18.61786412639...	613.0	345.6717963894...	259.0	0.6599609375	0....	37.3496987939662	15.77952056821...	18
2	2	2319	7448	3491	16.42230229224...	421.0	297.8434670116...	240.0	0....	0....	38.65229618017...	4....	17
3	3	2304	14415	4281	16.38681751812...	456.0	300.8298611111...	245.0	0....	0....	34.19064199455...	17.73828125	13
4	4	3278	13804	5139	18.43048549951...	467.0	316.1446003660...	249.0	0....	0....	34.84250278036...	15.52287980475...	10
5	5	1501	3315	1681	14.20563625190...	458.0	302.147235176549	236.0	0....	0....	17.97220075561...	6....	6.
6	6	2341	6061	2714	16.47407088948...	594.0	355.4446817599...	261.0	0....	0....	30.67572330035...	16.54250320375...	6.
7	7	1725	3584	1940	14.87979081163...	568.0	343.7866666666...	257.0	0....	0....	17.72004514666...	7.804637681159 42	7.
8	8	1502	3840	1753	14.20879025650...	431.0	290.0659121171...	235.0	0....	0....	18.57417562100...	8....	6.
9	9	1602	4080	1894	14.51737058294...	475.0	297.8008739076...	241.0	0....	0....	18.70828693386...	8....	8.
10	10	1395	3600	1624	13.86304166283...	424.0	304.8494623655...	247.0	0....	0.3875	17.60681686165...	7....	7.
11	11	609	1100	697	10.51654029260...	323.0	274.2528735632...	241.0	0....	0....	13.45362404707...	3....	4.
12	12	1686	3757	1894	14.76679738567...	460.0	303.8303677342...	240.0	0....	0....	17.97220075561...	9....	7.
13	13	2157	5184	2531	16.03062694504...	576.0	339.990264255911	270.0	0....	0....	19.54482028569...	8....	8.
14	14	863	2340	1032	11.81237949737...	327.0	272.4449594438...	237.0	0....	0....	16.0312195418814	6....	5.

Many dimensions



Few dimensions

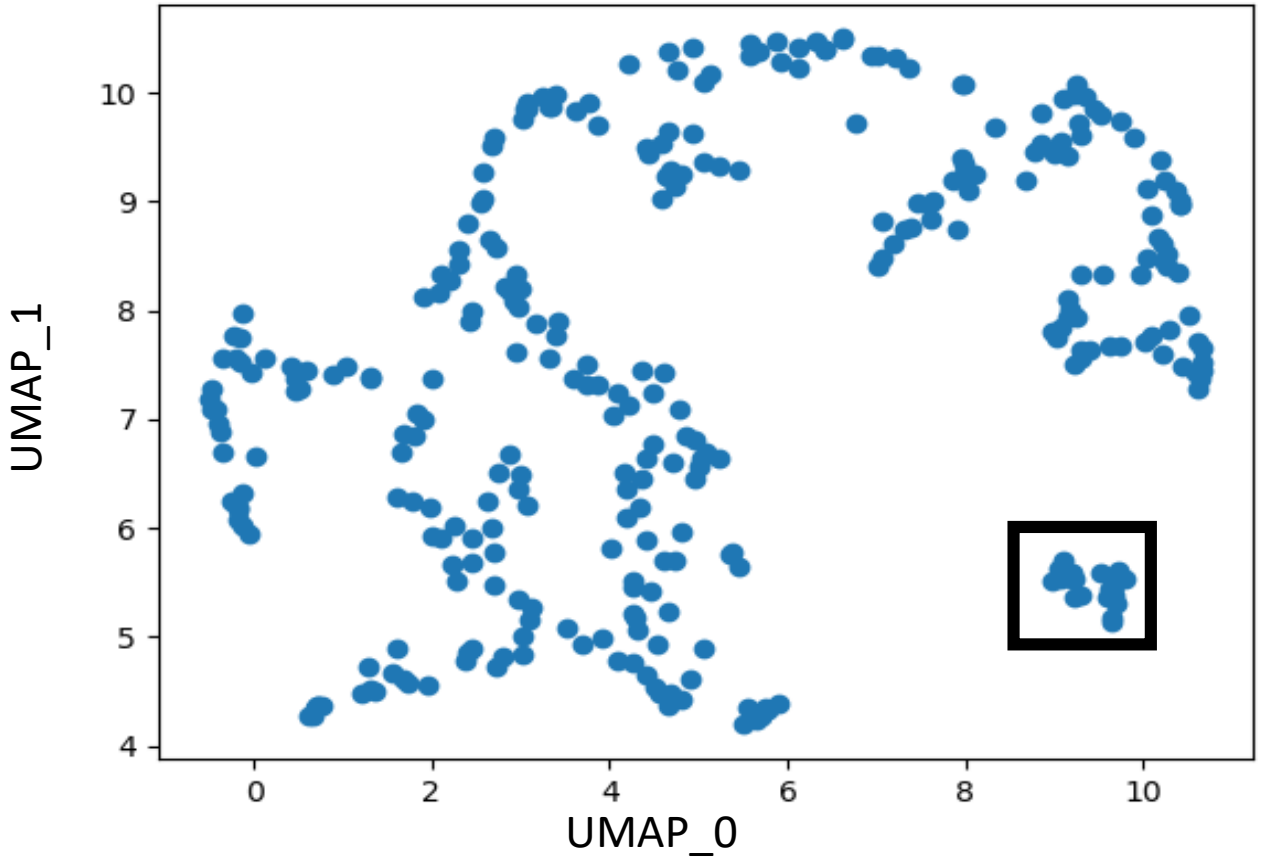
Learn more: https://umap-learn.readthedocs.io/en/latest/how_umap_works.html

UNIFORM MANIFOLD UMAP APPROXIMATION & PROJECTION

https://umap-learn.readthedocs.io/en/latest/index.html

Clustering

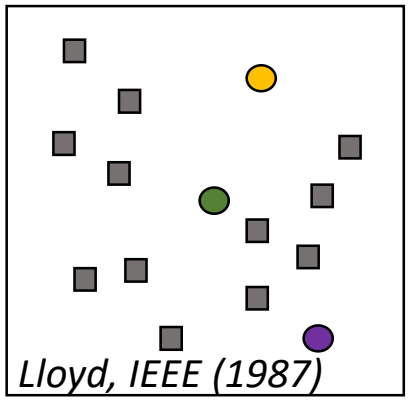
- **Starting point:** Feature space or dimensionality reduction reveals “groups” in our data
- **Can we automatically identify these groups?**



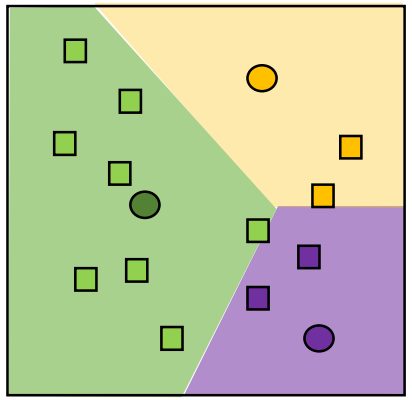
→ **Clustering** allows to stratify data into groups *without previous annotations*

K-means clustering

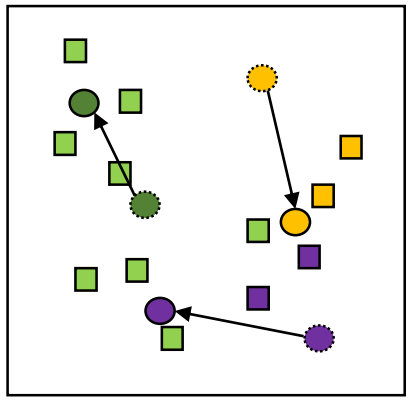
Strategy: Group data points into n groups so that variance within group is minimal



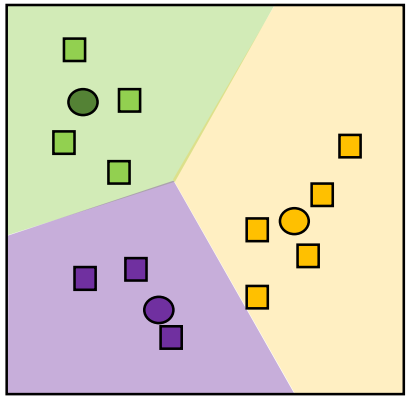
Step1: Random initialization of cluster centers



Step2: Tessellation of space into cluster regions



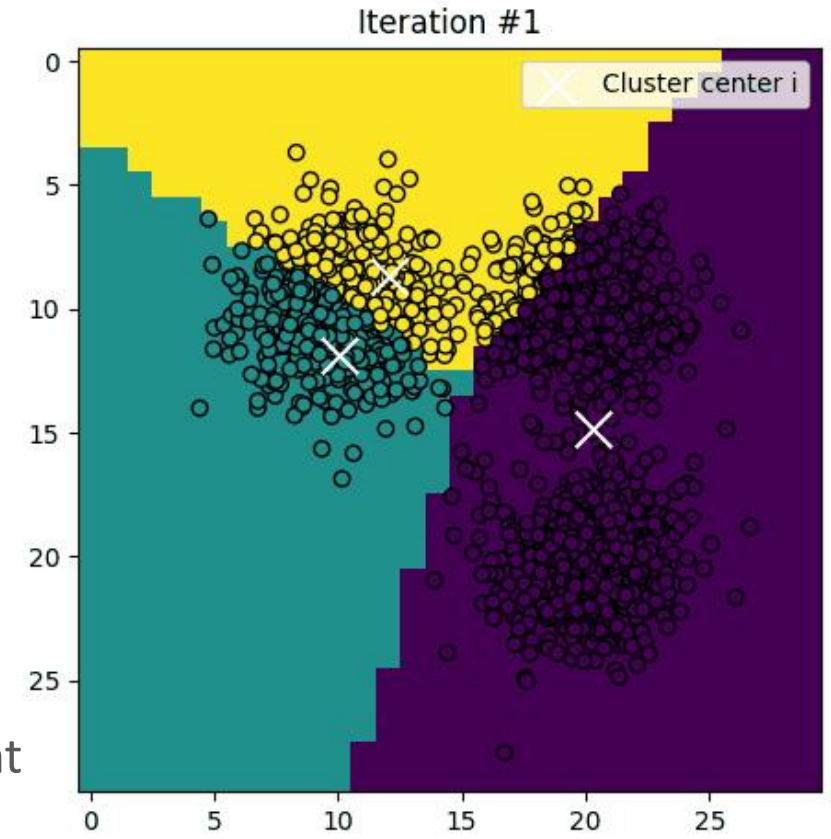
Step3: Replace cluster center with centroids



Step4: Repeat 2&3 until convergence

Strength and weaknesses

- Number of clusters needs to be known
- Clusters can not capture more complex topologies
- Based on Euclidian metrics → every new point can be assigned to a cluster



PoL
Physics of Life
TU Dresden

CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

Napari-clusters-plotter plugin

@zoccolermarcelo

Dimensionality reduction

Uniform manifold approximation and projection (UMAP)

t-distributed stochastic neighbor embedding (t-SNE)

Principal component analysis (PCA)

Laura Žigutytė
@zigutyte

Ryan Savill
@RyanSavill4

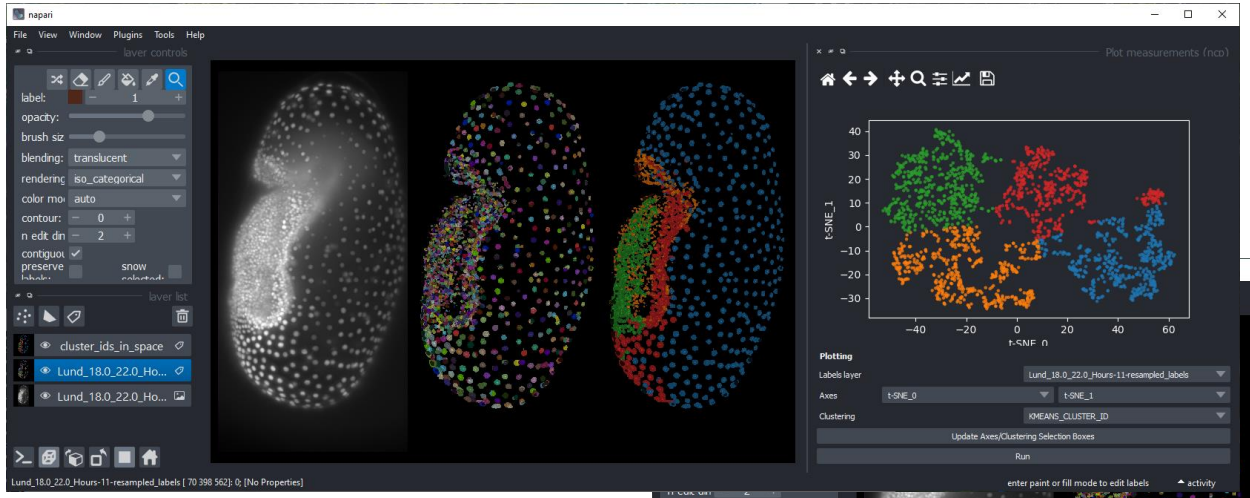
Marcelo L. Zoccoler
Robert Haase,
Thorsten Wagner,
Johannes Soltwedel

@zoccolermarcelo

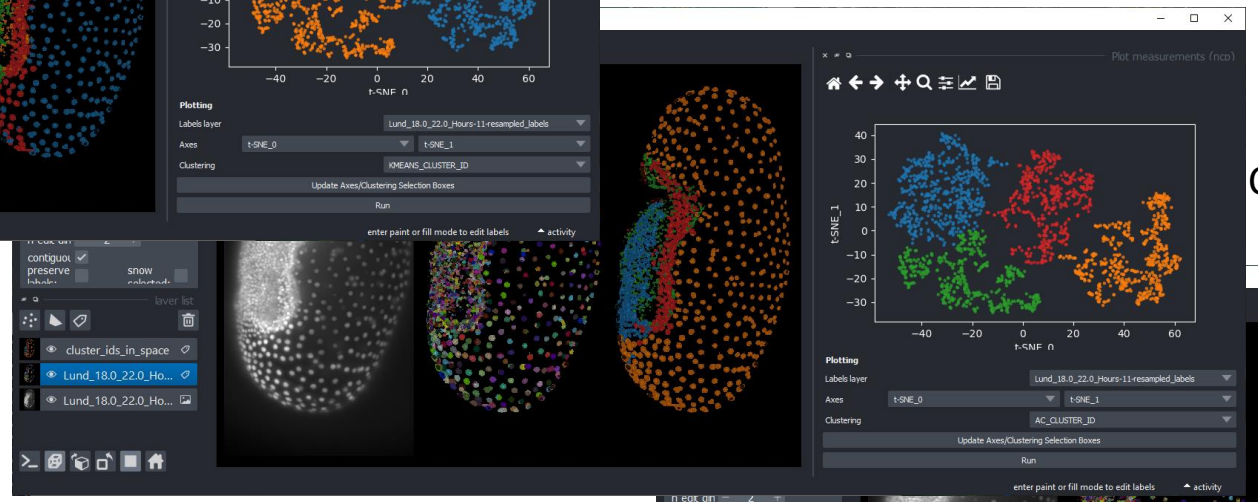
https://github.com/BiAPoL/napari-clusters-plotter

Image Data Source: Daniela Vorkel, Myers lab, MPI-CBG/CSBD Dresden

Clustering

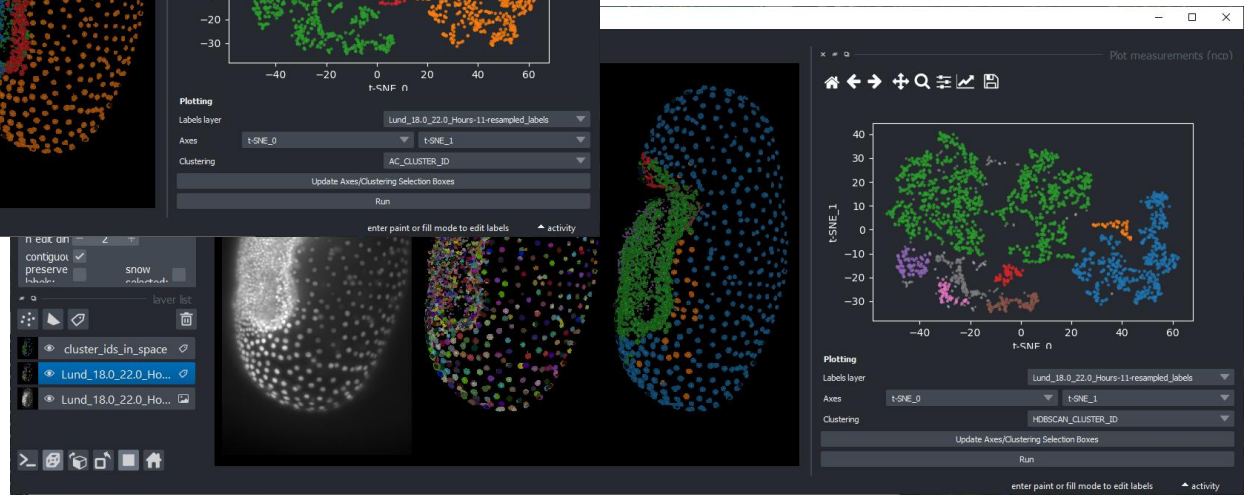


K-means clustering



Agglomerative clustering

Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN)



Laura Žigutytė
@zigutyte

Ryan Savill
@RyanSavill4

Marcelo L. Zoccoler
Robert Haase,
Thorsten Wagner,
Johannes Soltwedel

https://github.com/BiAPoL/napari-clusters-plotter

Image Data Source: Daniela Vorkel, Myers lab, MPI-CBG/CSBD Dresden

Data Exploration

- … using interactive clustering

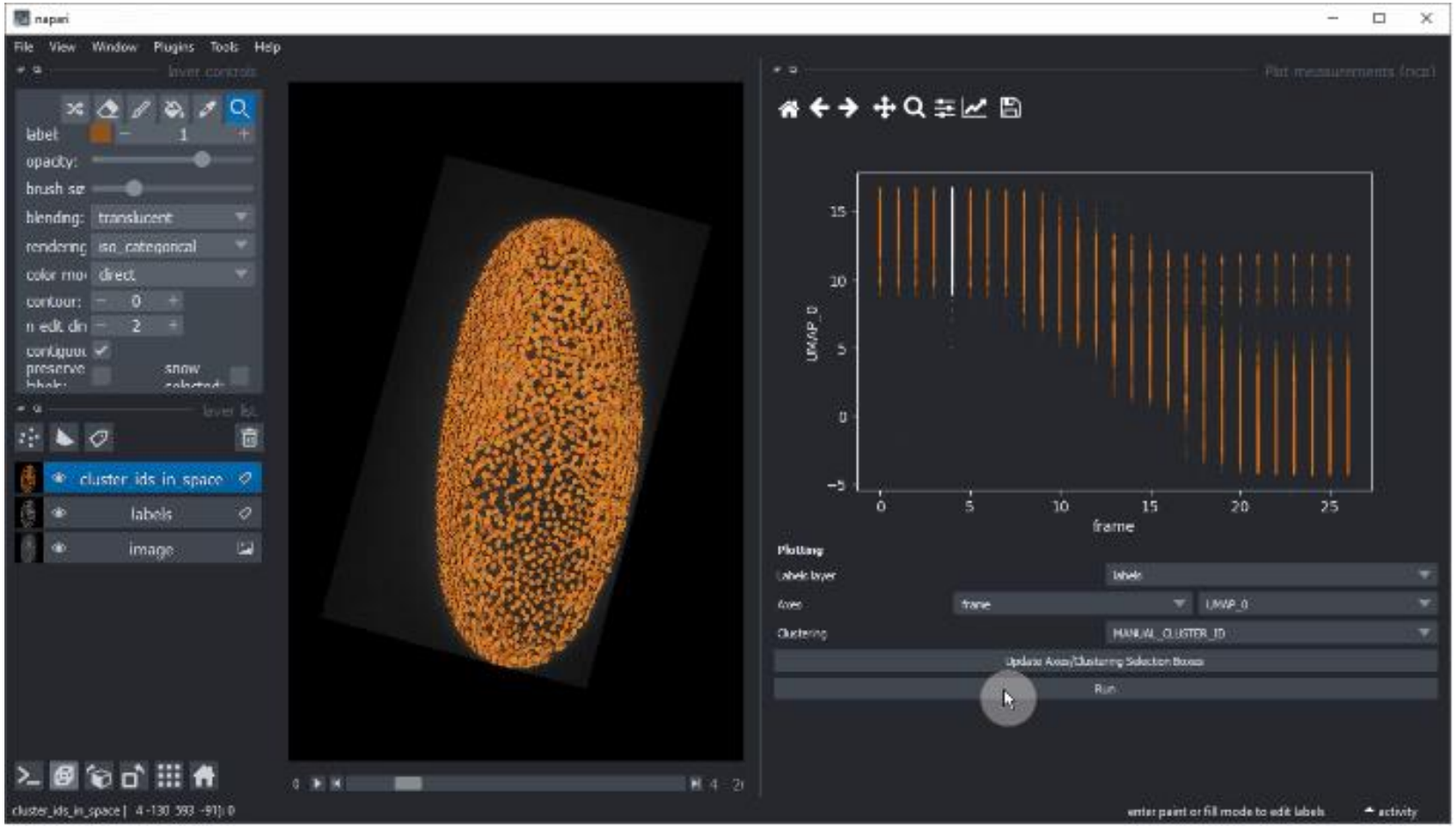


Image data source: Daniela Vorkel, Myers lab, MPI-CBG/CSBD

Laura Žigutytė @zigutyte

Ryan Savill @RyanSavill4

Marcelo L. Zoccoler
Robert Haase,
Thorsten Wagner,
Johannes Soltwedel

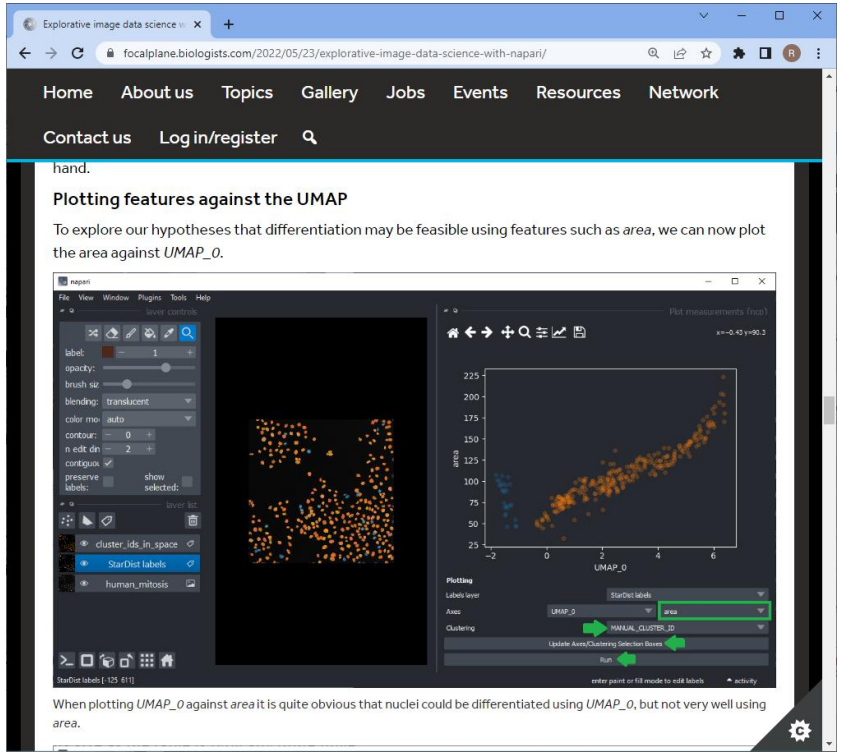
@zoccolermarcelo

https://github.com/BiAPoL/napari-clusters-plotter

Image Data Source: Daniela Vorkel, Myers lab, MPI-CBG/CSBD Dresden

Further reading

- Blog posts on the Focalplane about Unsupervised Machine Learning



https://focalplane.biologists.com/2022/05/23/explorative-image-data-science-with-napari/

Exercise

Unsupervised Obejct Classification

- Use Napari to classify objects (nuclei) without ground-truth

Interactive unsupervised object classification in Napari

In this exercise we will perfom Feature Extraction, Dimensionality Reduction with UMAP and Clustering with HDBSCAN to assign objects (nuclei) to different classes in napari. We will use the napari plugin napari-clusters-plotter plugin.

Getting started

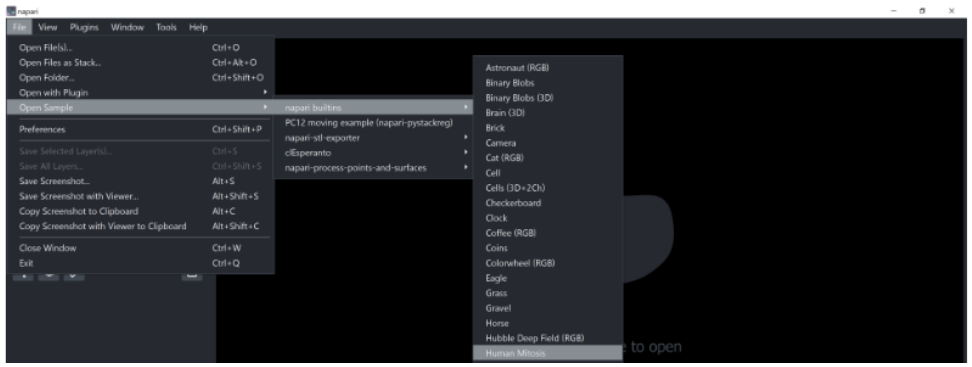
Open a terminal window and activate your conda environment:

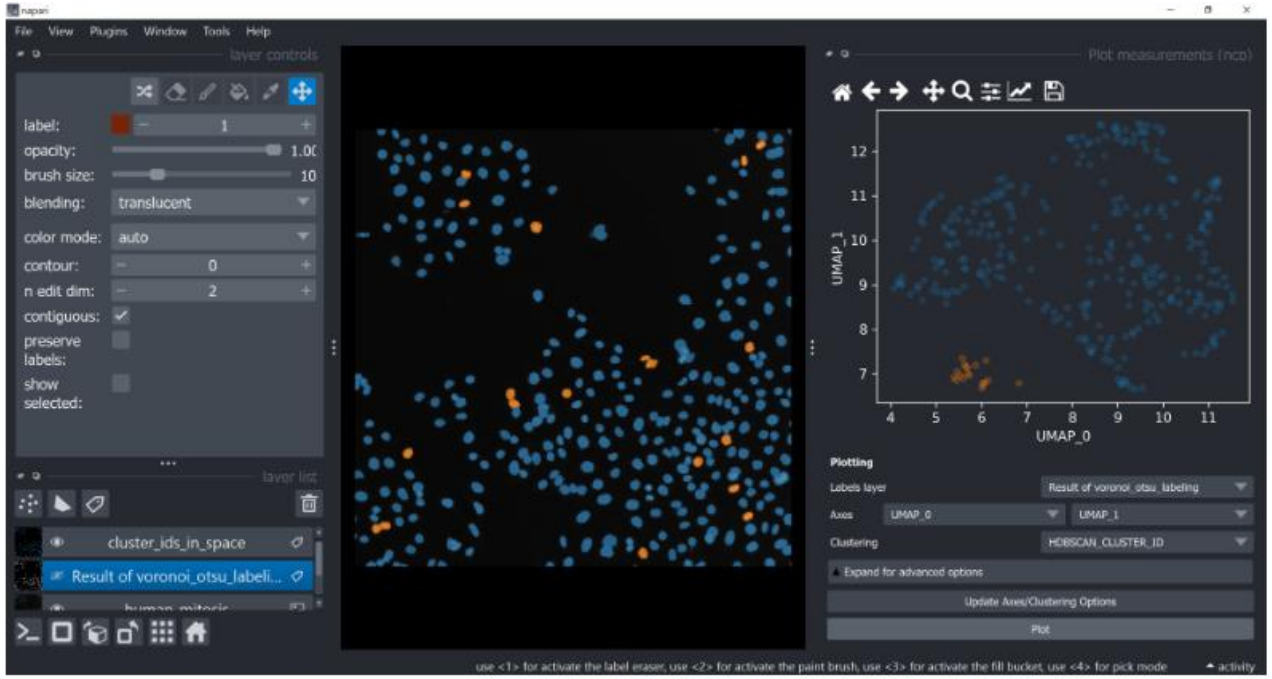
```
mamba activate devbio-napari
```

Afterwards, start up Napari:

```
napari
```

Load the "human-mitosis" example dataset from the menu `File > Open Sample > napari builtins > Human Mitosis`





https://zoccoler.github.io/QM_Course_Bio_Image_Analysis_with_napari_2024/Machine_Learning_with_napari/interactive_unsupervised_object_classification/readme.html

PoL
Physics of Life
TU Dresden

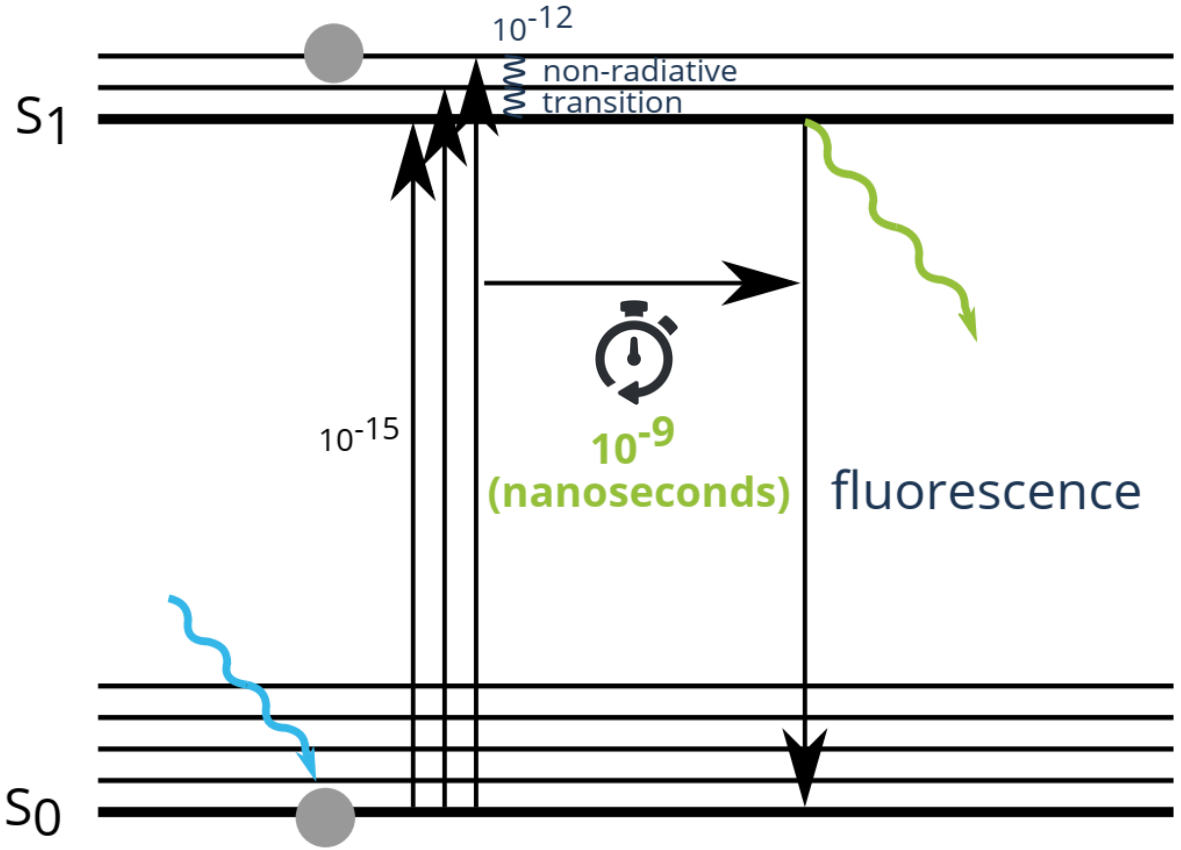
CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

Fluorescence Lifetime Imaging Microscopy

@zoccolermarcelo

FLIM: Fluorescence LIFETIME Imaging Microscopy

FLIM measures the precise timing of fluorescence



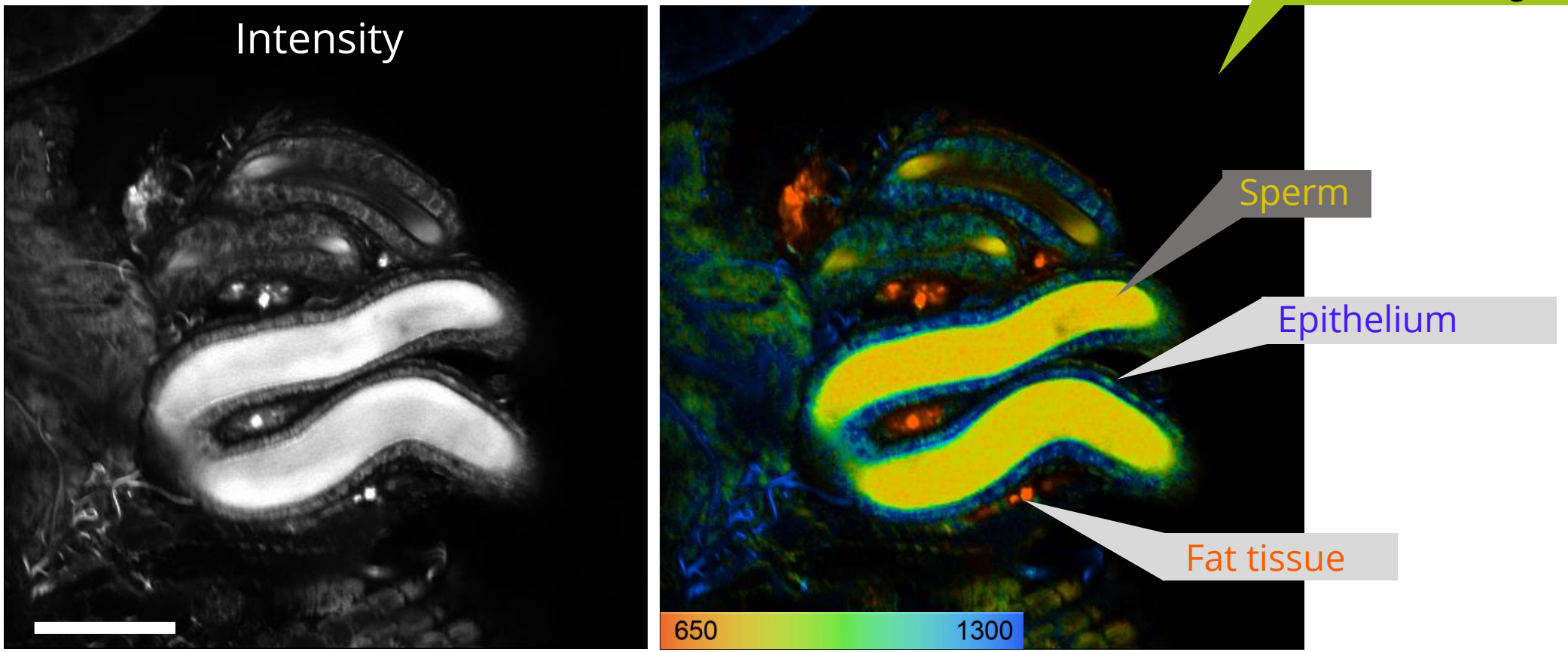
The power of Fluorescence LIFETIME Imaging Microscopy (FLIM)

PoL Physics of Life TU Dresden

Probe cellular states:

Metabolism using intrinsic NAD(P)H

NAD(P)H autofluorescence



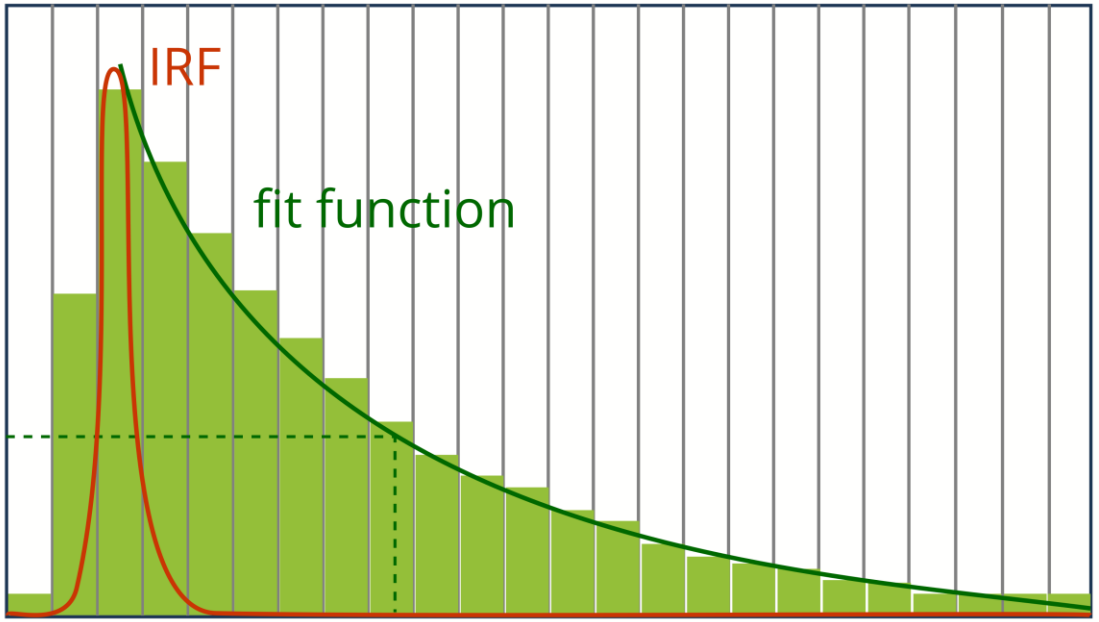
- Seminal receptable of *Drosophila melanogaster*

An additional dimension of contrast = information

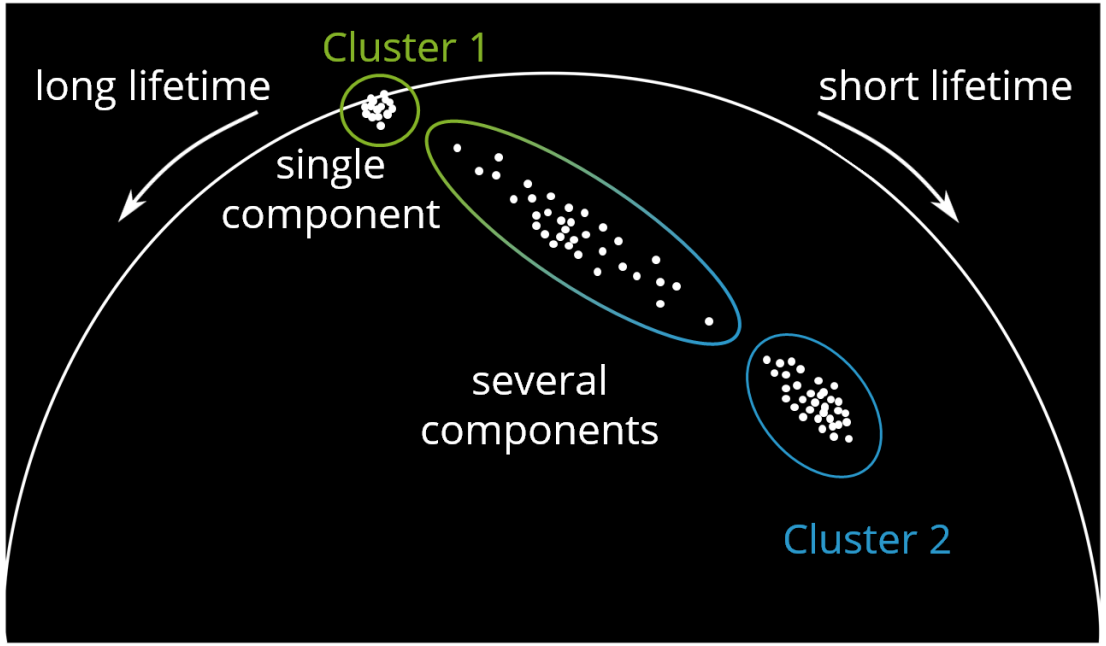
FLIM Data Analysis

PoL Physics of Life TU Dresden

Curve fitting



Phasor plot



PoL
Physics of Life
TU Dresden

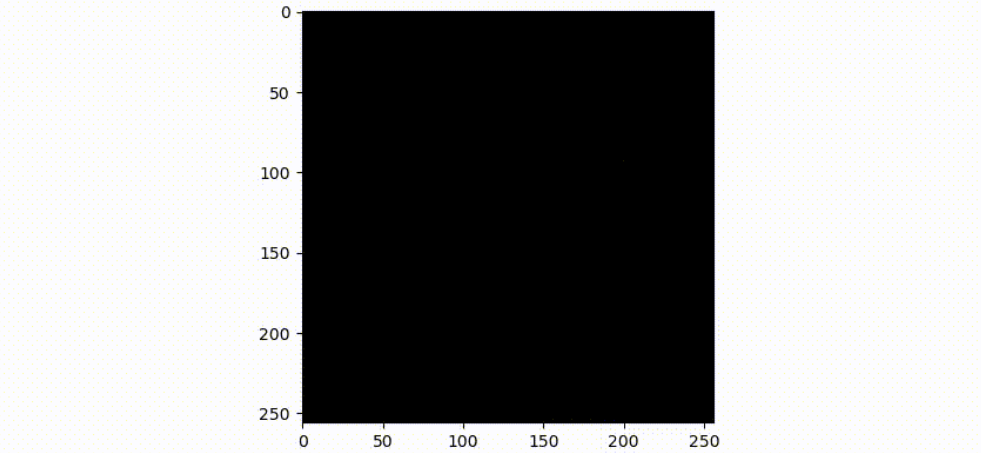
CENTER FOR
SYSTEMS BIOLOGY
DRESDEN

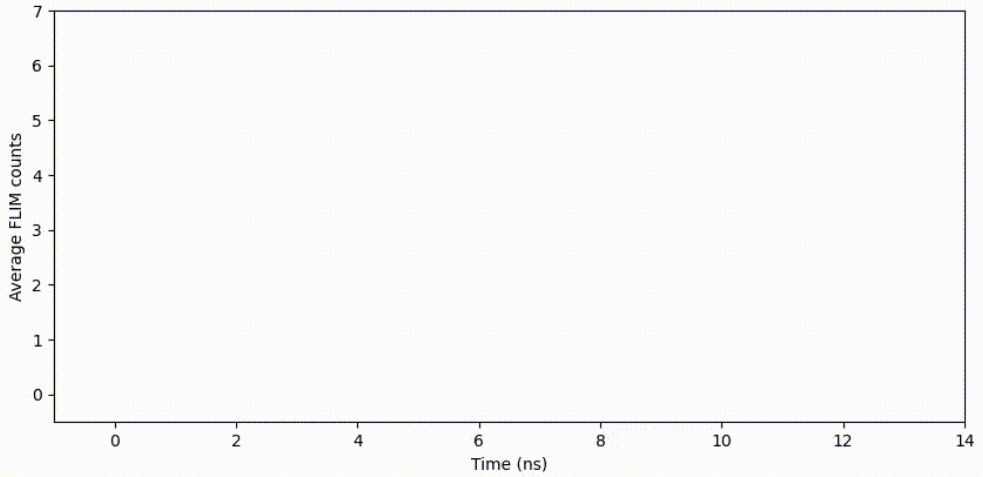
Napari-flim-phasor-plotter plugin

@zoccolermarcelo

Visualizing FLIM raw data in napari

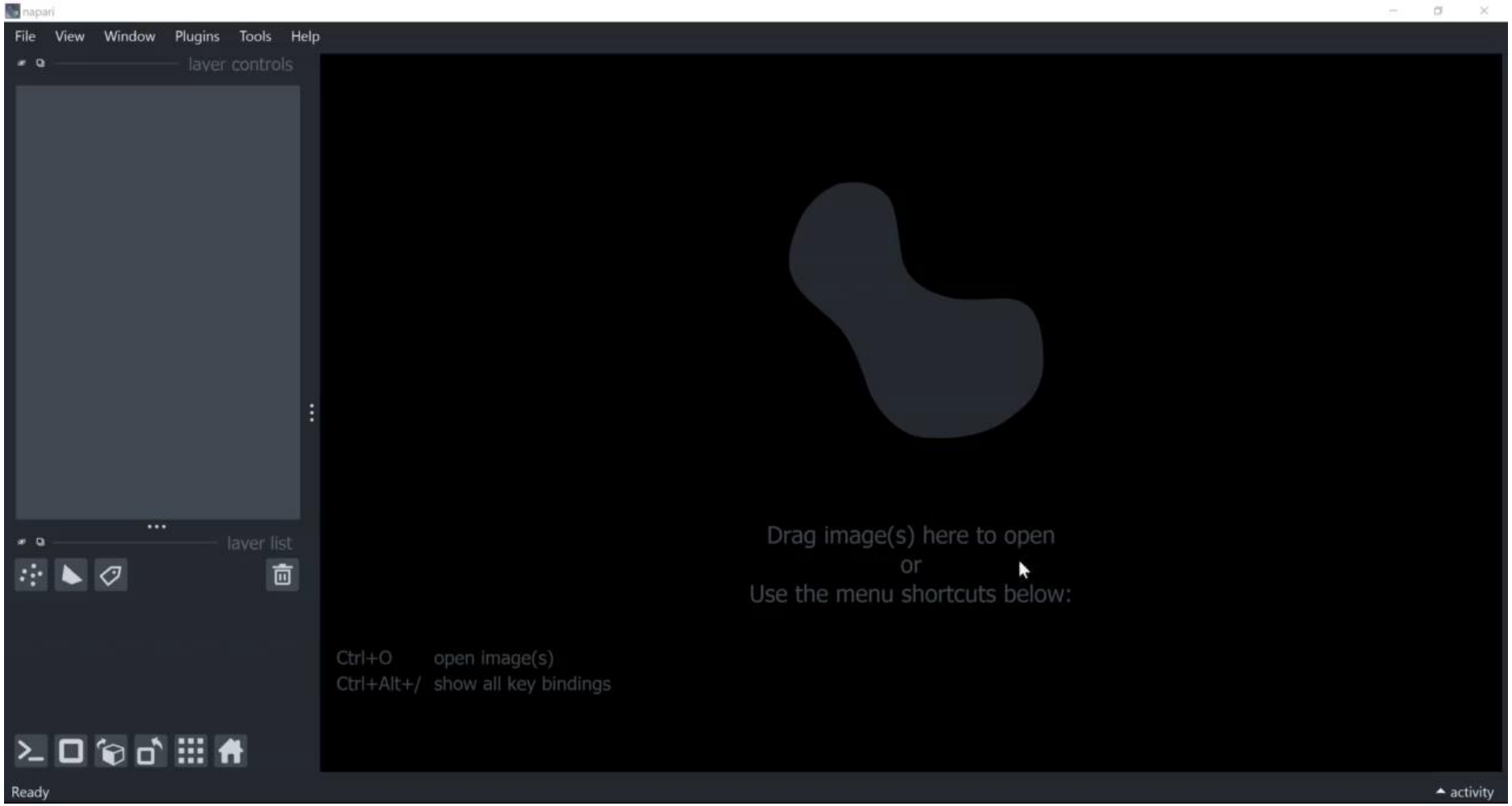
- Hazelnut example





Plugin demonstration with Synthetic Data

- Lifetime Cat

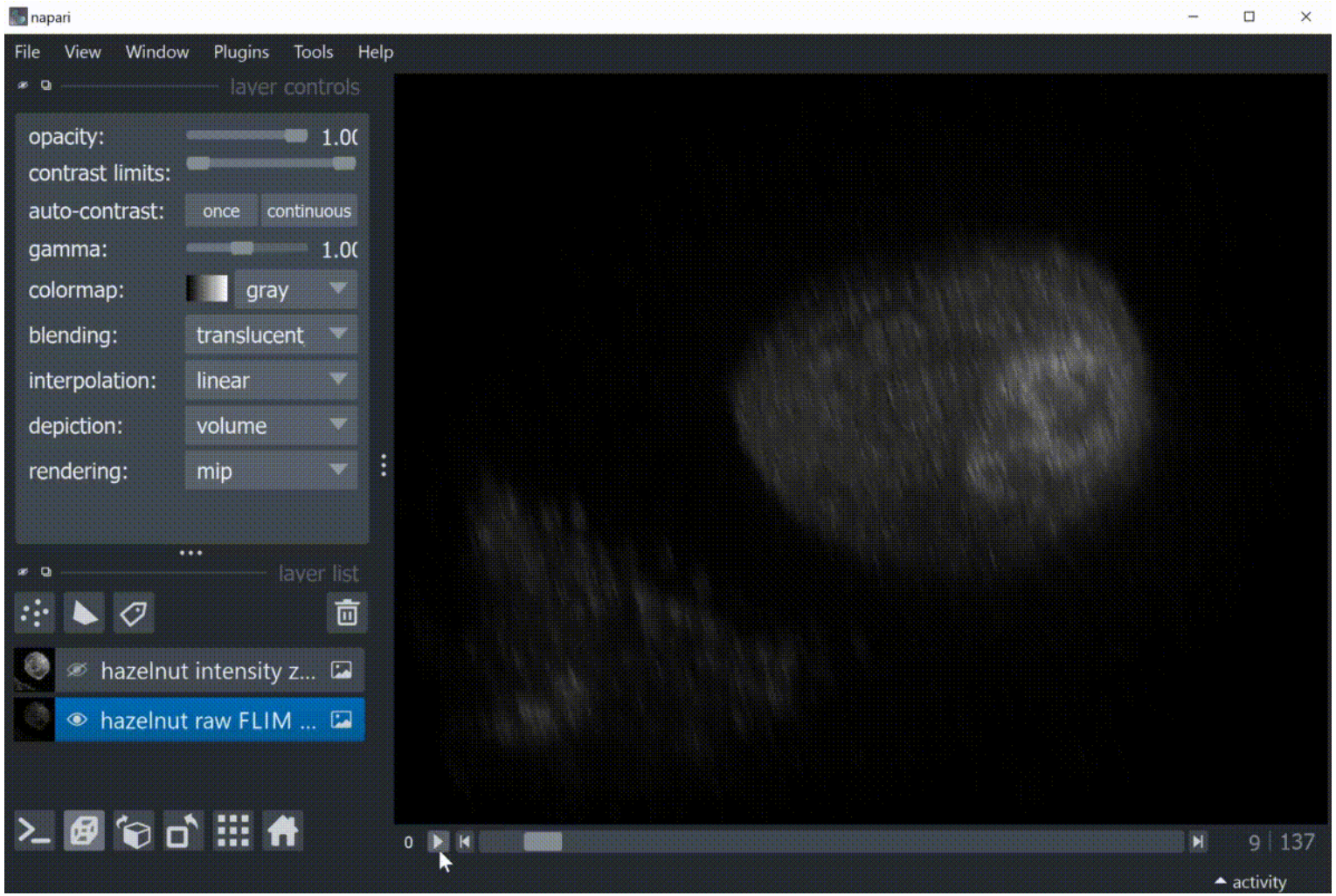


Data created by Svetlana Iarovenko (IMP Viena)

https://github.com/zoccoler/napari-flim-phasor-plotter

FLIM z-stacks

- Hazelnut 3D example



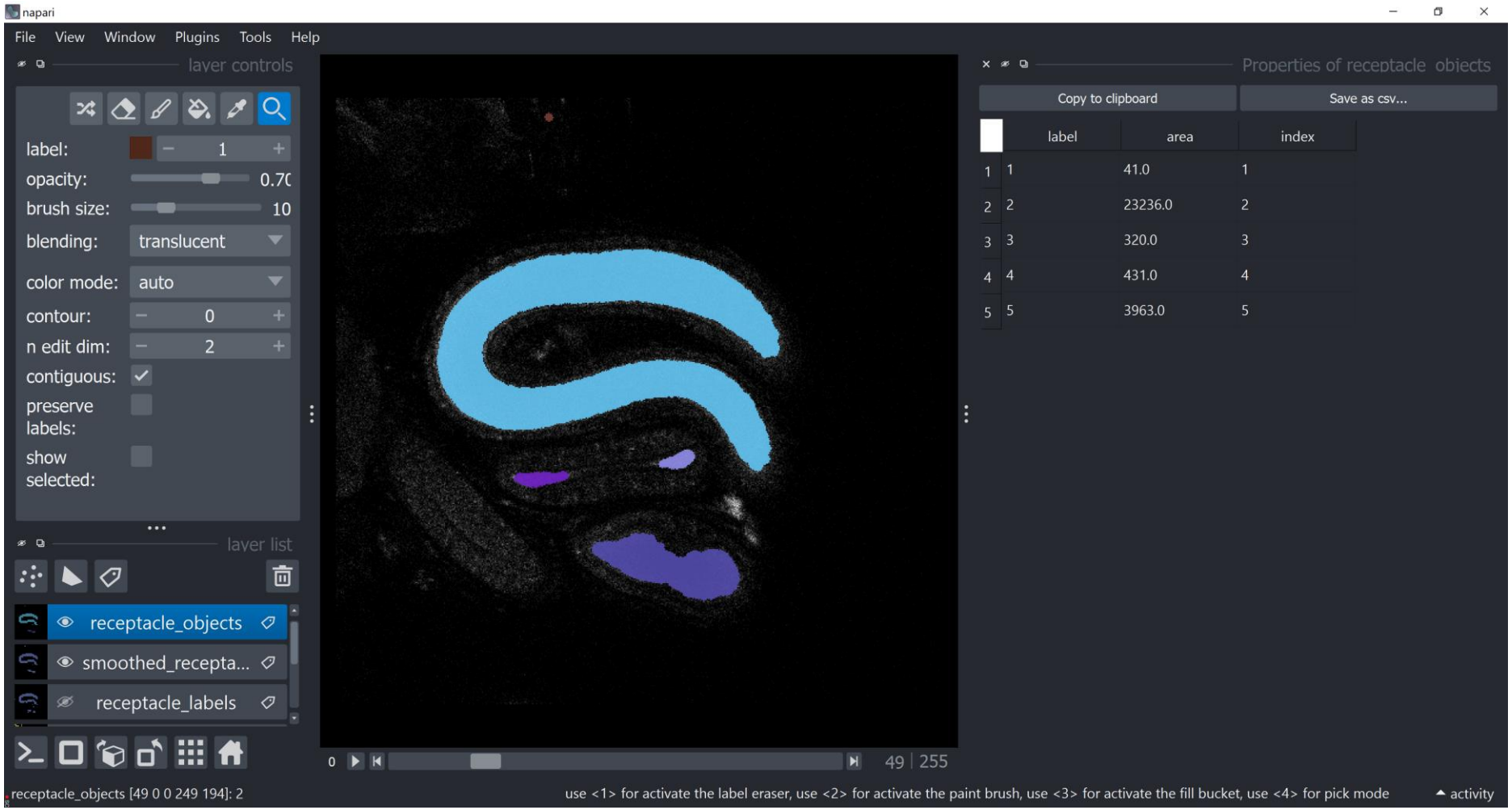
https://github.com/zoccoler/napari-flim-phasor-plotter

Integrating FLIM analysis into a bio-image analysis workflow

- Seminal receptacle example

Example workflow (under development):

- Generate a Phasor Plot
- Change Phasor Plot display options
- Apply a clustering algorithm to the Phasor Plot (HDBSCAN)
- Extract cluster/class of interest
- Post-processing: apply morphological operations
- Perform instance segmentation
- Extract Features

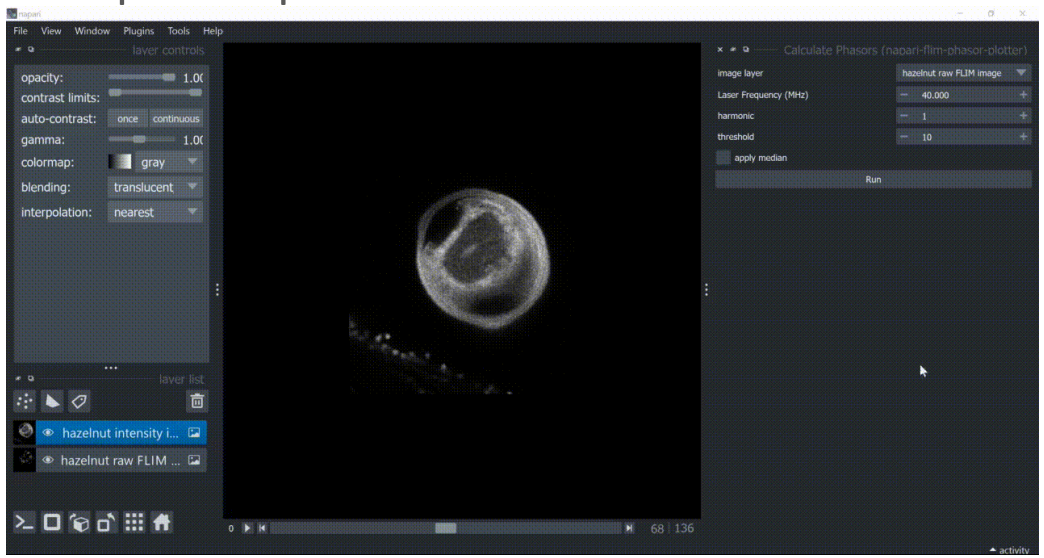


https://github.com/zoccoler/napari-flim-phasor-plotter

Exercise

Segment another example image with the phasor plot

- Open the hazelnut 2D, hazelnut 3D or the synthetic lifetime cat sample data
- Calculate the phasor plot
- Select clusters and check the segmentation results
- Re-calculate the phasor plot exploring other parameters (median, harmonic, threshold) to improve clusters display in the phasor plot.



https://zoccoler.github.io/QM_Course_Bio_Image_Analysis_with_napari_2024/Machine_Learning_with_napari/FLIM/readme.html

Acknowledgements

PoL Physics of Life TU Dresden

BiAPoL team

- Marcelo Zoccoler
- Johannes Soltwedel
- Maleeha Hassan
- Stefan Hahmann
- Somashekhar Kulkarni

Former lab members:

- Robert Haase
- Allyson Ryan
- Till Korten
- Mara Lampert
- Svetlana Iarovenko
- Ryan George Savill
- Laura Zigutyte

Networks

BioDIP Biopolis Dresden Imaging Platform

CENTER FOR SYSTEMS BIOLOGY DRESDEN

image.sc

NFDI4 BIOIMAGE

Funding

DFG public

Chan Zuckerberg Initiative