

Advanced Methods and Human Cell Technologies (AMHCT)
RegBioMed Master's Course

Navigating the Reproducibility Storm with Bio-Image Analysis

Dr. Marcelo Leomil Zoccoler

Bio-Image Analysis Technology Development Group - Physics of Life (PoL) – TU Dresden

Re-using material from:

Robert Haase (Scads.AI Leipzig); Maleeha Hassan (Helmholtz AI Dresden); Johannes Soltwedel (PoL – TU Dresden);

Table of Contents

- Installation and introduction to **napari**
- Introduction to **OMERO** and **napari-omero**
- Segmentation with Machine Learning using **napari-apoc**
- Quick view of Segmentation with Machine Learning using **micro-sam**
- Feature Extraction with **napari-skimage-regionprops**
- Object Classification with **napari-apoc** and **napari-clusters-plotter**
- **Exercises:**
 - Instance Segmentation with **napari-apoc**
 - Creating a workflow with **napari-omero** and **napari-apoc**
 - Reproducing a workflow
- **Discussion**



https://biapol.github.io/AMHCT_Bio_Image_Analysis_2025/intro.html

napari Installation

- napari as a Python Library
- napari as a Bundle App
- Installing Plugins

Installing napari as a Python package

napari is a Python library

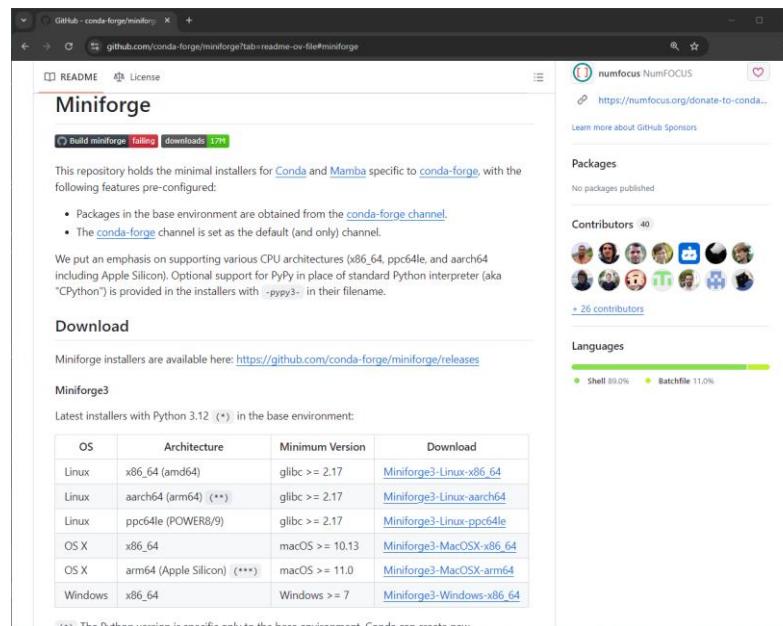
The recommended way to install napari is as a Python package

<https://napari.org/stable/tutorials/fundamentals/installation.html#install-as-python-package-recommended>

Your computer needs Python to run a Python package

— How do I install Python?

Install Miniforge!



<https://github.com/conda-forge/miniforge?tab=readme-ov-file#miniforge>

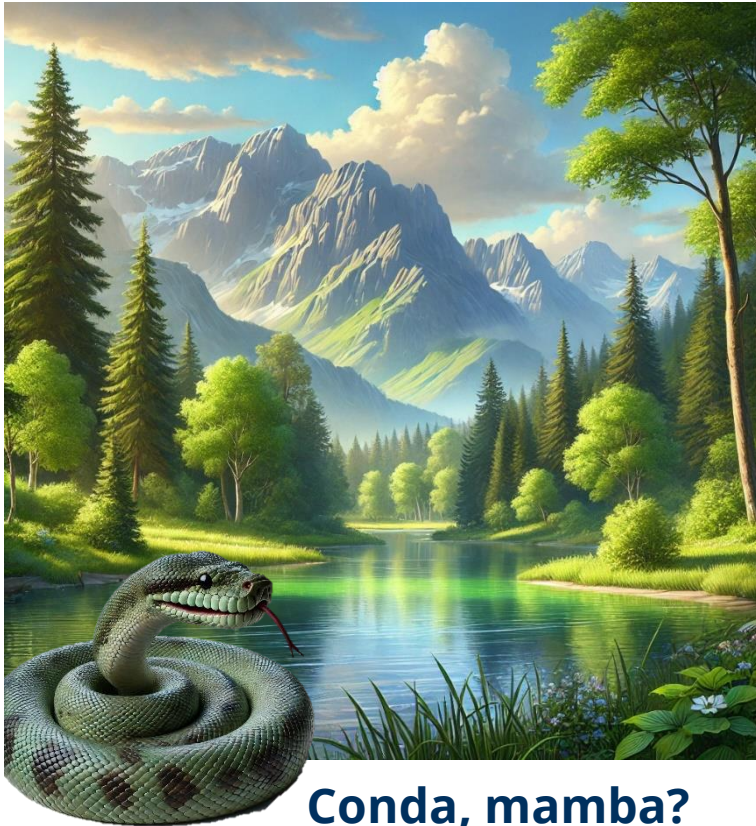
- Miniforge is a minimal installer for `conda` and `mamba` (efficient conda in C++)
- `conda` is an environment and a package manager
- `conda` can create Virtual Enviroments and install Python (and other) packages, including napari

https://hackmd.io/@talley/SJB_IObBi#Terms

Installing napari as a Python package



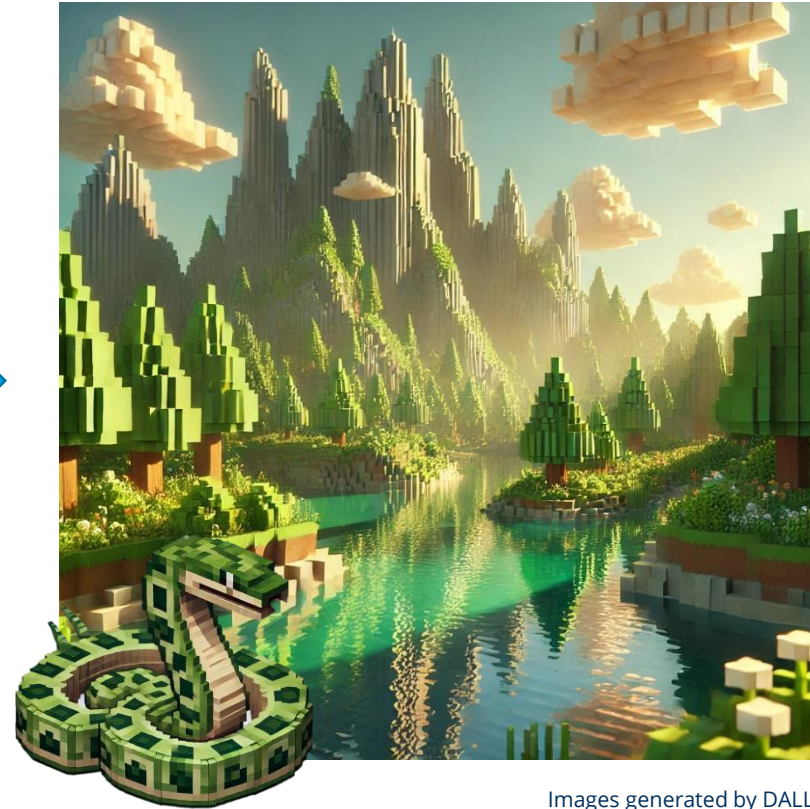
Wait, **environments** ?



Conda, mamba?



Yes, virtual environments!



Images generated by DALL-E, developed by OpenAI.

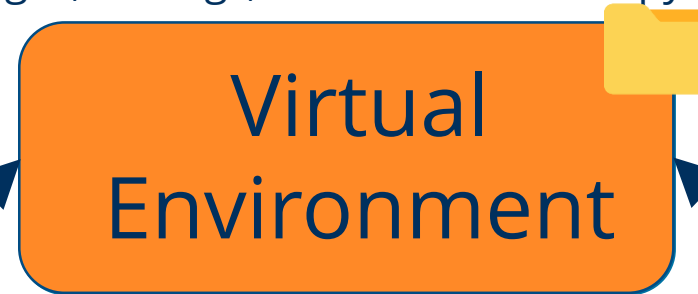
Smiley icons from Flaticon.com

Installing napari as a Python package

Install it as a Python package in a Virtual Environment

— A virtual environment is an isolated collection of packages, settings, and an associated python interpreter

```
Miniforge Prompt
(base) C:\Users\mazo260d>conda activate napari-intro-env
(napari-intro-env) C:\Users\mazo260d>
```



- python 3.10.8
- numpy 1.23.5
- pandas 2.0.3
- **napari 0.5.1**
- **pyqt 5.15.9**
- ...

create
activate

in

install

Miniforge

conda/mamba

napari

Commands:

name

channel

package (version)



mamba **create** -y -n napari-intro-env -c conda-forge python=3.10

mamba **activate** napari-intro-env

mamba **install** -c conda-forge napari pyqt

Some packages can only be installed with `pip`
`conda` checks for version compatibility before installing

- `pip` tries to install, then lets you know if something went wrong
- `pip` cannot create environments

Installing napari

Napari can also be downloaded and installed like other software

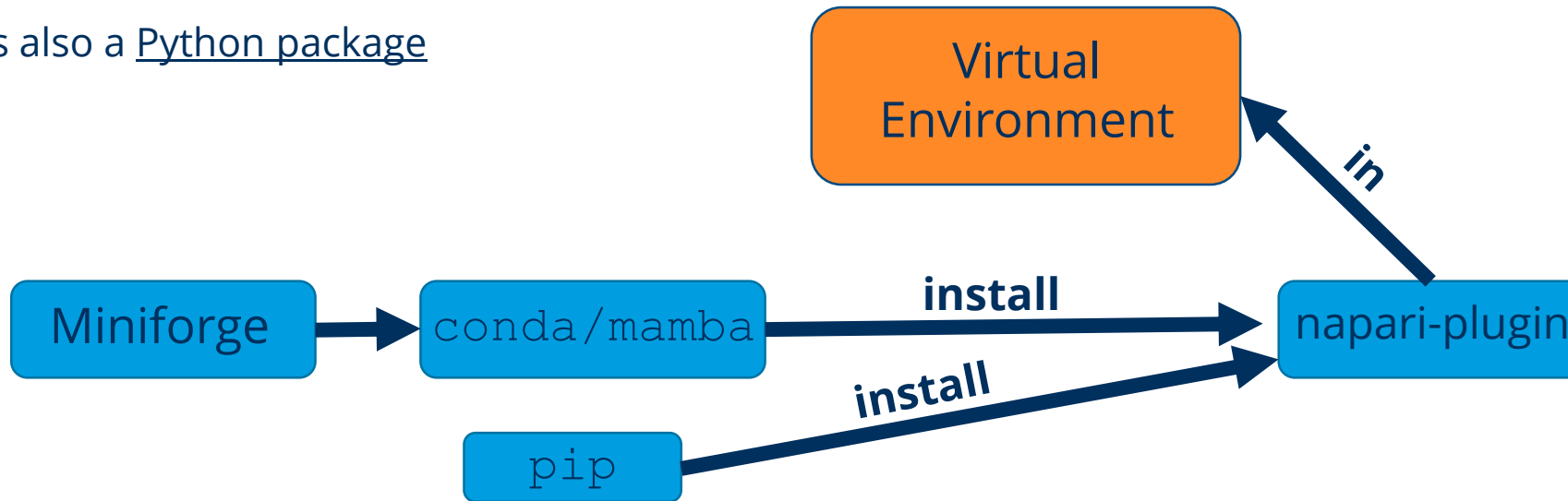
- https://napari.org/stable/tutorials/fundamentals/installation_bundle_conda.html#how-to-install-napari-as-a-bundled-app

This is more convenient, however it may be difficult to manage different plugin versions mid-long term

- In case of a dependency version mismatch, it may be necessary to uninstall and re-install the software and all the plugins again

Installing a plugin

A napari plugin is also a Python package



Commands:

mamba activate `napari-intro-env`

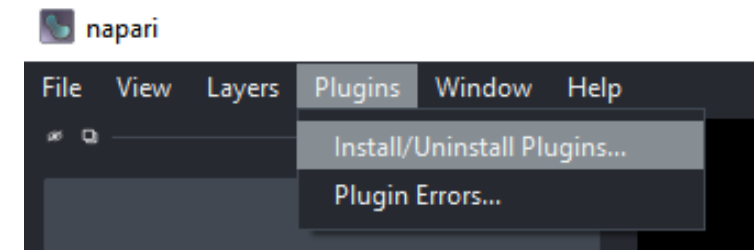
mamba install `napari-plugin` **OR** **pip** install `napari-plugin`



conda install `napari-plugin`

Always check plugin
installation instructions!

Via Plugins Menu:



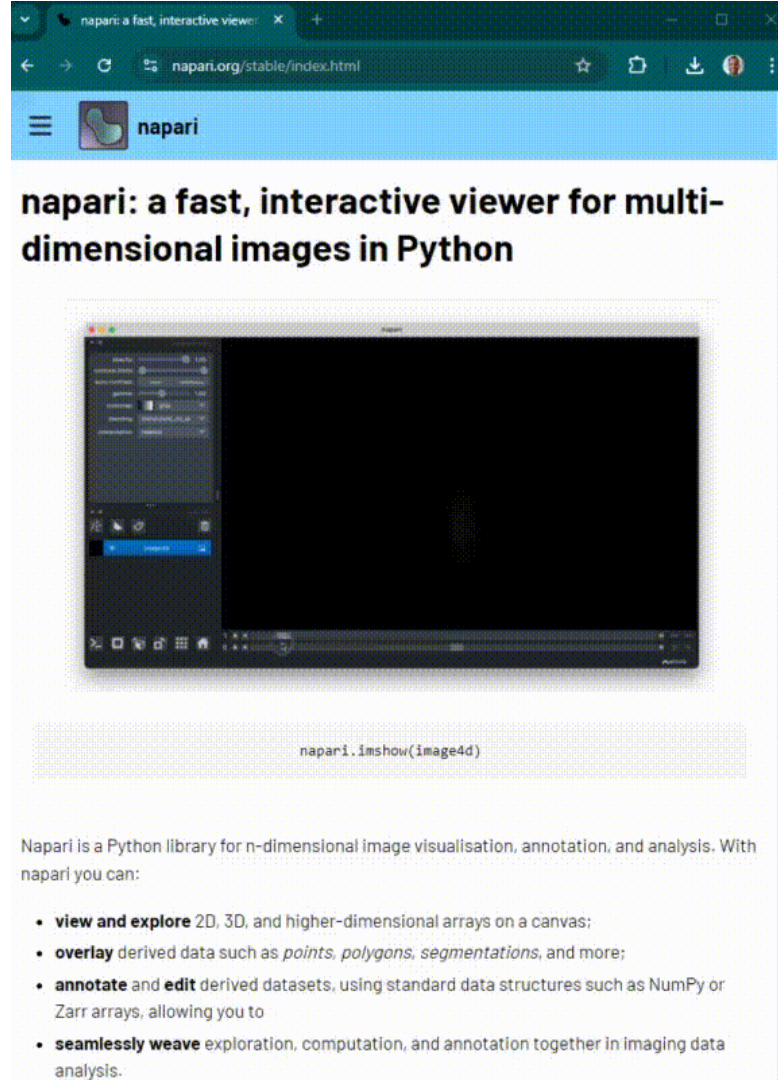
Introduction to napari

- napari Viewer
- Layer Types
- napari Plugins

Napari: 3D viewer for Python

Multi-dimensional image viewer in python

<https://napari.org/>



Napari: 3D viewer for Python



<https://napari.org/>

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

Napari user interface

Menus

View configuration /
tools

`layer.opacity = 0.5`

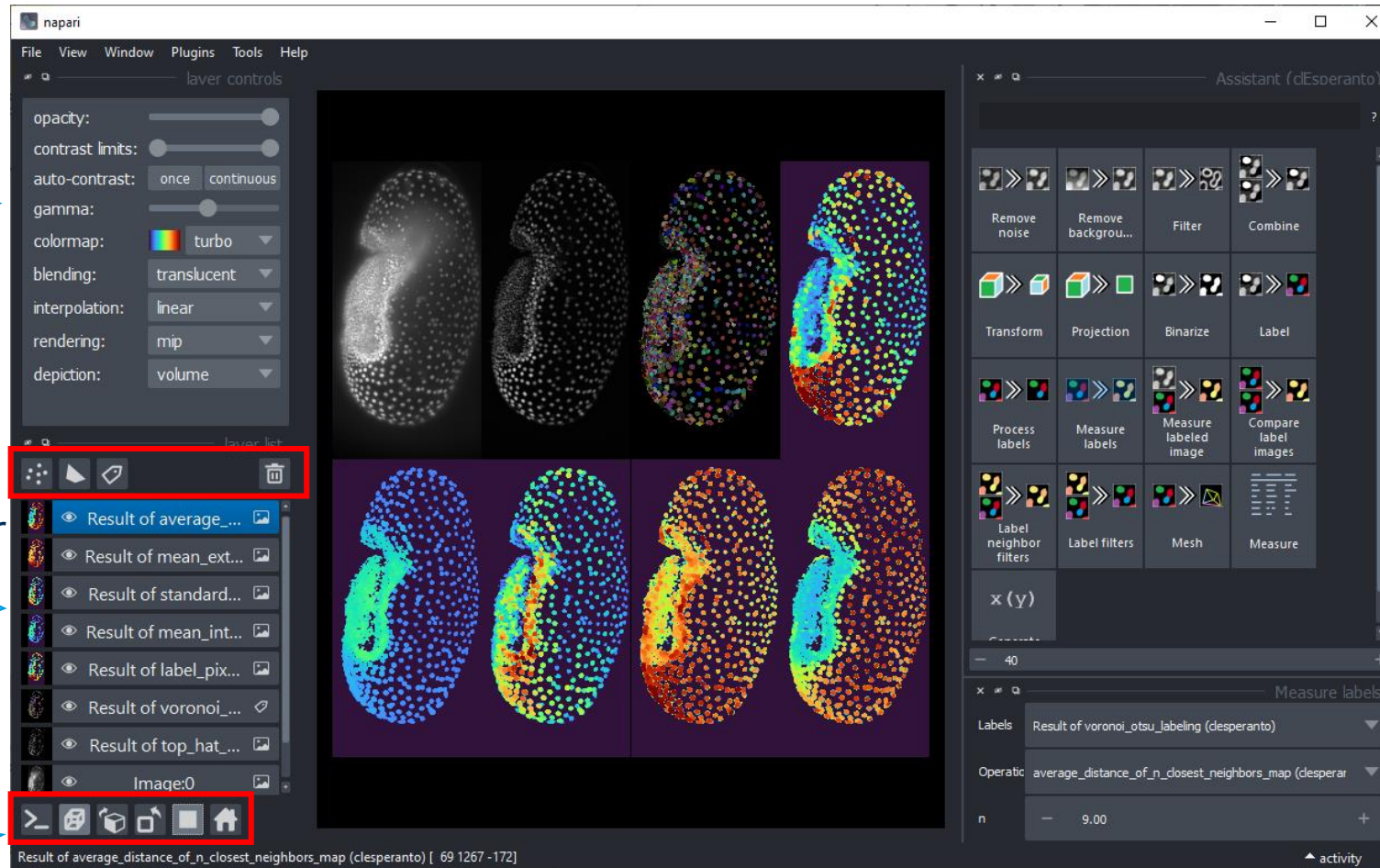
Add empty layers

Delete layer

Layers

`layer.visible = False`

Viewer controls



Dock widgets
(custom plugins)

Function widgets
(custom plugins)



Open Python Console



2D/3D view



Change axes order



Transpose visible axes



Grid mode



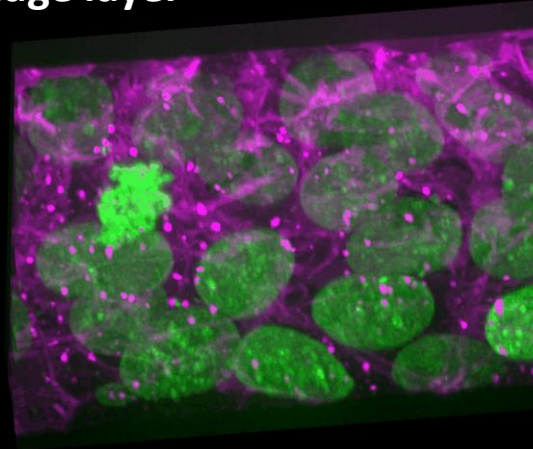
Reset view

<https://napari.org/tutorials/fundamentals/viewer.html>

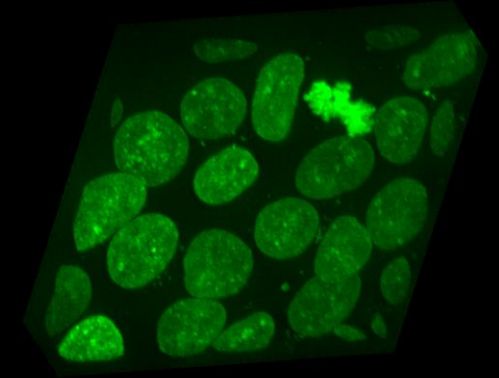
Layer types

- **Image layer:** Can be n-dimensional grayscale data (e.g., [CTZYX])
- **Labels layer:** Similar to image layer, but contains only integer numbers (e.g., 0,1,2,3,...)
- **Points layer:** List of coordinates in space
- **Vectors layer:** Direction from point A to point B
- **Surface layer:** Mesh (Vertices, Faces, Values)
- **Tracks layer:** Follow objects through time

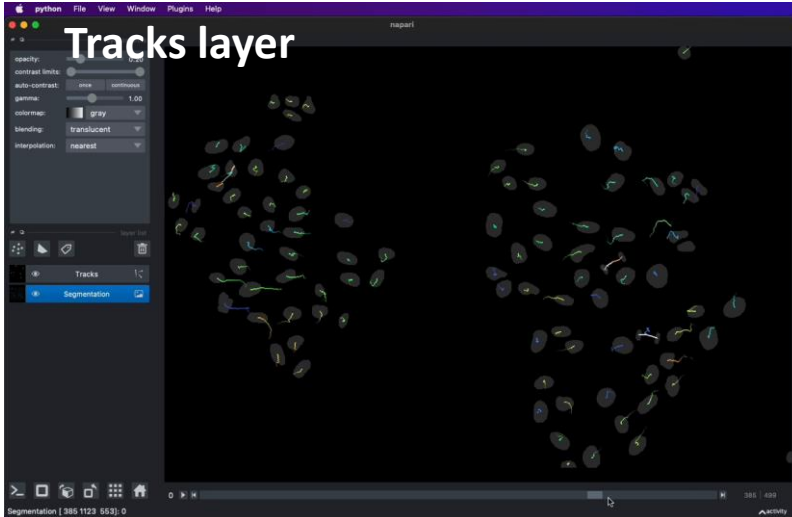
Image layer



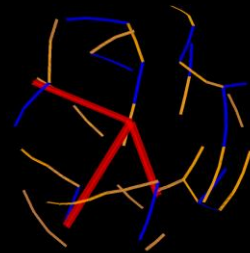
Labels layer



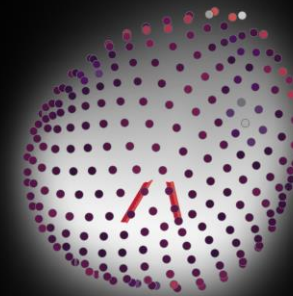
Tracks layer



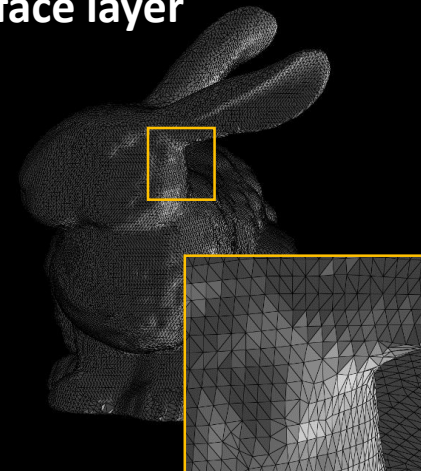
Vectors layer



Points layer



Surface layer



<https://github.com/quantumjot/arboretum> licensed under [MIT license](#)

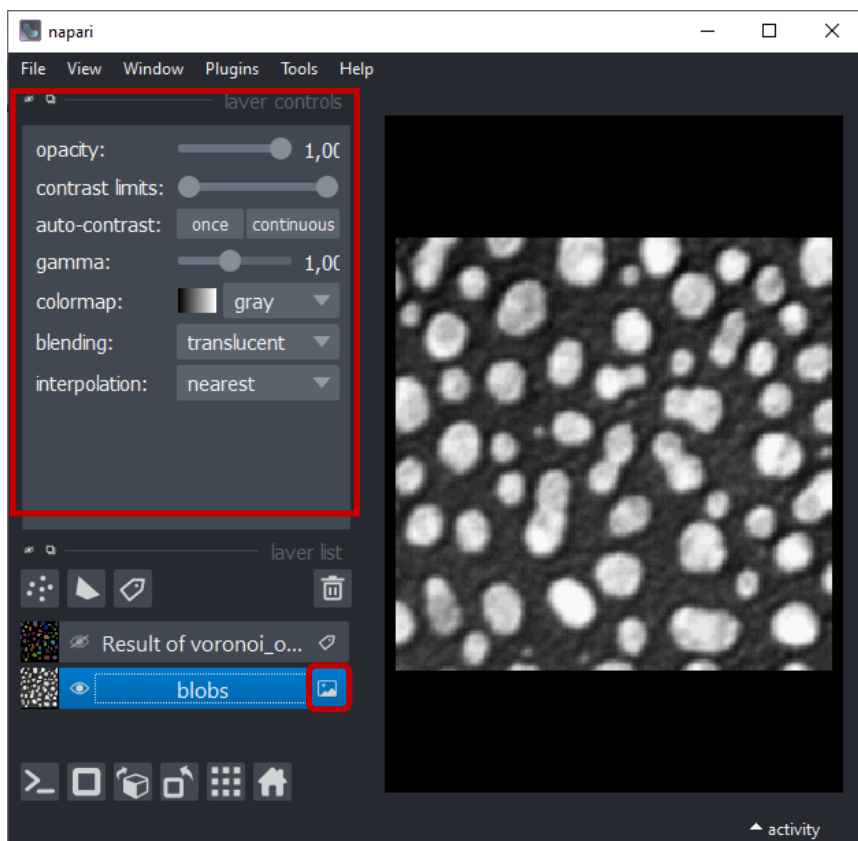
<https://github.com/campaslab/napari-stress>

<https://gitlab.kitware.com/vtk/vtk>

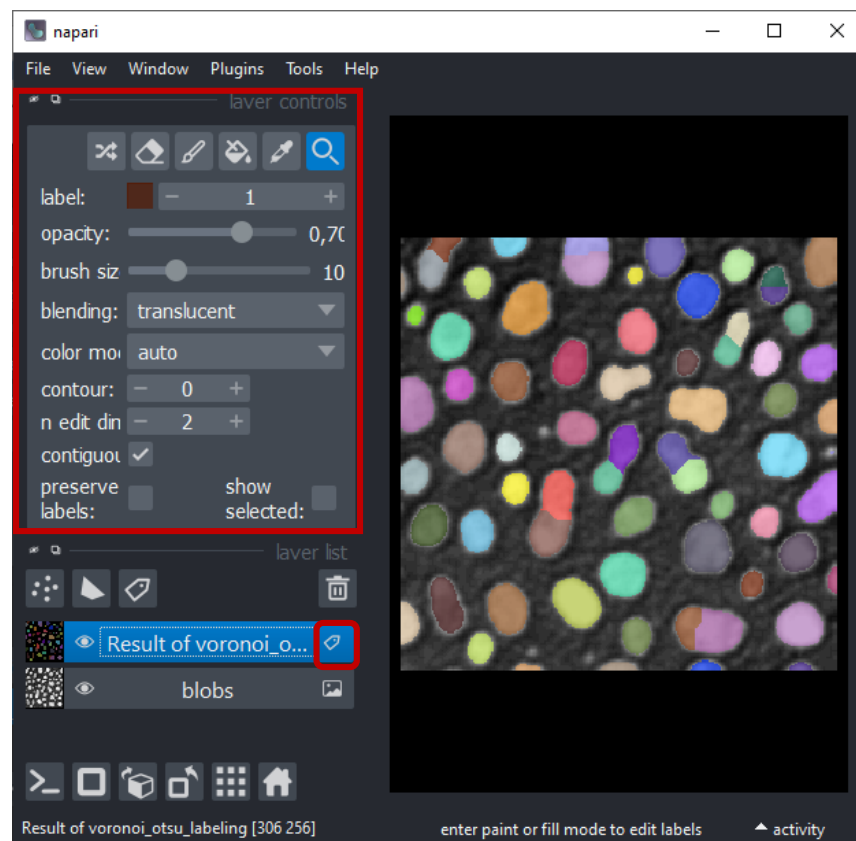
Layer types

Different layers have different tools and options

Image Layer



Labels Layer



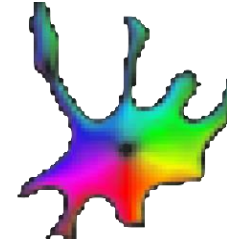
Python image & data analysis tools are powerful!



Python scientific image computing
toolbox
<https://github.com/scikit-image>



Python machine-learning toolbox
<https://github.com/scikit-learn>



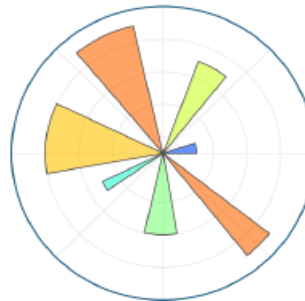
Cell segmentation
<https://github.com/MouseLand/cellpose>



Object detection with Stardist
<https://github.com/stardist>



Data visualization & exploration
<https://github.com/matplotlib>
<https://github.com/seaborn>



GPU-accelerated image processing
<https://github.com/clEsperanto>

cle._

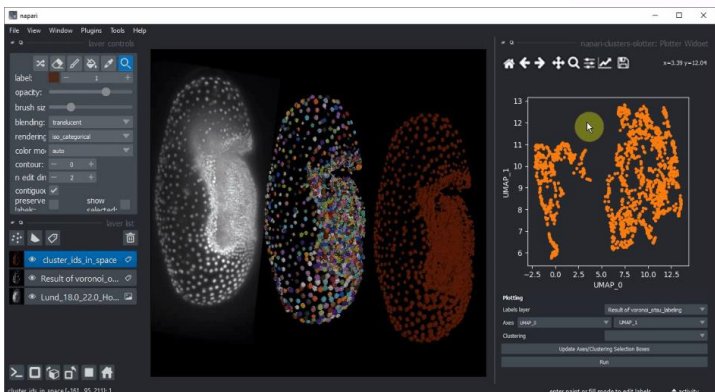
Napari



Existing functionality can
be turned into plugins!
→ Interactivity
→ Automatic GUI
generation

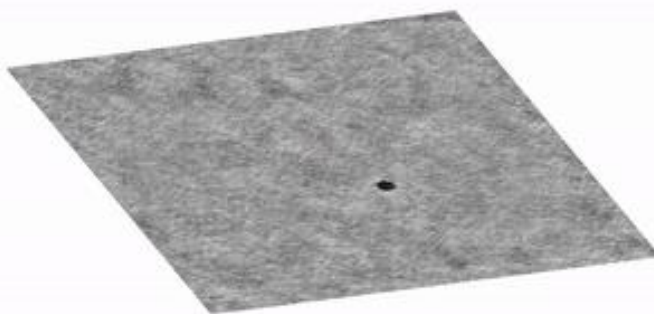
napari plugins

clusters-plotter



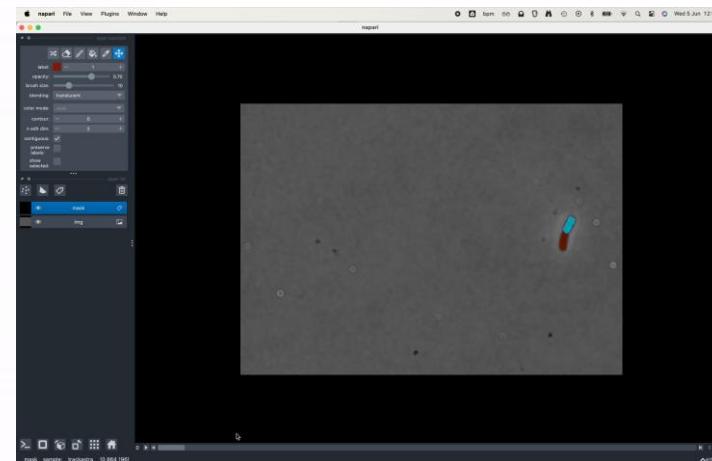
<https://github.com/BiAPoL/napari-clusters-plotter>

animation



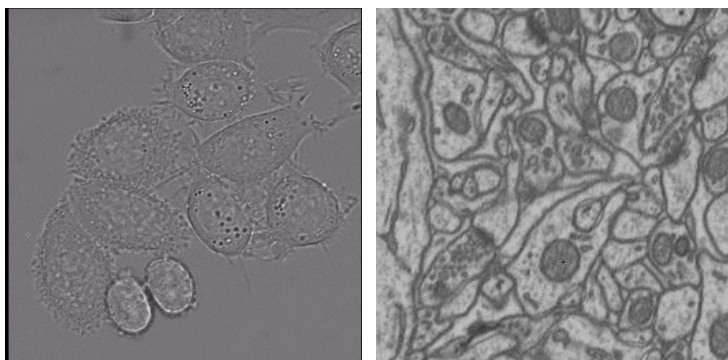
<https://github.com/napari/napari-animation>

trackastra



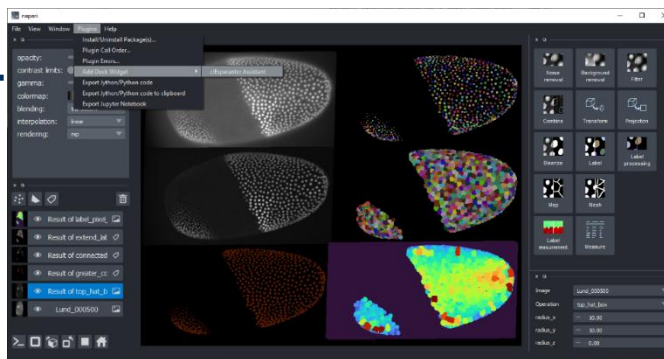
<https://github.com/weigertlab/trackastra>

micro-sam



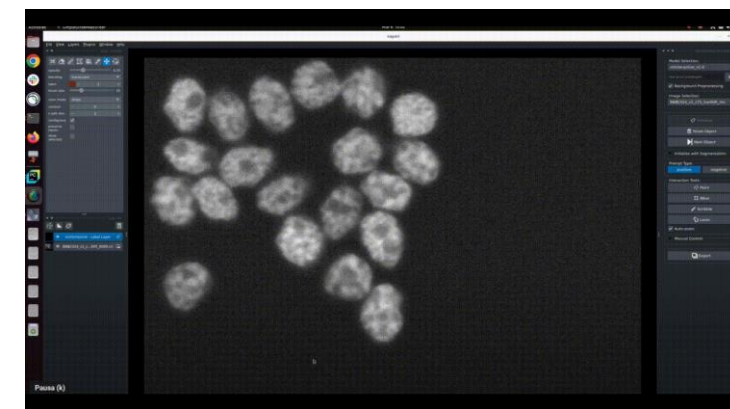
<https://github.com/computational-cell-analytics/micro-sam>

devbio-napari



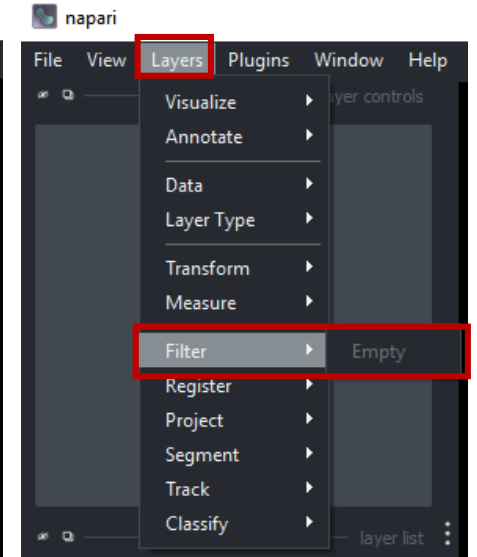
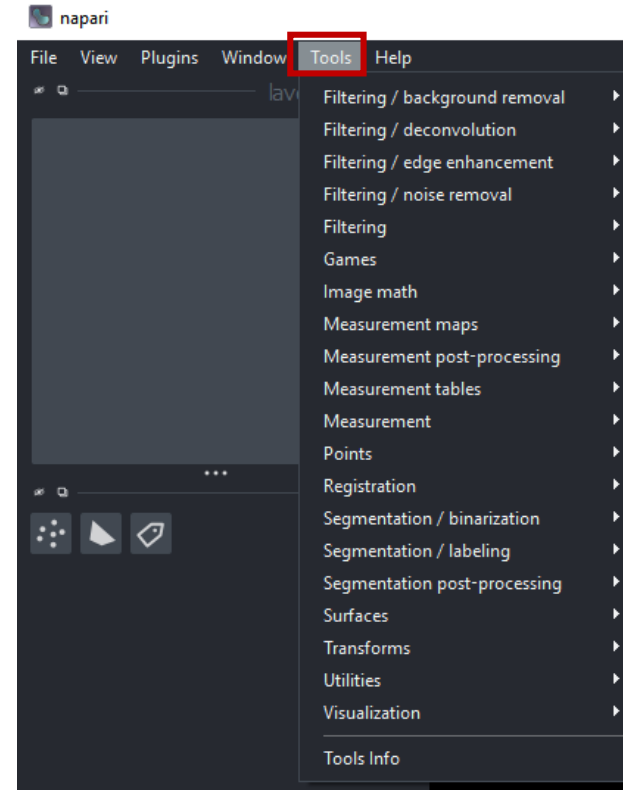
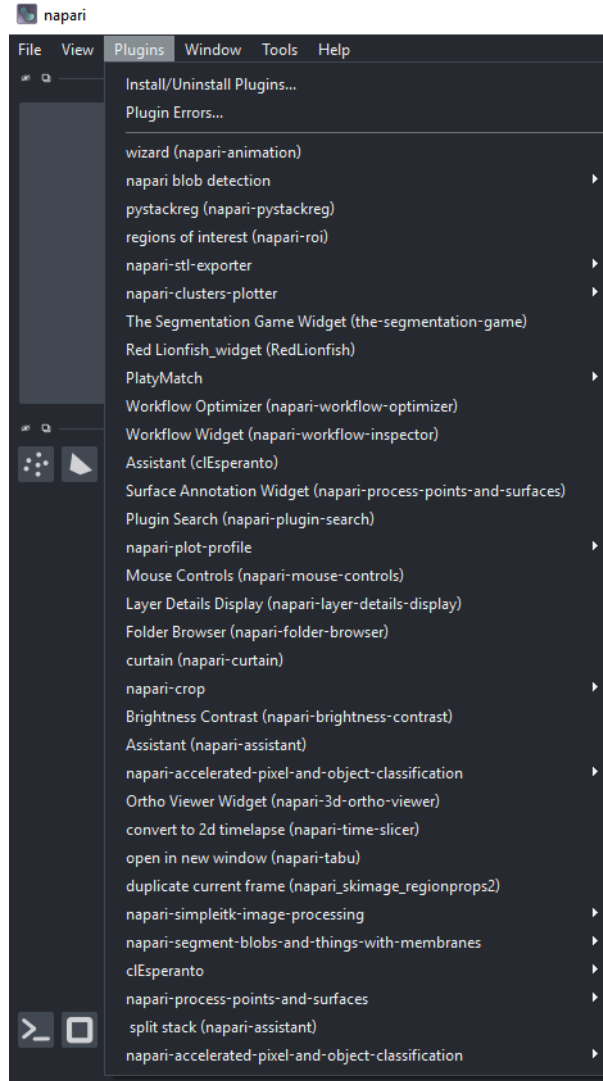
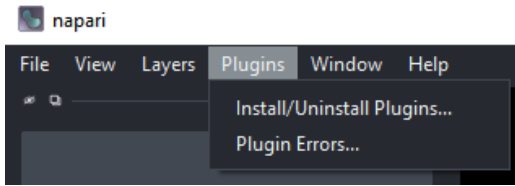
<https://github.com/haesleinhuepf/devbio-napari>

nninteractive



<https://github.com/MIC-DKFZ/napari-nninteractive>

Plugins and Layers Menus



devbio-napari plugin
bundle

napari $\geq$ 0.5.0

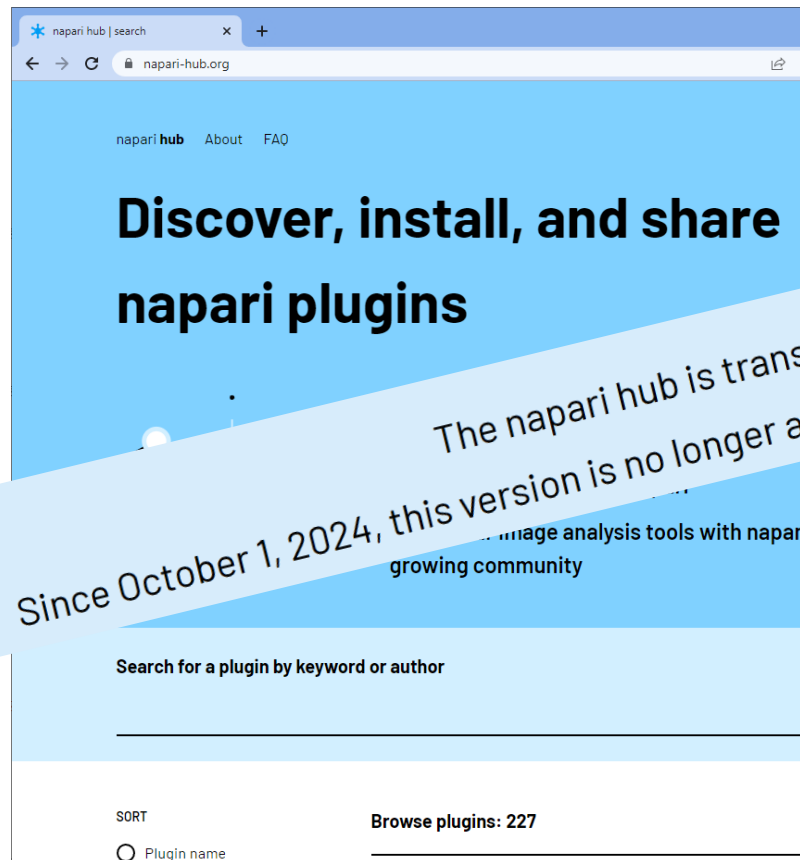
Layers sub-menu
items depend on
plugin
developers
populating them

The plugin you are looking for may be near you!

Search engine for napari plugins

A search bar from the Napari Hub website. It features a blue star icon on the left, followed by the text "napari hub | search" in a light blue font. To the right of the text is a search input field with a magnifying glass icon.

Search engine for napari plugins



Inspecting plugin usage and maintenance - GitHub

The screenshot shows the GitHub repository page for 'empanada-napari'. The repository is public and has 3 watches, 9 forks, and 22 stars. The main branch is 'main', and there are 17 branches and 10 tags. The repository is managed by 'barrybarry9', who merged pull request #34 from 'volume-em/empanada_v1.1.1' 8 months ago. The repository contains 210 commits.

Annotations on the screenshot include:

- A red box highlights the commit hash 'ffa7f39' and the text '8 months ago'.
- A red arrow points from the text 'Latests updates' to the commit hash.
- A red arrow points from the text 'Documentation' to the 'About' section, specifically to the link 'empanada.readthedocs.io/en/latest/emp...'.
- A red arrow points from the text 'Documentation' to the 'Releases' section, specifically to the 'Zenodo First Release' link.

File	Commit Message	Commit Date
.github/workflows	deployment ready	3 years ago
.napari	model and data docs	3 years ago
custom_configs	allow model architecture customization	3 years ago
empanada_napari	Updated requirements.txt and Count labels module.	8 months ago
images	docs	3 years ago
.gitignore	v1.1 release	last year
LICENSE	Initial commit	4 years ago
MANIFEST.in	deployment ready	3 years ago
README.md	Updated setup.cfg and requirements.txt file to include comp...	8 months ago
pyproject.toml	code addition/deletion for project package install	last year
requirements.txt	Updated setup.cfg and requirements.txt file to include comp...	8 months ago
setup.cfg	Updated setup.cfg and requirements.txt file to include comp...	8 months ago
tox.ini	preprint citation	3 years ago

Image Data Management and Access: Napari – OMERO integration

- Demonstration: napari-OMERO plugin

Working with images in python

Open images

```
from skimage.io import imread  
image = imread("blobs.tif")
```

image

```
array([[ 40,  32,  24, ..., 216, 200, 200],  
       [ 56,  40,  24, ..., 232, 216, 216],  
       [ 64,  48,  24, ..., 240, 232, 232],  
       ...,  
       [ 72,  80,  80, ...,  48,  48,  48],  
       [ 80,  80,  80, ...,  48,  48,  48],  
       [ 96,  88,  80, ...,  48,  48,  48]], dtype=uint8)
```

Images are *just* multi-dimensional arrays or "arrays of arrays".



https://biapol.github.io/AMHCT_Bio_Image_Analysis_2025/01_images_are_numpy_arrays.html

Opening images should, but it is not always simple

- **Paths** need attention
 - Backslashes often cause problems on Windows
OSError: [Errno 22] Invalid argument:
Add a lowercase 'r' before path to fix that: `image = imread(r'C:\data\blobs.tif')`
- **Data may be too big** for your computer
 - They would need to be opened *lazily*
- **Proprietary file formats** ('.czi', '.lif', '.nd2', '.ims', etc) may need additional packages
 - In Fiji, Bioformats usually takes care of opening many different proprietary file formats
 - In Python/napari, you may need to install:
 - <https://github.com/bioio-devs/bioio>
 - <https://github.com/AllenCellModeling/napari-aicsimageio>
- **Metadata** may be lost
 - For example, some libraries return an image with dimensions like this

What dimension is this? (1, 1, **1**, 80, 520, 692, 1)

?

Tile

Z

Time

I don't know

Data Structure

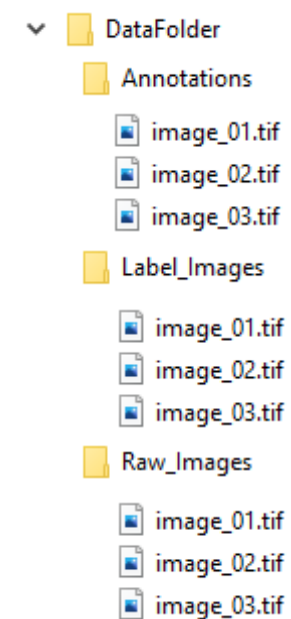
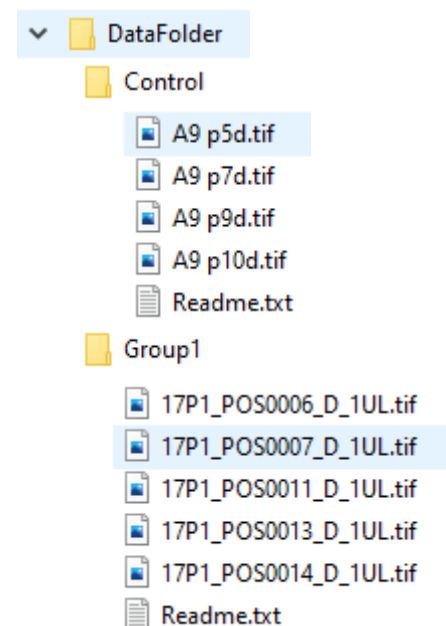
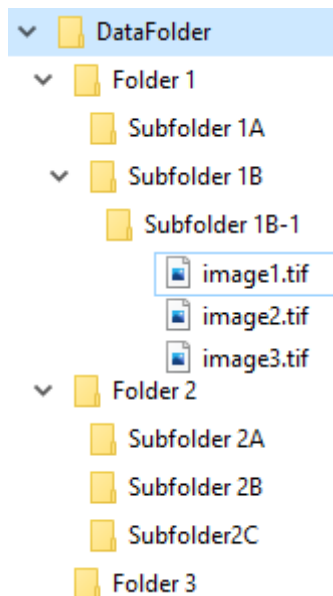
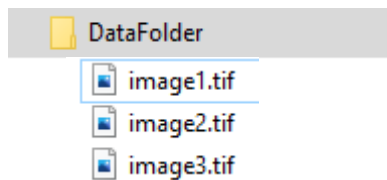


Image by [eikira](https://pixabay.com/photos/trunk-automobile-vehicle-luggage-1478832/) from [Pixabay](https://pixabay.com/photos/trunk-automobile-vehicle-luggage-1478832/) (<https://pixabay.com/photos/trunk-automobile-vehicle-luggage-1478832/>)

OMERO

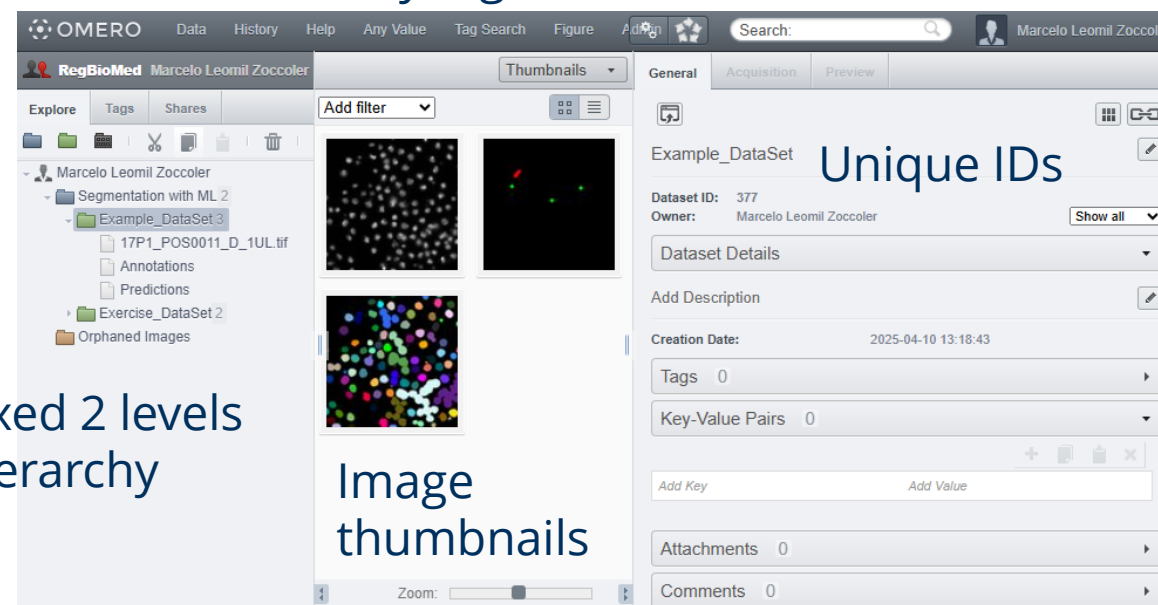
OMERO is designed with scientists in mind



OMERO handles all your images using a secure central repository, from the microscope to **publication**.

Using the power of Bio-Formats, OMERO supports over 140 image file formats, including all major microscope formats.

Filter by tags and more...



Fixed 2 levels hierarchy

Image thumbnails

Unique IDs

Metadata



IMPORT DATA

Upload your data via the desktop app or command line, use our custom dropbox tool, or have your facility manager import it for you



ORGANIZE

View and organize your data from anywhere you have internet access



VIEW

Work with your images using the OMERO.insight desktop app, with OMERO.web in your browser, or from 3rd party software



ANALYZE

Use ImageJ, MATLAB, R, scripts and other tools to run analysis on data in OMERO and save results on the server



SHARE

Share data with collaborators or your PI using the Groups/Users permission system.



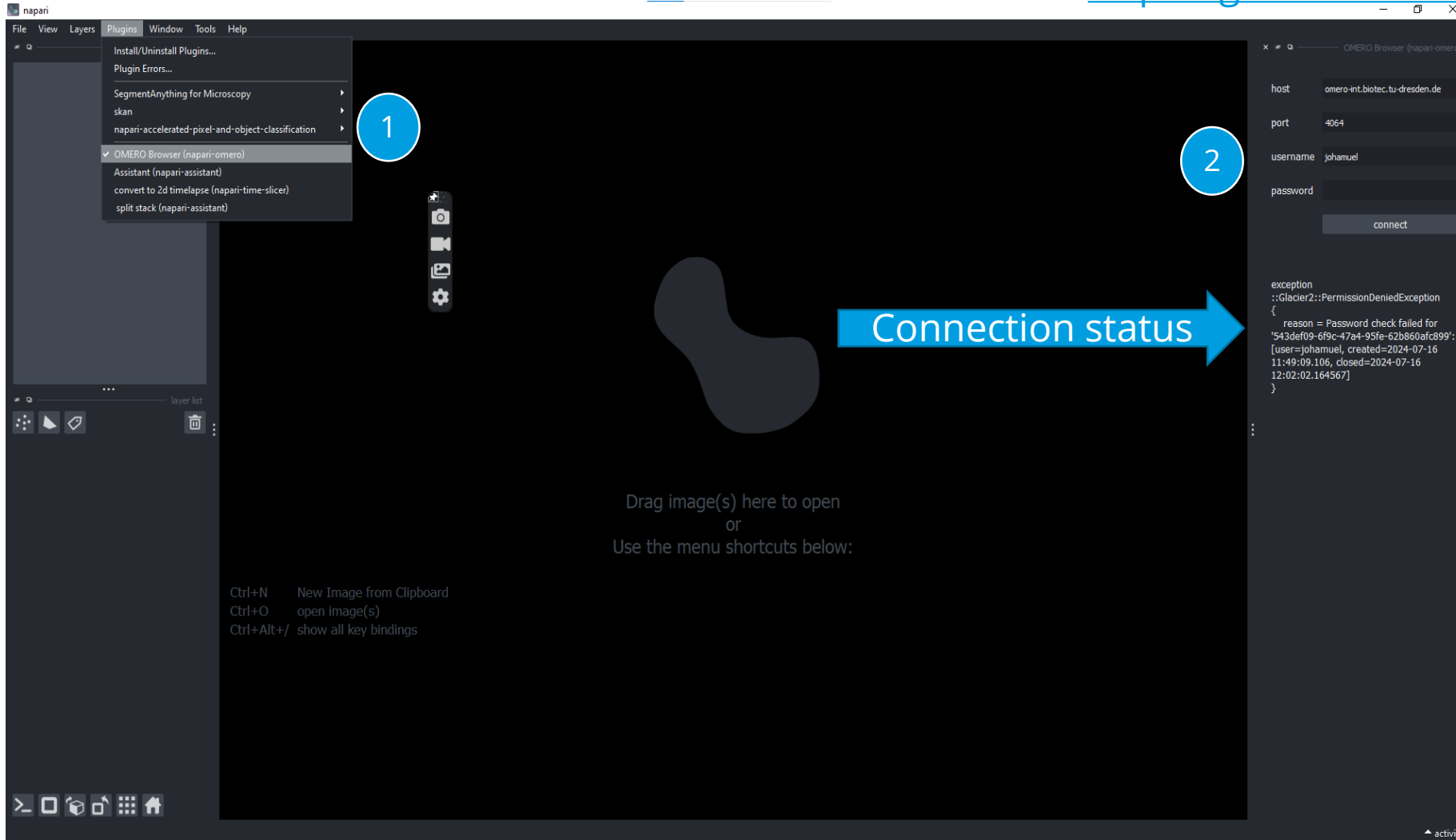
PUBLISH

Publish direct from the server via a customized website or embedded viewer

<https://www.openmicroscopy.org/omero/scientists/>

Napari-OMERO usage

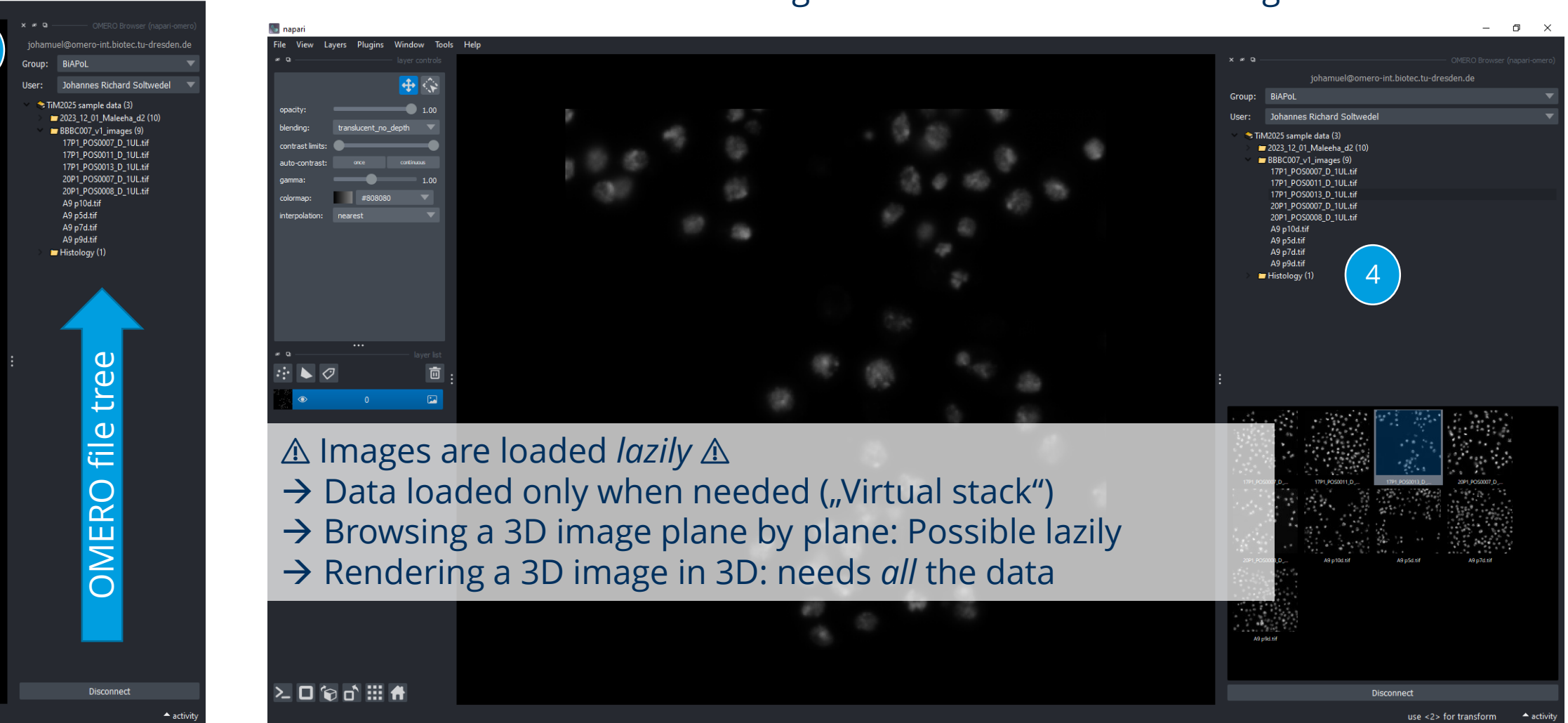
<https://github.com/tlambert03/napari-omero>



1. Select Plugin from plugin menu
2. Log into BiAPoL OMERO with ZIH credentials

Napari-OMERO usage

3. Select group and user
4. Selecting item from file tree loads image into viewer



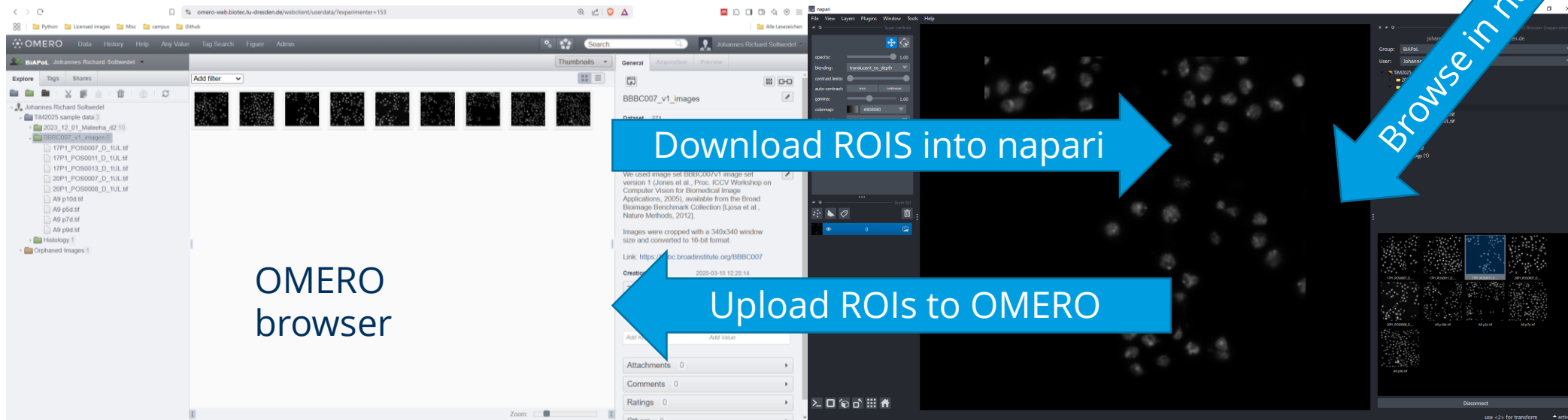
The screenshot displays the Napari-OMERO interface. On the left, the 'OMERO Browser' sidebar shows a file tree with a blue arrow pointing upwards and the text 'OMERO file tree'. A blue circle with the number '3' is next to the 'Group' dropdown, and another blue circle with the number '4' is next to a file in the tree. The main viewer area shows a 3D image of a cell. A semi-transparent text box in the center contains the following text:

⚠ Images are loaded *lazily* ⚠
→ Data loaded only when needed („Virtual stack“)
→ Browsing a 3D image plane by plane: Possible lazily
→ Rendering a 3D image in 3D: needs *all* the data

Some tips and tricks for OMERO from Python: <https://biapol.github.io/omero-tools>

Napari-OMERO usage

- ROI Upload
- Multiscale support: Browse larger-than-RAM samples (*3D limitation remains*)
- **Coming soon:**
- ROI download



Segmentation and Supervised Machine Learning

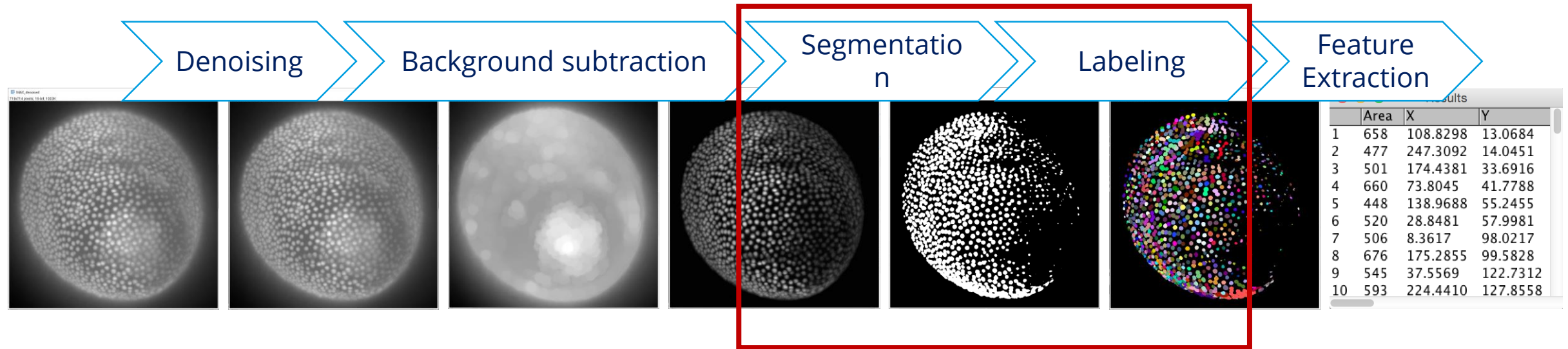
Random Forest Classifiers

- Pixel Classifier

Image Analysis Classic Workflows

Image analysis workflow is a series of processes/functions (not always linear) applied to images to achieve a certain goal (usually some measurements)

Here is a classic example:



Application: Segmentation

Aim:

Separate background from foreground

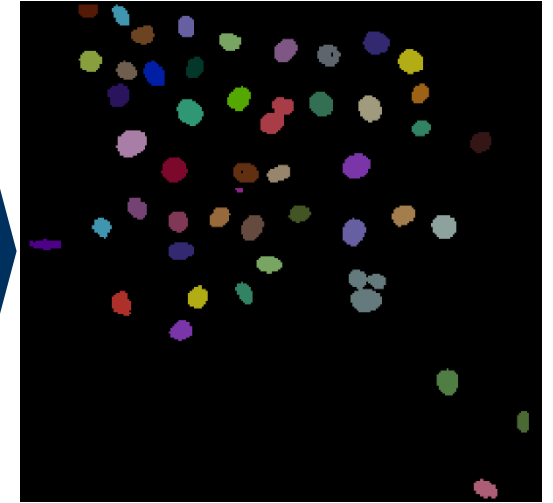
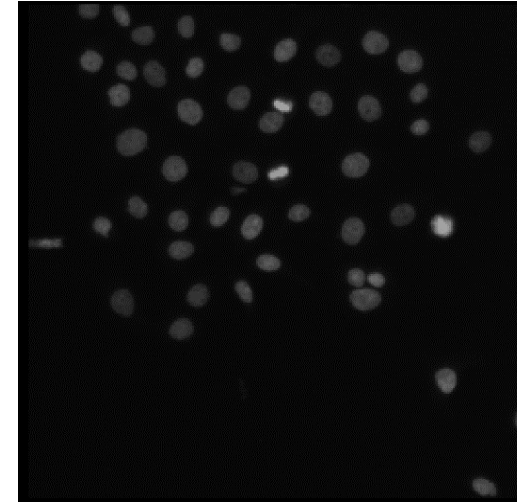
Vocabulary:

- **Segmentation:**
 - Assigning a meaningful *label* to each pixel
 - Segmentation is a *classification* problem
- **Semantic segmentation:**

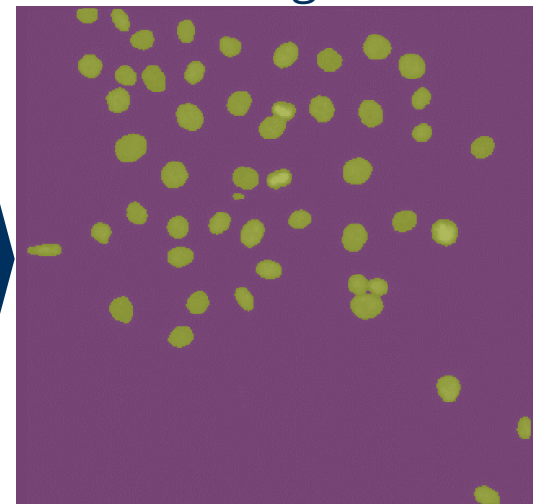
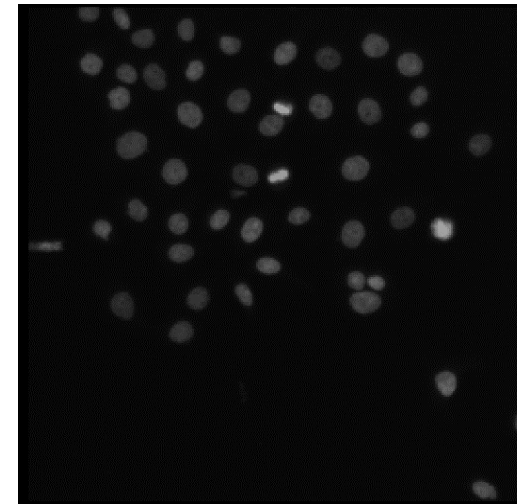
Differentiate pixels into multiple *classes* (e.g., membrane, nucleus, cytosol, etc.)
- **Instance segmentation:**

Differentiate multiple occurrences of the same class into separate instances of this class (e.g., separate *label* for each cell in image)

<https://scikit-image.org/docs/stable/api/skimimage.data.html>



Instance segmentation



Semantic segmentation

Instance segmentation

In order to allow the computer differentiating objects, connected component analysis (CCA) is used to mark pixels belonging to different objects with different numbers

Background pixels are marked with 0.

The maximum intensity of a labelled map corresponds to the number of objects.

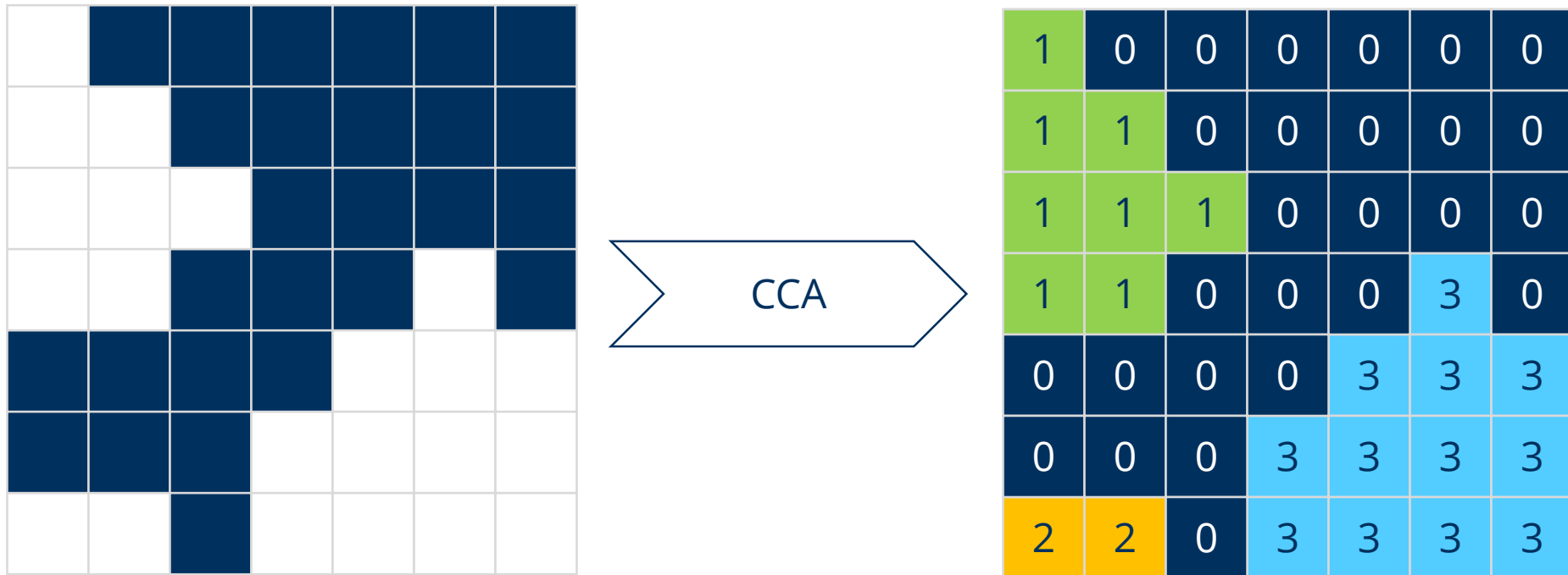


Image segmentation using thresholding

Finding the right workflow towards a good segmentation takes time

A priori, we usually don't know which information in the image is useful for a good segmentation

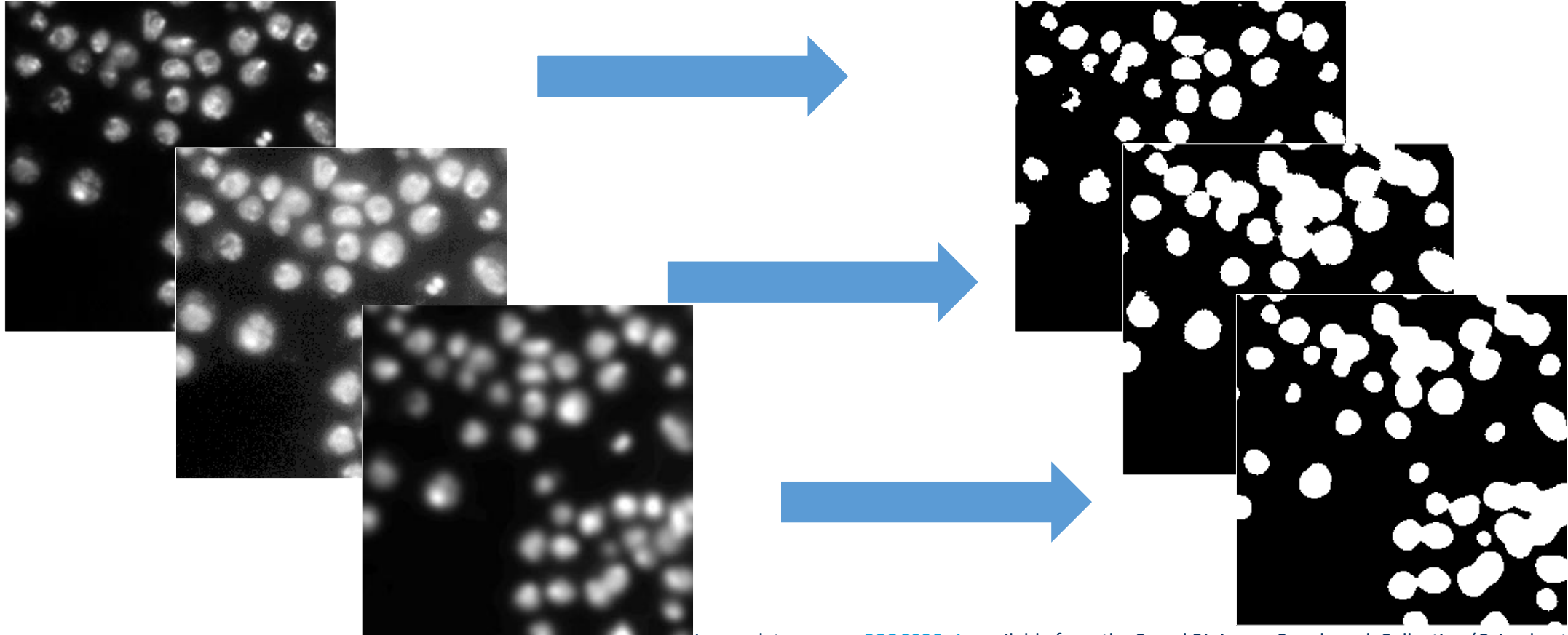


Image data source: [BBBC038v1](#), available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019)

Machine learning

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

Artificial intelligence

Machine learning

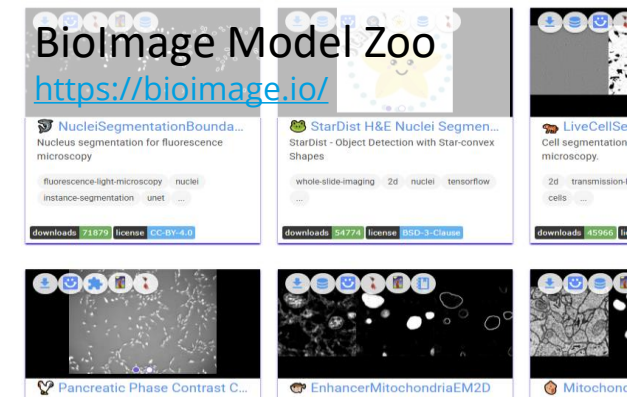
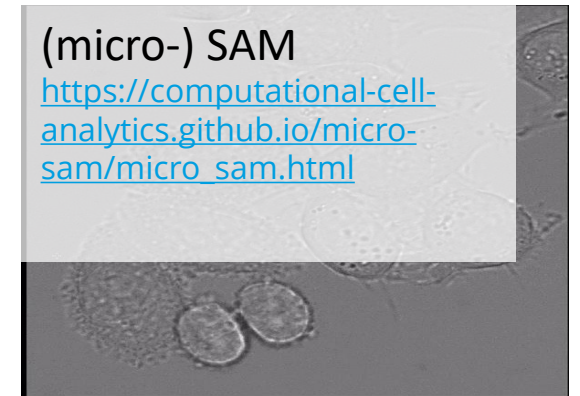
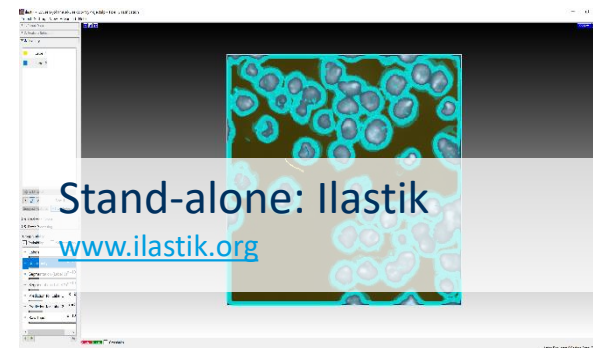
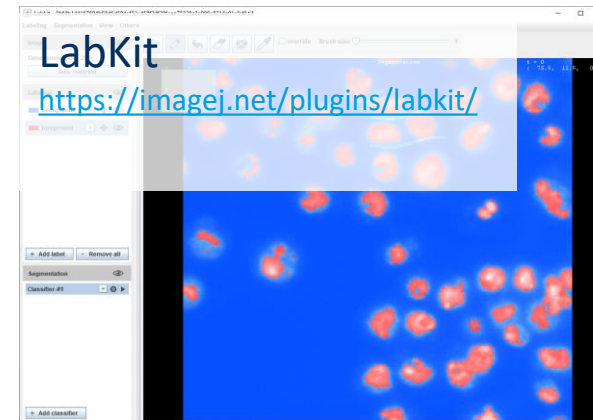
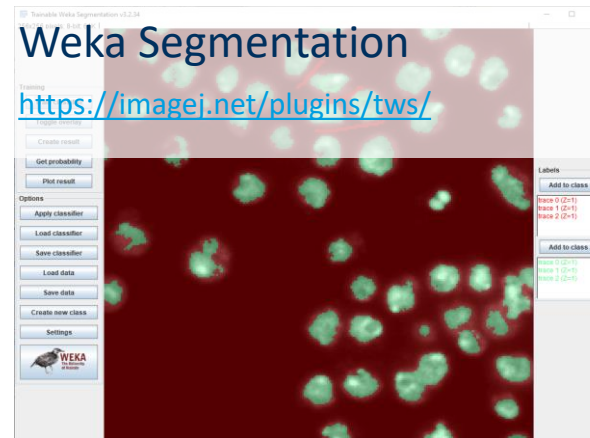
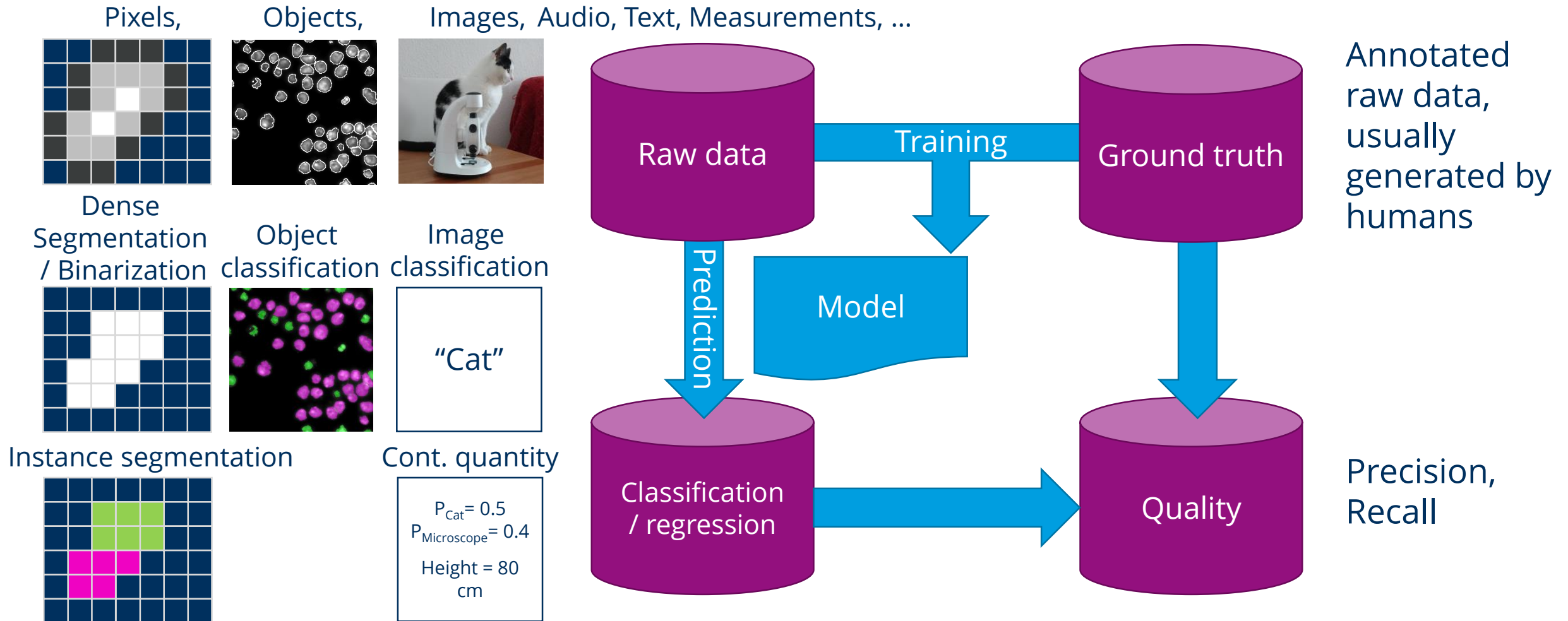


Image data source: [BBBC038v1](#), available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019)

Machine learning

Automatic construction of predictive models from given data



Segmentation: Latest developments

1970s-2010: Filtering, thresholding

A Threshold Selection Method from Gray-Level Histograms

NOBUYUKI OTSU

Abstract—A nonparametric and unsupervised method of automatic threshold selection for picture segmentation is presented. An optimal threshold is selected by the discriminant criterion, namely, so as to maximize the separability of the resultant classes in gray levels. The procedure is very simple, utilizing only the zeroth- and the first-order cumulative moments of the gray-level histogram. It is straightforward to extend the method to multithreshold problems. Several experimental results are also presented to support the validity of the method.

2010s: Random forests et al.



Stand-alone: Ilastik
www.ilastik.org

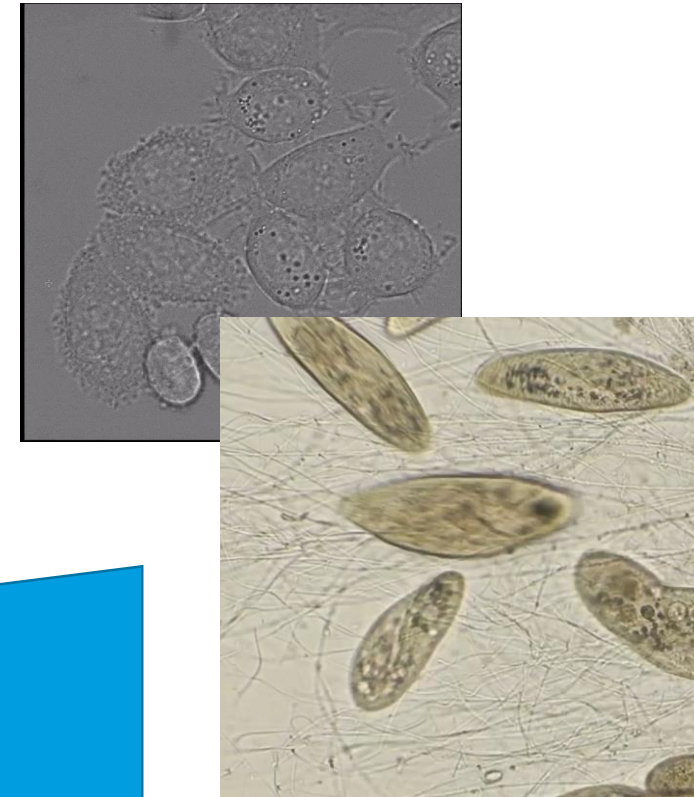
Deep learning

2015: UNet
2018: Stardist



Foundational models

2023: Segment anything (SAM)
2024: SAM 2



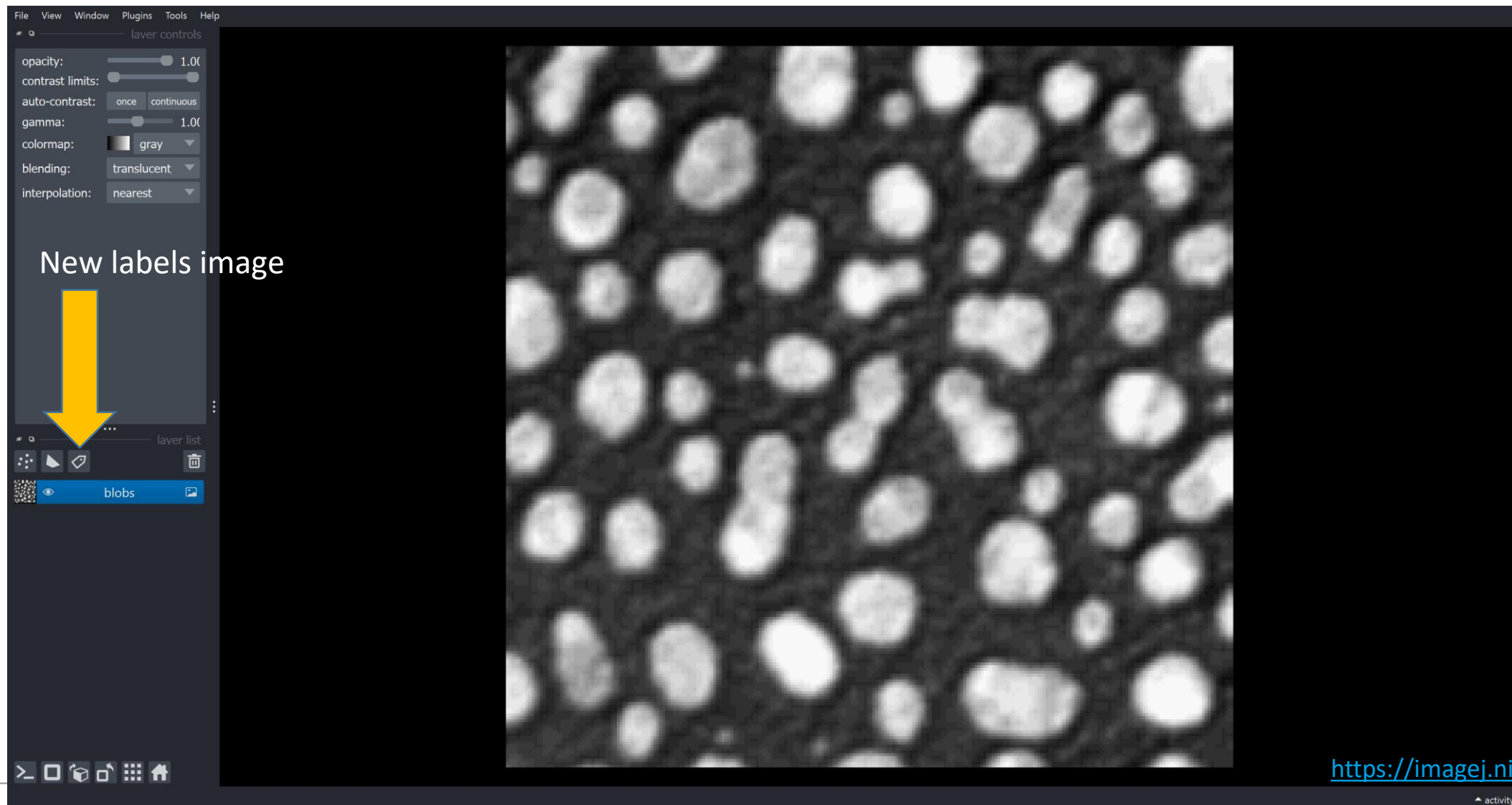
Computational demand

Segmentation and Supervised Machine Learning in napari

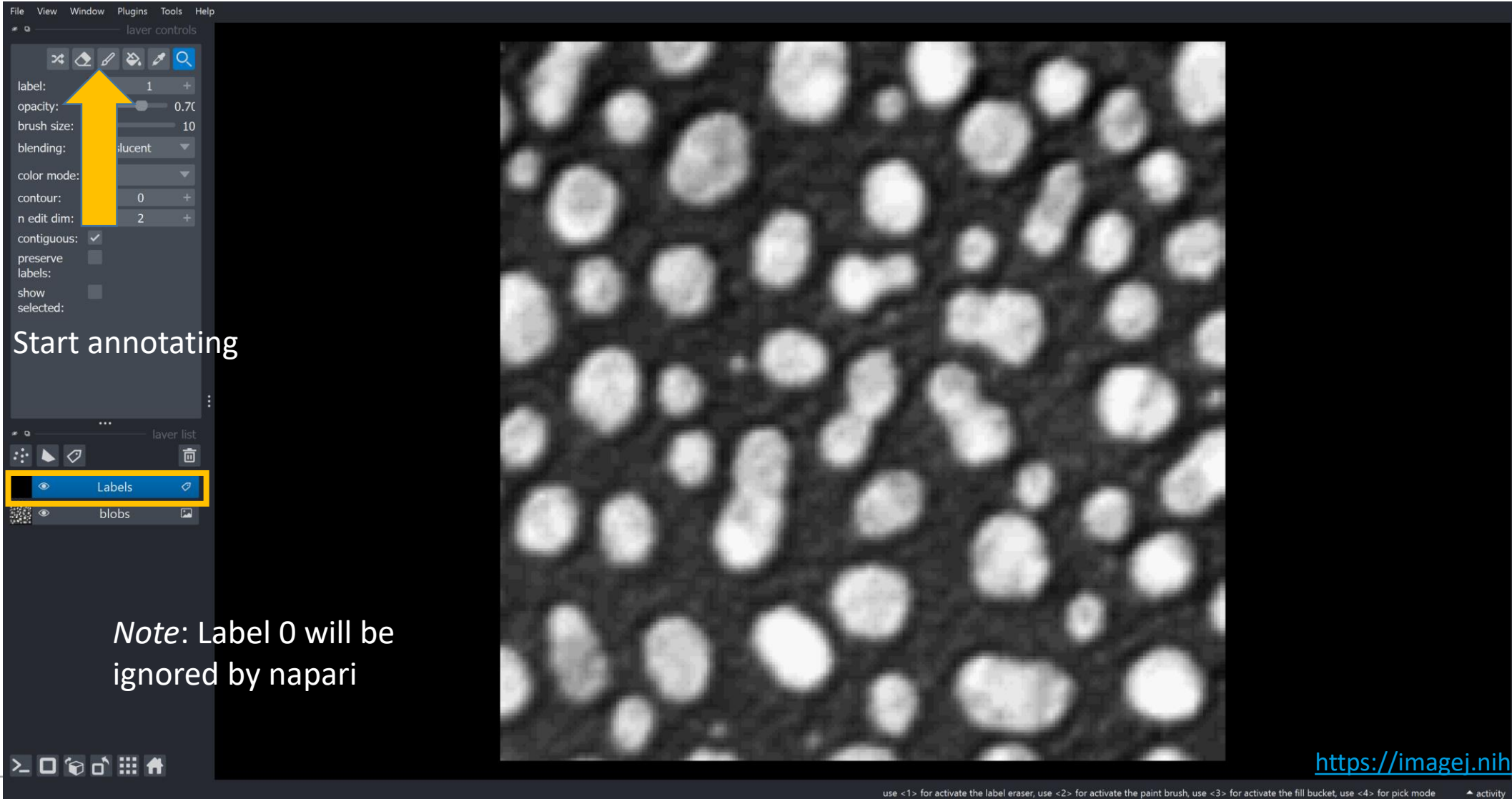
Random Forest Classifiers

- Pixel Classifier
- **napari-apoc plugin**

In napari: annotation



In napari: annotation



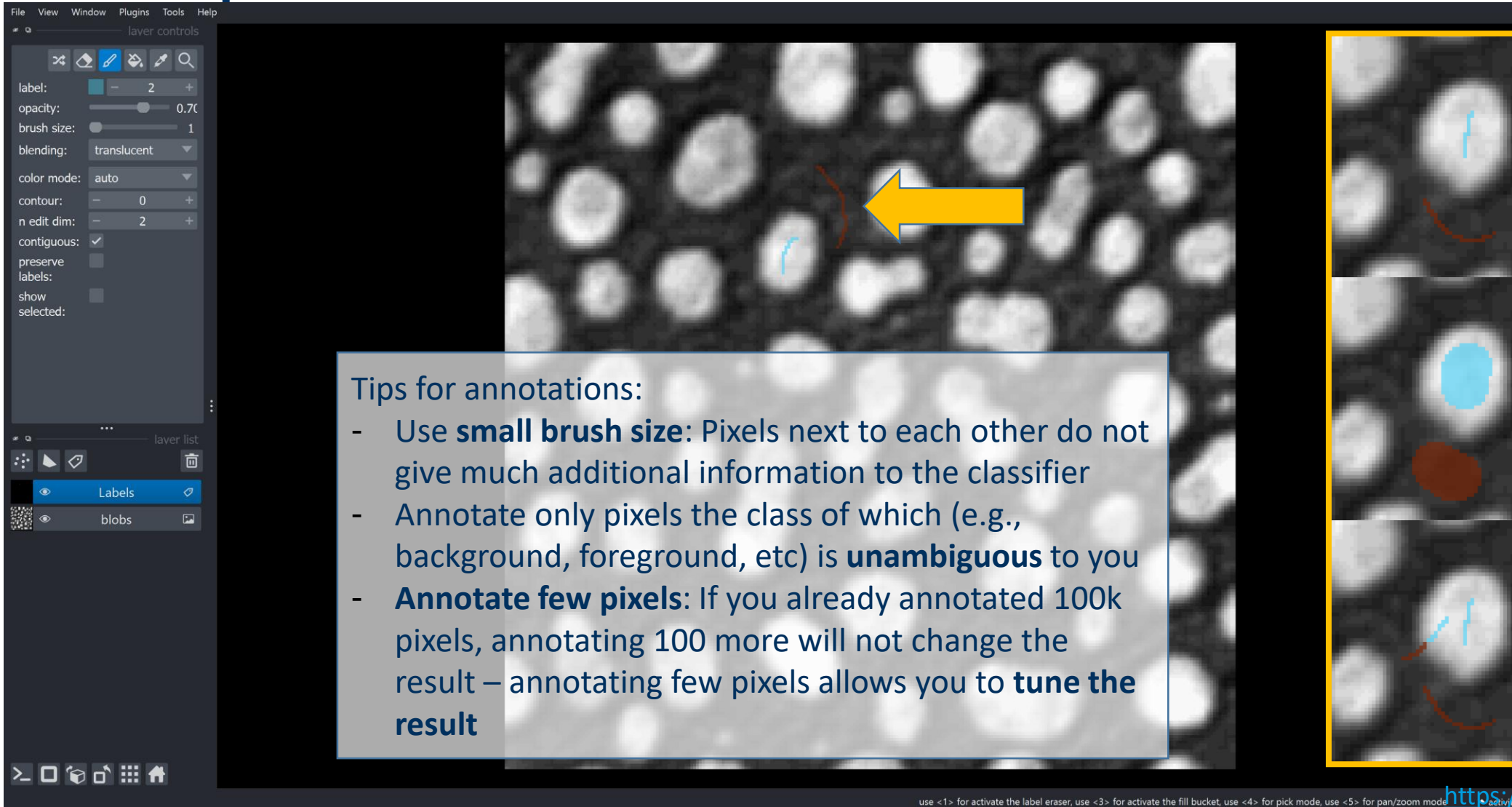
The screenshot shows the Napari software interface. On the left, the 'layer controls' panel is visible, featuring a toolbar with icons for selection, pan, zoom, and annotation. A yellow arrow points to the 'brush' icon in the toolbar. Below the toolbar, the 'layer controls' panel displays settings for the active layer (labeled '1'): label: 1, opacity: 0.70, brush size: 10, blending: translucent, color mode: (dropdown), contour: 0, n edit dim: 2, contiguous: [checked], preserve labels: [unchecked], show selected: [unchecked]. Below this, the 'layer list' panel shows a list of layers: 'Labels' (highlighted with a yellow box) and 'blobs'. The main view area displays a grayscale microscopy image of cells. At the bottom of the interface, a status bar contains keyboard shortcuts: 'use <1> for activate the label eraser, use <2> for activate the paint brush, use <3> for activate the fill bucket, use <4> for pick mode' and an 'activity' indicator.

Start annotating

Note: Label 0 will be ignored by napari

<https://imagej.nih.gov/ij/images/>

In napari: annotation



layer controls

label: 2
opacity: 0.70
brush size: 1
blending: translucent
color mode: auto
contour: 0
n edit dim: 2
contiguous: ☒
preserve labels: ☐
show selected: ☐

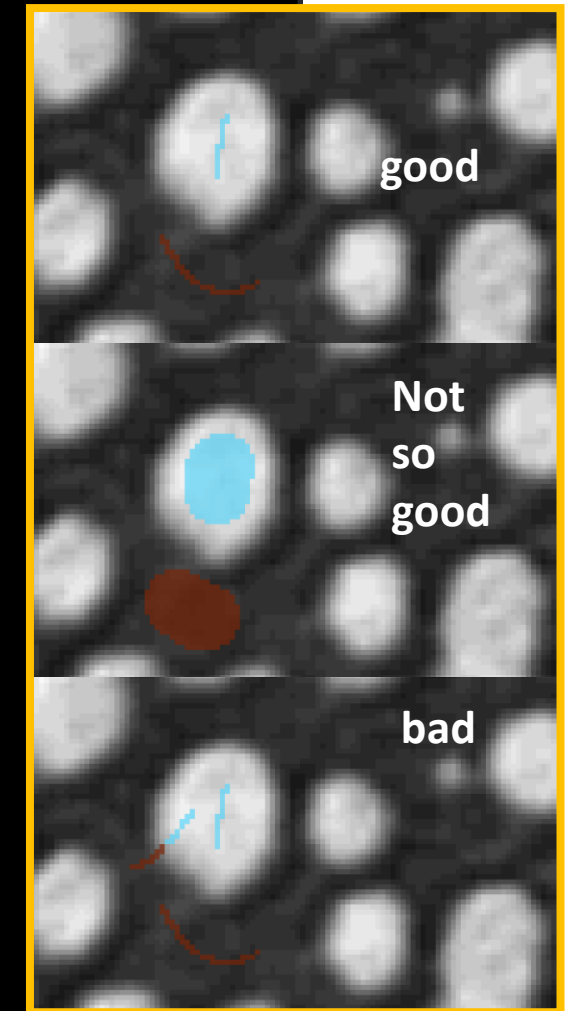
layer list

Labels
blobs

Tips for annotations:

- Use **small brush size**: Pixels next to each other do not give much additional information to the classifier
- Annotate only pixels the class of which (e.g., background, foreground, etc) is **unambiguous** to you
- **Annotate few pixels**: If you already annotated 100k pixels, annotating 100 more will not change the result – annotating few pixels allows you to **tune the result**

use <1> for activate the label eraser, use <3> for activate the fill bucket, use <4> for pick mode, use <5> for pan/zoom mode



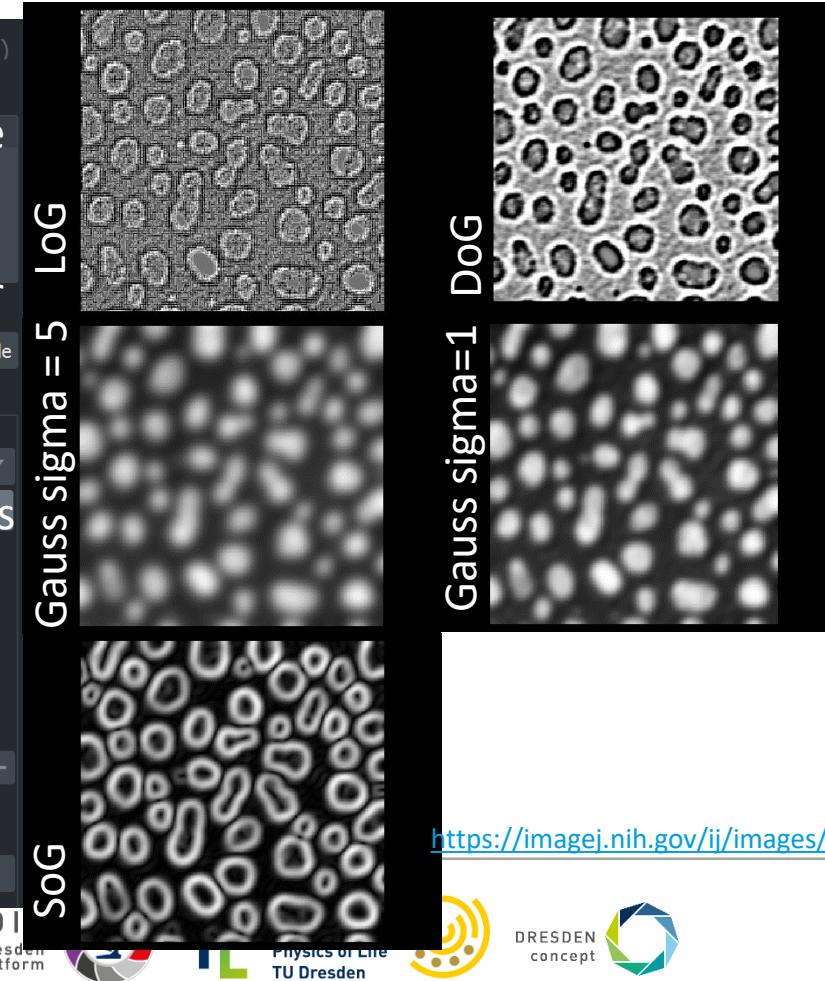
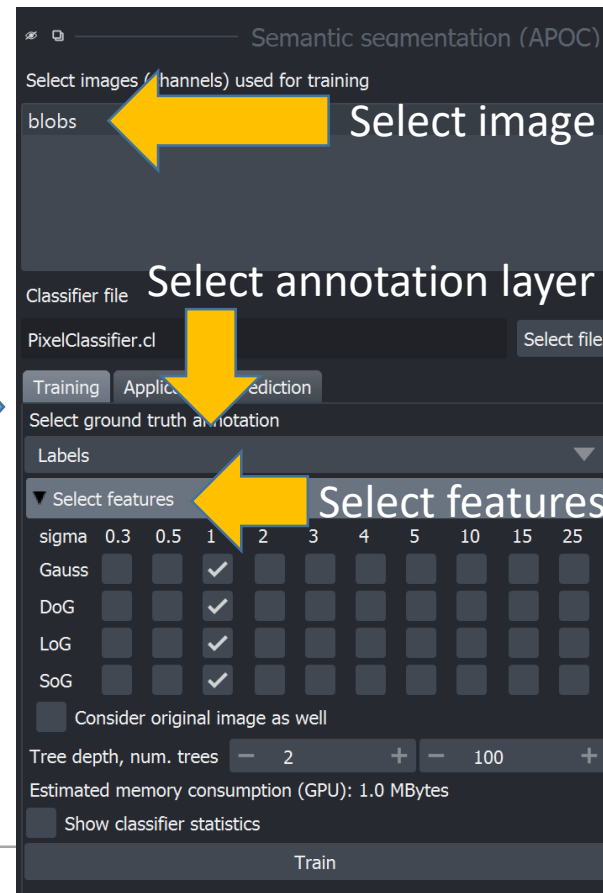
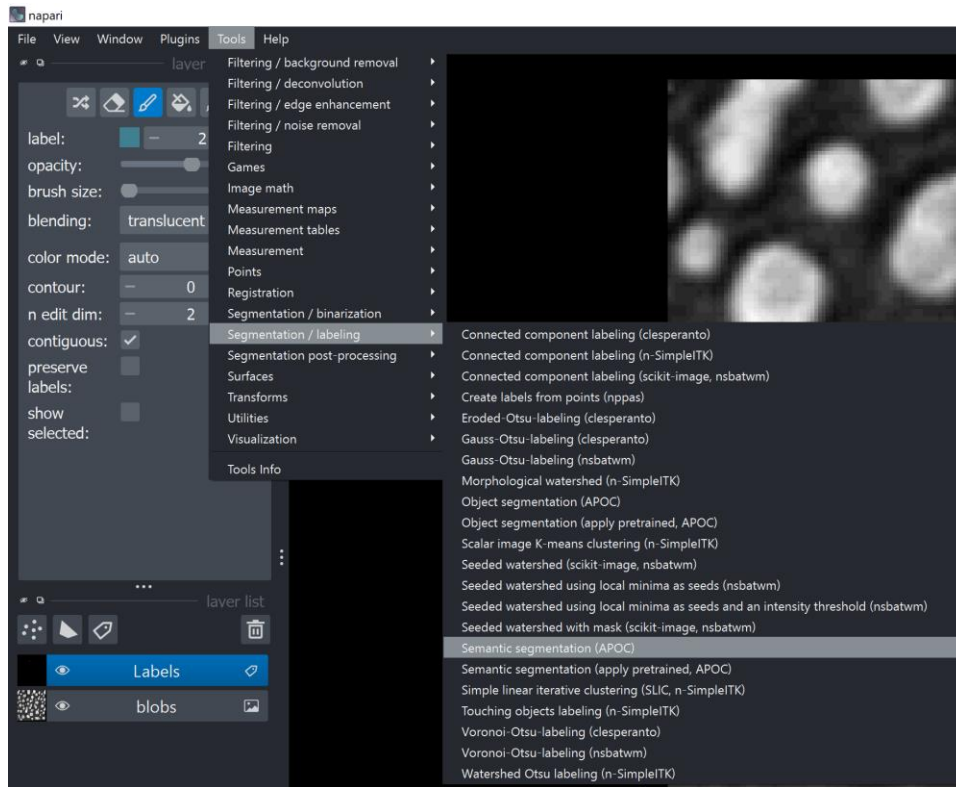
<https://imagej.nih.gov/ij/images/>

In napari: Semantic segmentation (training)

Two options:

Semantic segmentation: Predict class of every pixel according to annotation

Object segmentation: Assumes that class "1" refers to background – applies connected component analysis to other class



In napari: Semantic segmentation (training)

Unhappy with it? Simply select the “Labels” layer and annotate some more

layer controls

label: 1
opacity: 0.70
brush size: 10
blending: translucent
color mode: auto
contour: 0
n edit dim: 2
contiguous: ☒
preserve labels: ☐
show selected: ☐

layer list

- Result of PixelClassi...
- Labels
- blobs

Semantic segmentation (APOC)

Select images (channels) used for training

blobs

Classifier file

PixelClassifier.cl Select file

Training Application / Prediction

Select ground truth annotation

Labels

Select features

	sigma	0.3	0.5	1	2	3	4	5	10	15	25
Gauss				☒							
DoG				☒							
LoG				☒							
SoG				☒							

Consider original image as well

Tree depth, num. trees 2 100

Estimated memory consumption (GPU): 1.0 MBytes

Show classifier statistics

Train

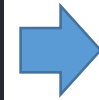
<https://imagej.nih.gov/ij/images/>

use <1> for activate the label eraser, use <2> for activate the paint brush, use <3> for activate the fill bucket, use <4> for pick mode

Semantic segmentation: Choosing the right features

Why not just do this?

- Not all features are equally relevant!
- Calculating the features takes time and computation resources!



	1	2
1 gaussian_blur=1	0.126	0.121
2 difference_of_gaussian=1	0.322	0.241
3 laplace_box_of_gaussian_blur=1	0.007	0.017
4 sobel_of_gaussian_blur=1	0.014	0.017
5 gaussian_blur=0.3	0.105	0.241
6 gaussian_blur=0.5	0.105	0.069
7 difference_of_gaussian=0.5	0.007	0.052
8 difference_of_gaussian=0.3	0.007	0.000
9 laplace_box_of_gaussian_blur=0.3	0.000	0.000
10 laplace_box_of_gaussian_blur=0.5	0.007	0.017
11 sobel_of_gaussian_blur=0.5	0.035	0.034
12 sobel_of_gaussian_blur=0.3	0.007	0.000
13 gaussian_blur=2	0.154	0.103
14 difference_of_gaussian=2	0.007	0.017
15 laplace_box_of_gaussian_blur=2	0.035	0.052
16 sobel_of_gaussian_blur=2	0.063	0.017

Strongly relevant

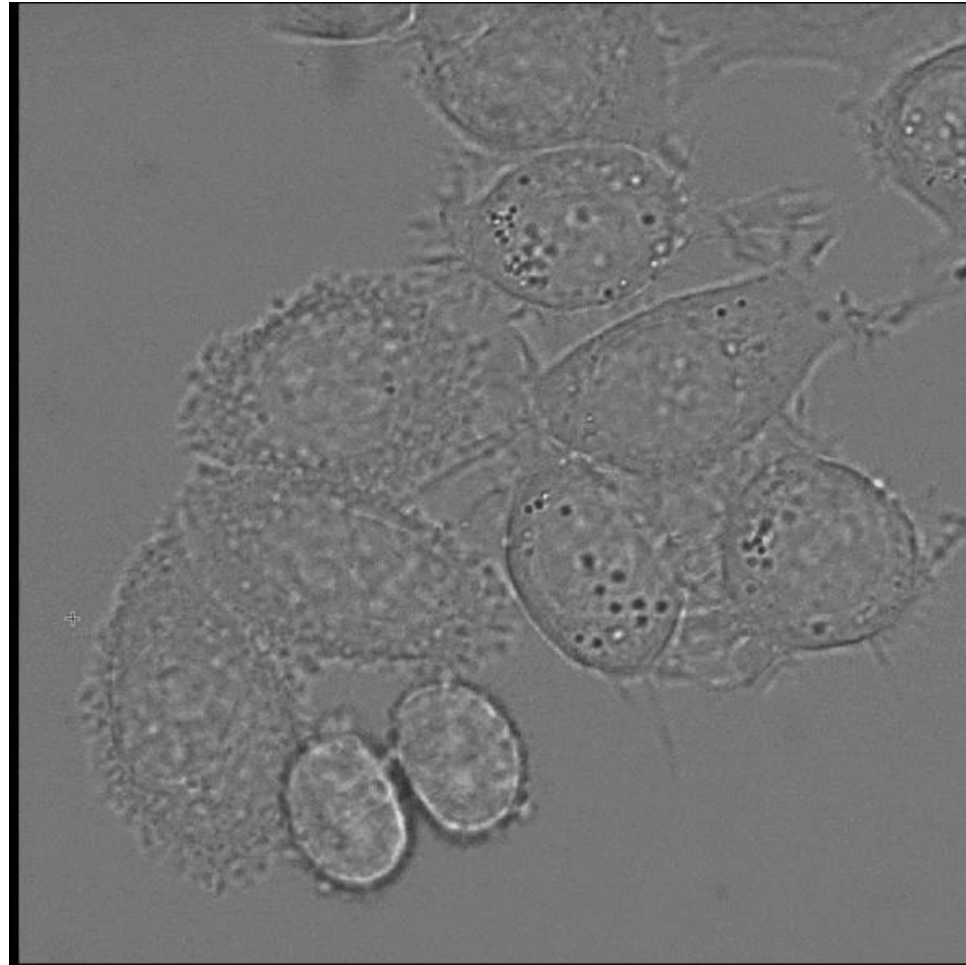
Hardly relevant

<https://imagej.nih.gov/ij/images/>

Segmentation and Supervised Machine Learning in napari

- Demonstration: Micro-sam plugin

Micro-sam



<https://github.com/computational-cell-analytics/micro-sam>



Feature Extraction

- `napari-skimage-regionprops`

Feature extraction

- 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:

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

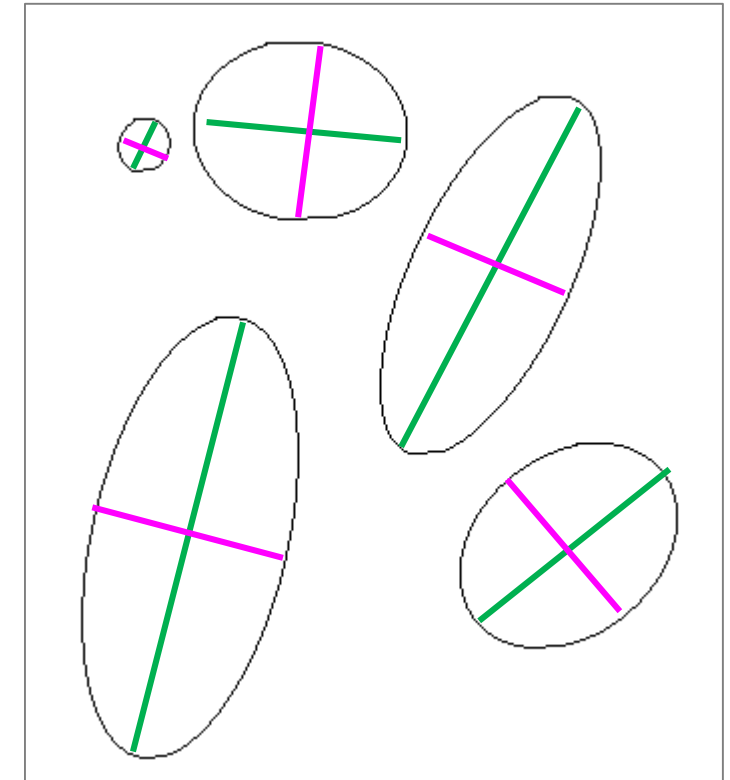
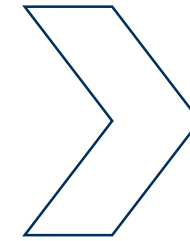
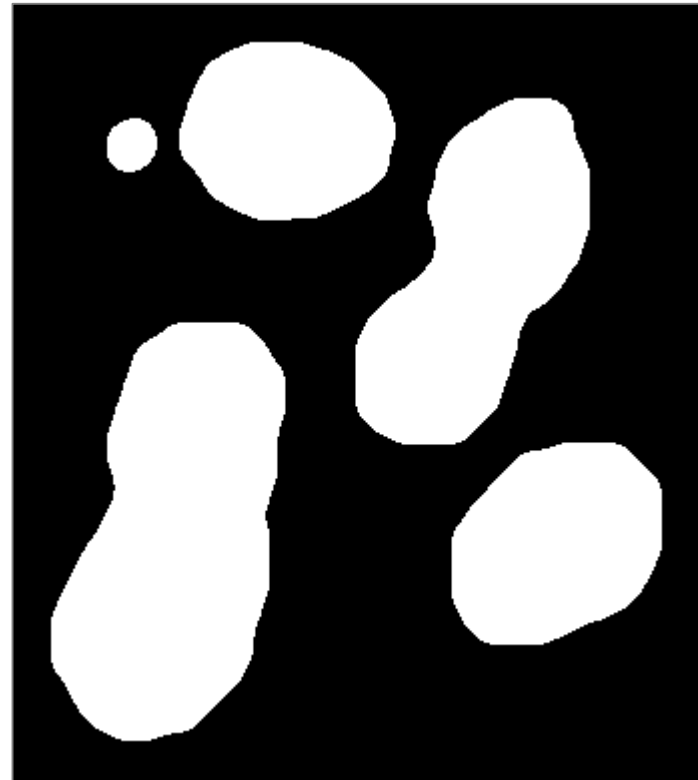
Fit ellipse

For every object, find the optimal ellipse simplifying the object.

Major axis ... long diameter

Minor axis ... short diameter

Major and minor axis are perpendicular to each other



napari-skimage-regionprops

The screenshot displays the napari-skimage-regionprops interface. The main window shows a segmentation map with various colored regions. The left sidebar contains a layer list with 'Segmentation' and 'Image' layers. The top menu bar includes File, View, Layers, Plugins, Window, Tools, and Help. The Tools menu is open, showing options like Filtering, Image math, Measurement maps, and Measurement post-processing. The right sidebar shows the 'Object Features/Properties (skikit-image, nsr)' panel with a table of region properties.

	label	area	bbox area	equivalent diameter	convex area	max intensity	mean intensity	min int
1	1	64.0	77.0	9.0270333367641	66.0	209.0	120.8125	36.0
2	2	108.0	126.0	11.726460285670077	112.0	185.0	107.611111111...	38.0
3	3	281.0	360.0	18.91508160359296	290.0	141.0	86.0747330960...	43.0
4	4	456.0	600.0	24.095585330081406	479.0	233.0	96.2785087719...	42.0
5	5	274.0	342.0	18.677998695187732	285.0	208.0	100.478102189...	39.0
6	6	249.0	380.0	17.805522925178455	267.0	214.0	107.289156626...	38.0
7	7	224.0	270.0	16.88803298257901	230.0	162.0	94.46875	37.0
8	8	278.0	361.0	18.81384047546846	294.0	194.0	96.3669064748...	35.0
9	9	246.0	323.0	17.697935698969246	252.0	165.0	90.3699186991...	40.0
10	10	109.0	130.0	11.780624362746345	113.0	159.0	98.6513761467...	41.0
11	11	194.0	238.0	15.716503163191918	199.0	255.0	124.685567010...	31.0
12	12	32.0	49.0	6.383076486422923	36.0	69.0	54.5625	42.0
13	13	279.0	342.0	18.847647942942654	287.0	177.0	100.906810035...	37.0
14	14	14.0	16.0	4.222008245644752	14.0	105.0	71.3571428571...	40.0
15	15	292.0	361.0	19.281751659604673	303.0	216.0	100.256849315...	37.0
16	16	472.0	783.0	24.514670406003766	502.0	233.0	94.9936440677...	38.0
17	17	219.0	270.0	16.698486766680407	231.0	181.0	92.4611872146...	39.0

use <7> for transform, use <1> for activate the label eraser, use <2> for activate the paint brush, use <3> for activate the polygon tool, use <4> for activate the fill bucket, use <5> for pick mode

<https://github.com/haesleinhuepf/napari-skimage-regionprops>

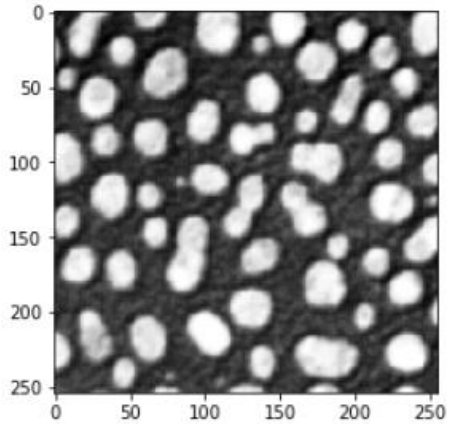
Object Classification and Supervised Machine Learning in napari

Random Forest Classifiers

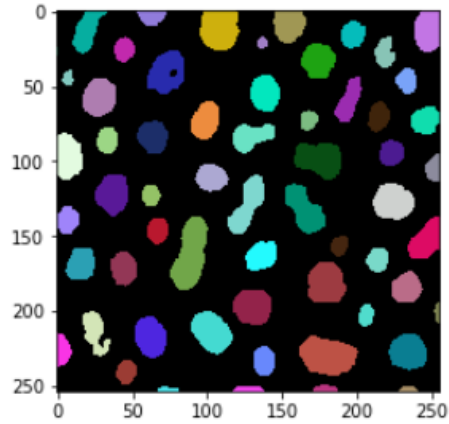
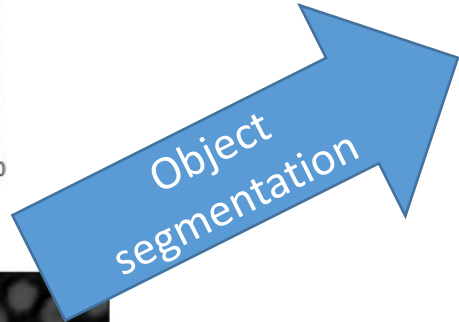
- Object Classifier
- **napari-apoc plugin**

Object classification

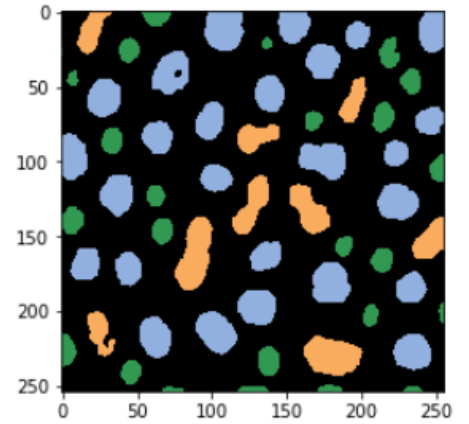
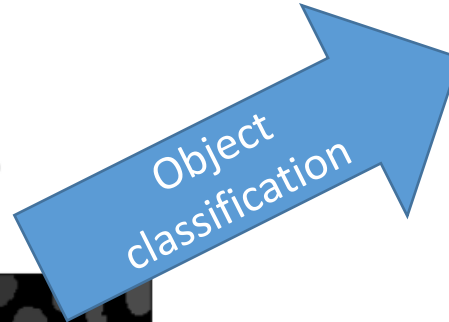
- Object classification is a task that can be applied to an existing instance segmentation in order to identify a particular group of objects



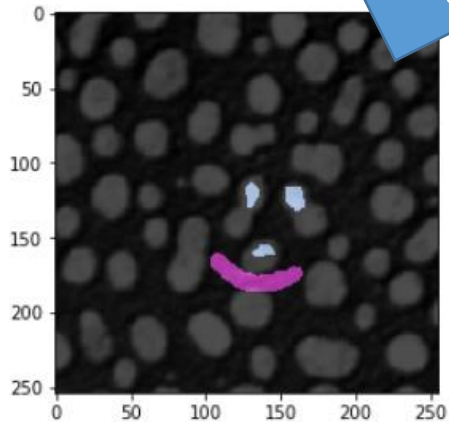
Raw image



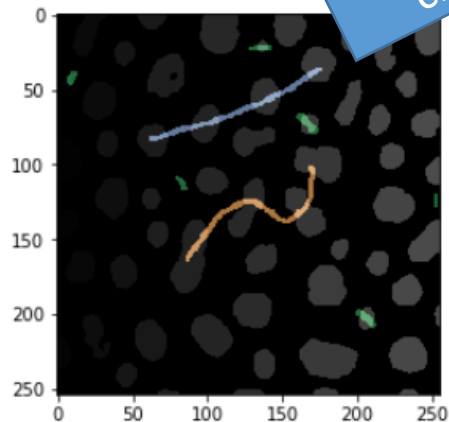
Object label image



Class label image



Pixel annotation



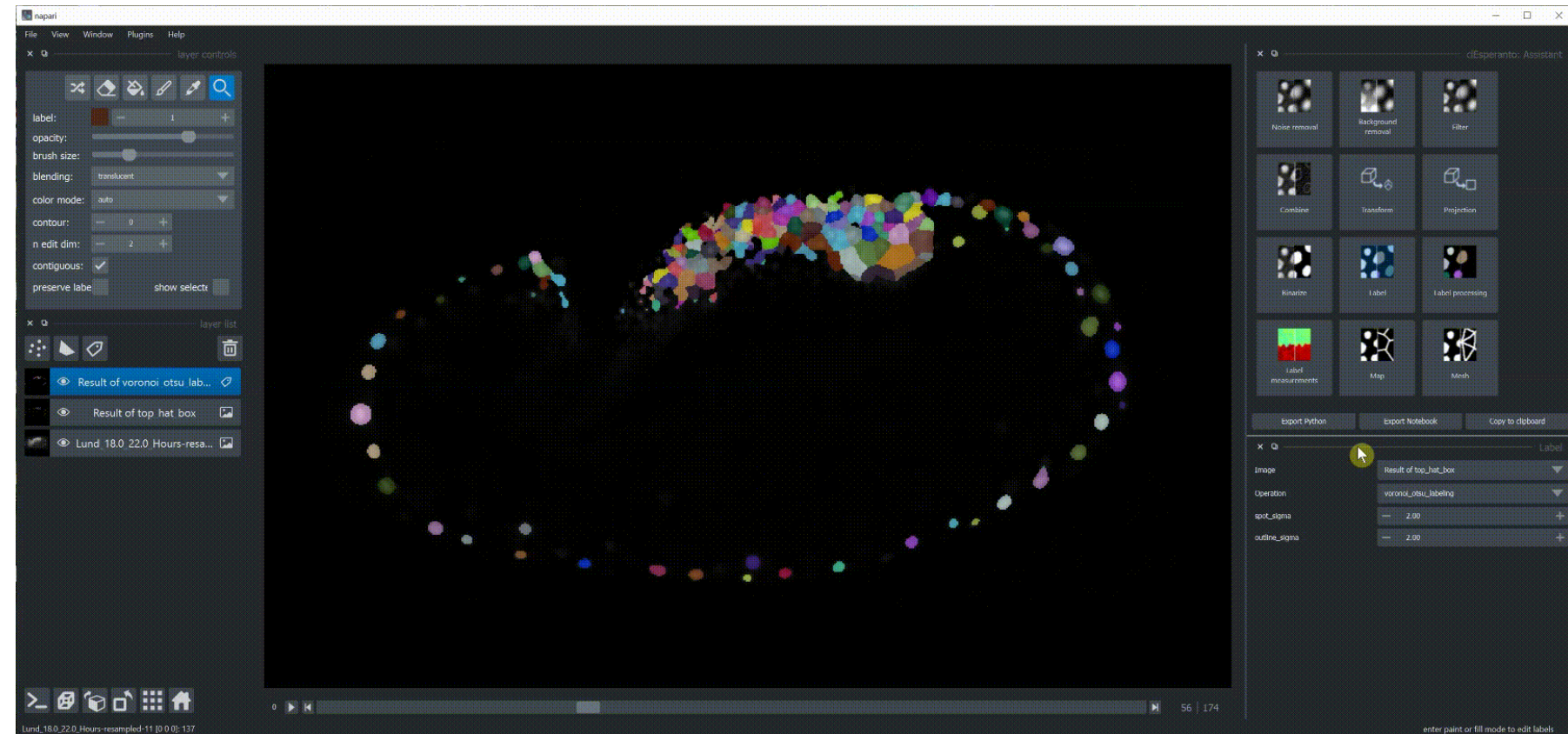
Object annotation

<https://imagej.nih.gov/ij/images/>

Object classification

Random Forest Classifiers based on

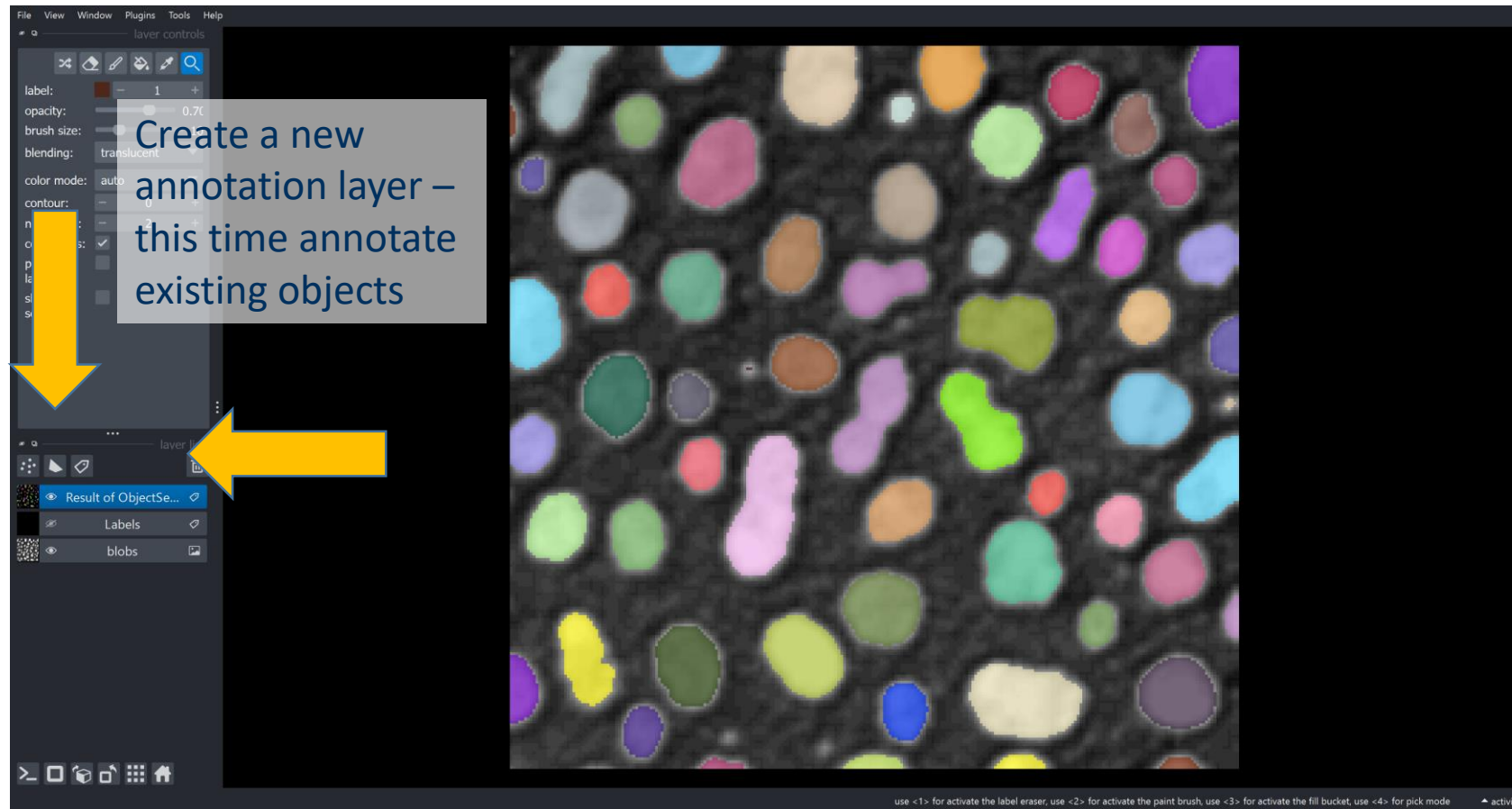
- scikit-learn and
- clesperanto



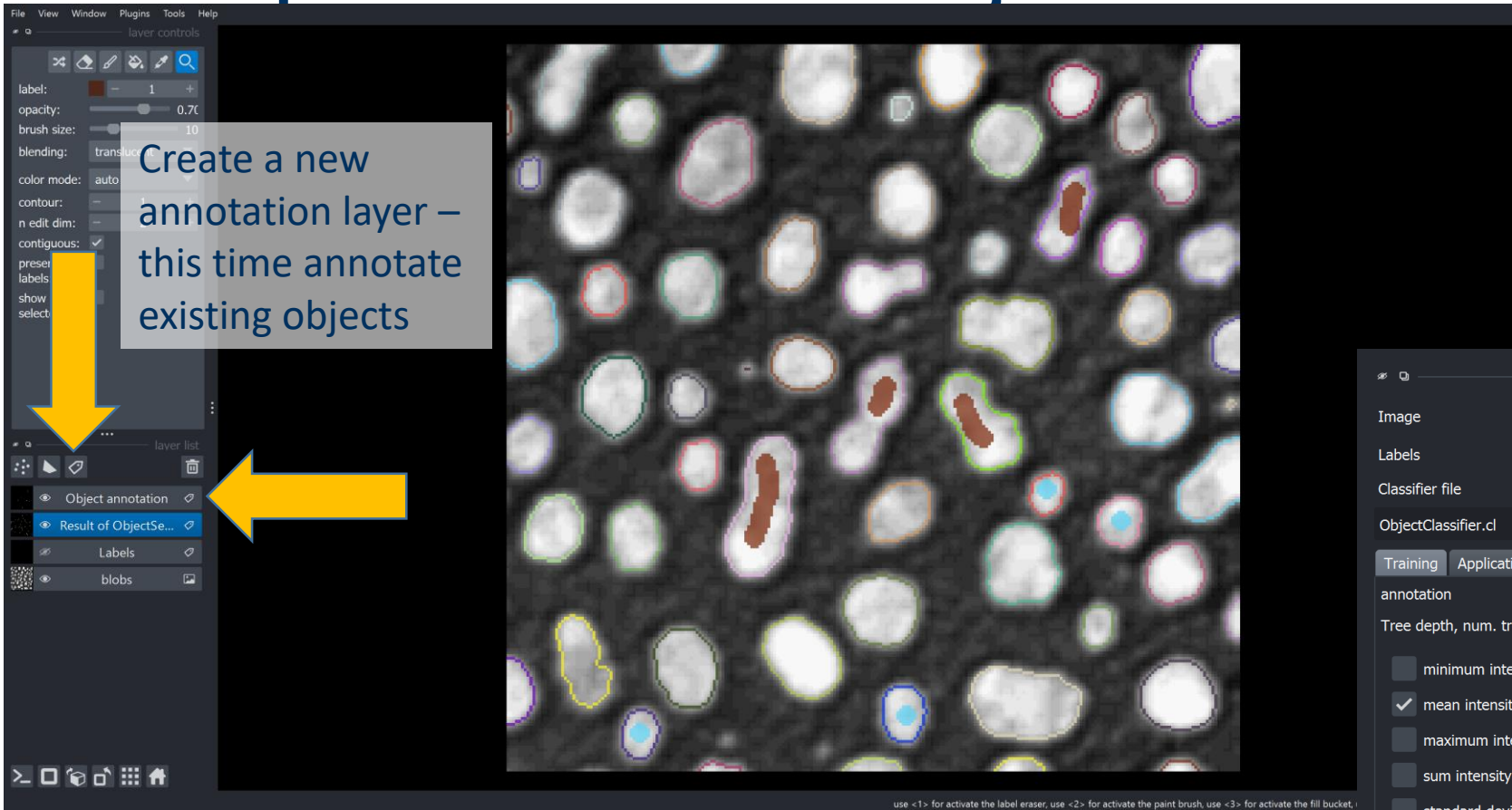
<https://github.com/haesleinhuepf/napari-accelerated-pixel-and-object-classification>

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

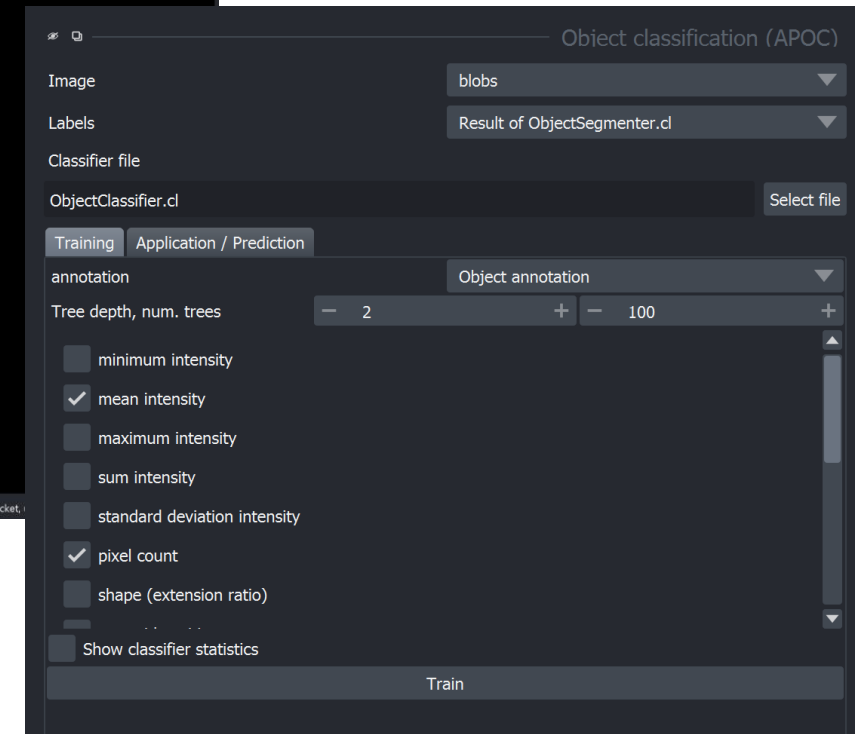
In napari: annotation



In napari: annotation and object classification



Tools ->
Segmentation post-
processing ->
Object Classification (APOC)



Telling different classes of objects apart requires:

- An **annotation** for some example objects
- **Features** for each objects upon which to make a prediction

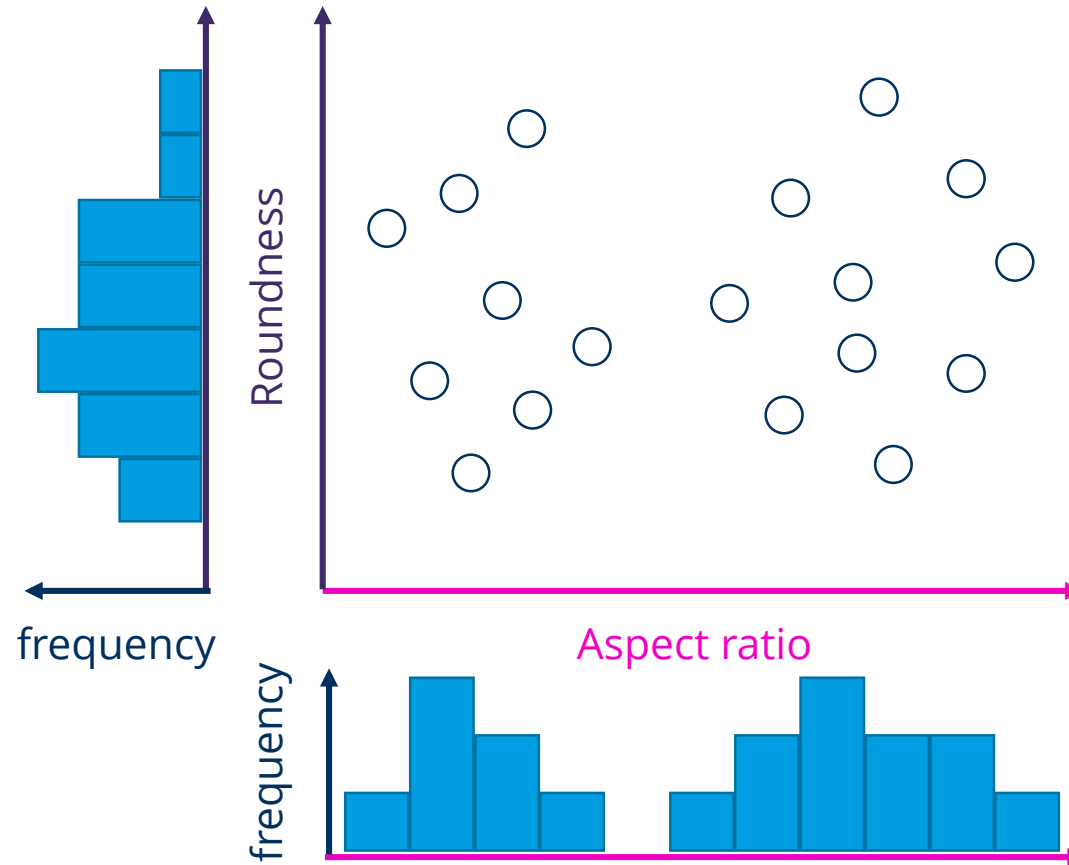
<https://imagej.nih.gov/ij/images/>

Object Classification and Unsupervised Machine Learning

- Dimensionality Reduction
- Clustering
- **napari-clusters-plotter plugin**

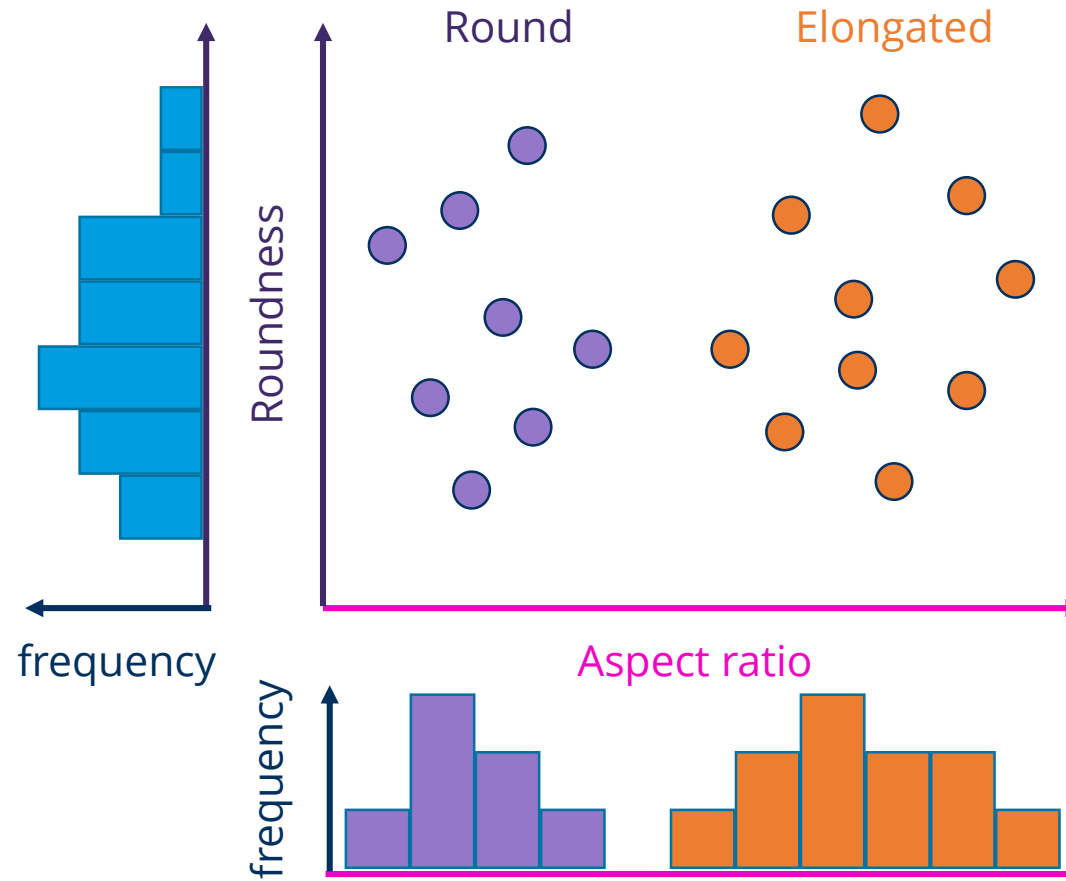
Unsupervised machine learning

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

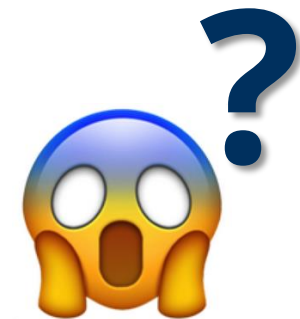


Unsupervised machine learning

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



- Mean intensity
- Standard deviation
- Total intensity
- Textures
- Area / Volume
- Roundness
- Solidity
- Circularity / Sphericity
- Elongation
- Centroid
- Bounding box
- ...



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

x

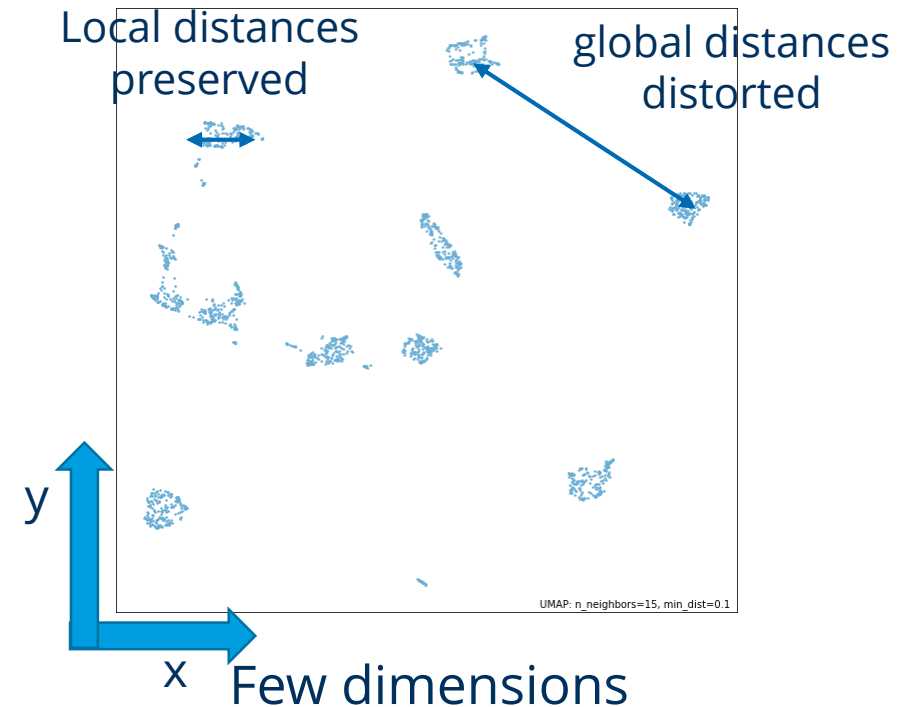
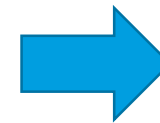
y

z

...

	label	area	bbox_area	convex_area	equivalent_diamete	max_intensity	mean_intensity	min_intensity	solidity	extent	eret_diameter_ma	local_centroid-0
1	1	3379	13949	5120	18.61786412639...	613.0	345.6717963894...	259.0	0.6599609375	0....	37.3496987939662	15.77952056821...
2	2	2319	7448	3491	16.42230229224...	421.0	297.8434670116...	240.0	0....	0....	38.65229618017...	4....
3	3	2304	14415	4281	16.38681751812...	456.0	300.8298611111...	245.0	0....	0....	34.19064199455...	17.73828125
4	4	3278	13804	5139	18.43048549951...	467.0	316.1446003660...	249.0	0....	0....	34.84250278036...	15.52287980475...
5	5	1501	3315	1681	14.20563625190...	458.0	302.147235176549	236.0	0....	0....	17.97220075561...	6....
6	6	2341	6061	2714	16.47407088948...	594.0	355.4446817599...	261.0	0....	0....	30.67572330035...	16.54250320375...
7	7	1725	3584	1940	14.87979081163...	568.0	343.7866666666...	257.0	0....	0....	17.72004514666...	7.80463768115942
8	8	1502	3840	1753	14.20879025650...	431.0	290.0659121171...	235.0	0....	0....	18.57417562100...	8....
9	9	1602	4080	1894	14.51737058294...	475.0	297.8008739076...	241.0	0....	0....	18.70828693386...	8....
10	10	1395	3600	1624	13.86304166283...	424.0	304.8494623655...	247.0	0....	0.3875	17.60681686165...	7....
11	11	609	1100	697	10.51654029260...	323.0	274.2528735632...	241.0	0....	0....	13.45362404707...	3....
12	12	1686	3757	1894	14.76679738567...	460.0	303.8303677342...	240.0	0....	0....	17.97220075561...	9....
13	13	2157	5184	2531	16.03062694504...	576.0	339.990264255911	270.0	0....	0....	19.54482028569...	8....
14	14	863	2340	1032	11.81237949737...	327.0	272.4449594438...	237.0	0....	0....	16.0312195418814	6....

Many dimensions

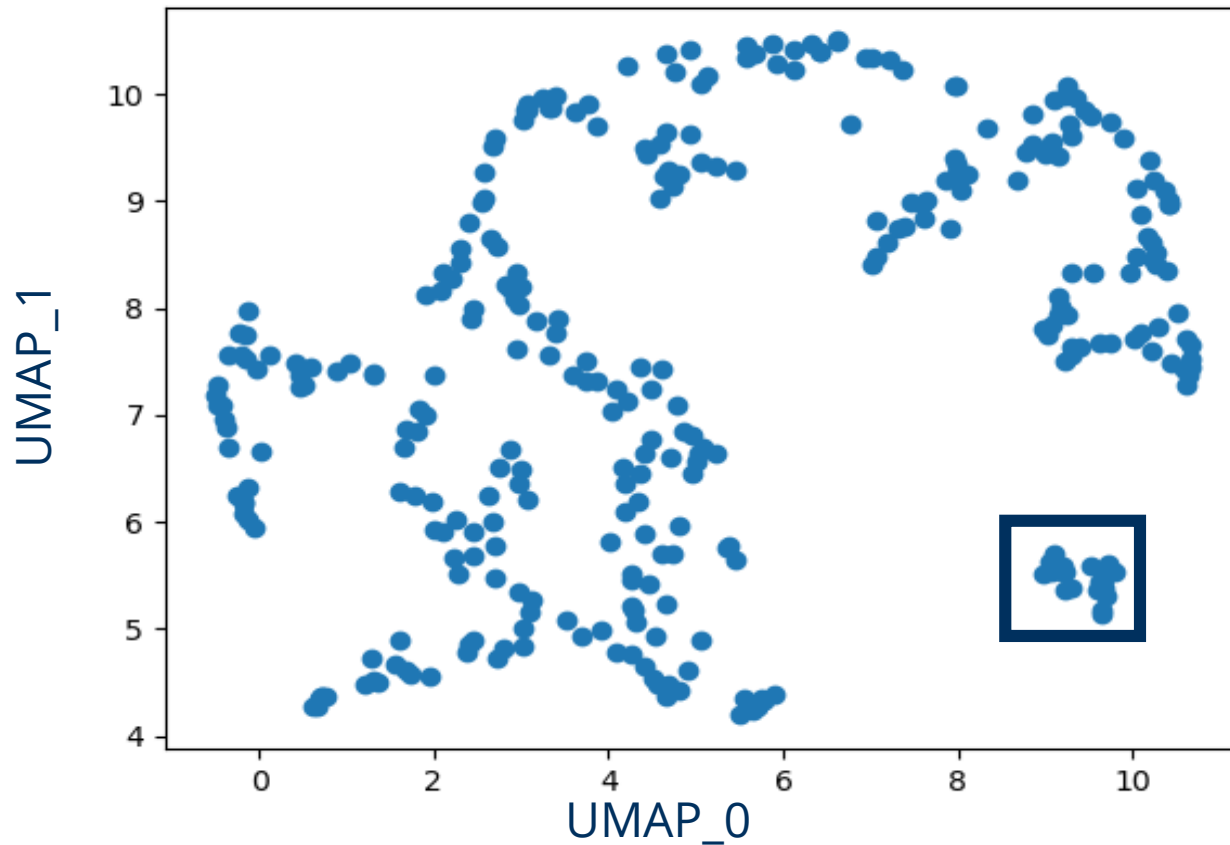


<https://umap-learn.readthedocs.io/en/latest/index.html>

Clustering

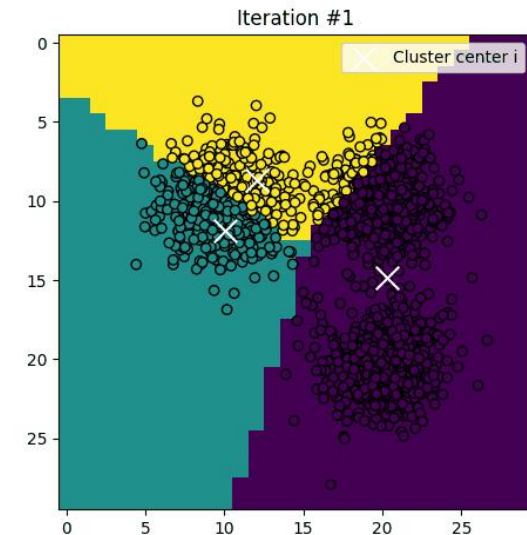
Starting point: Feature space or dimensionality reduction reveals “groups” in our data

Can we automatically identify these groups?



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

K-means clustering

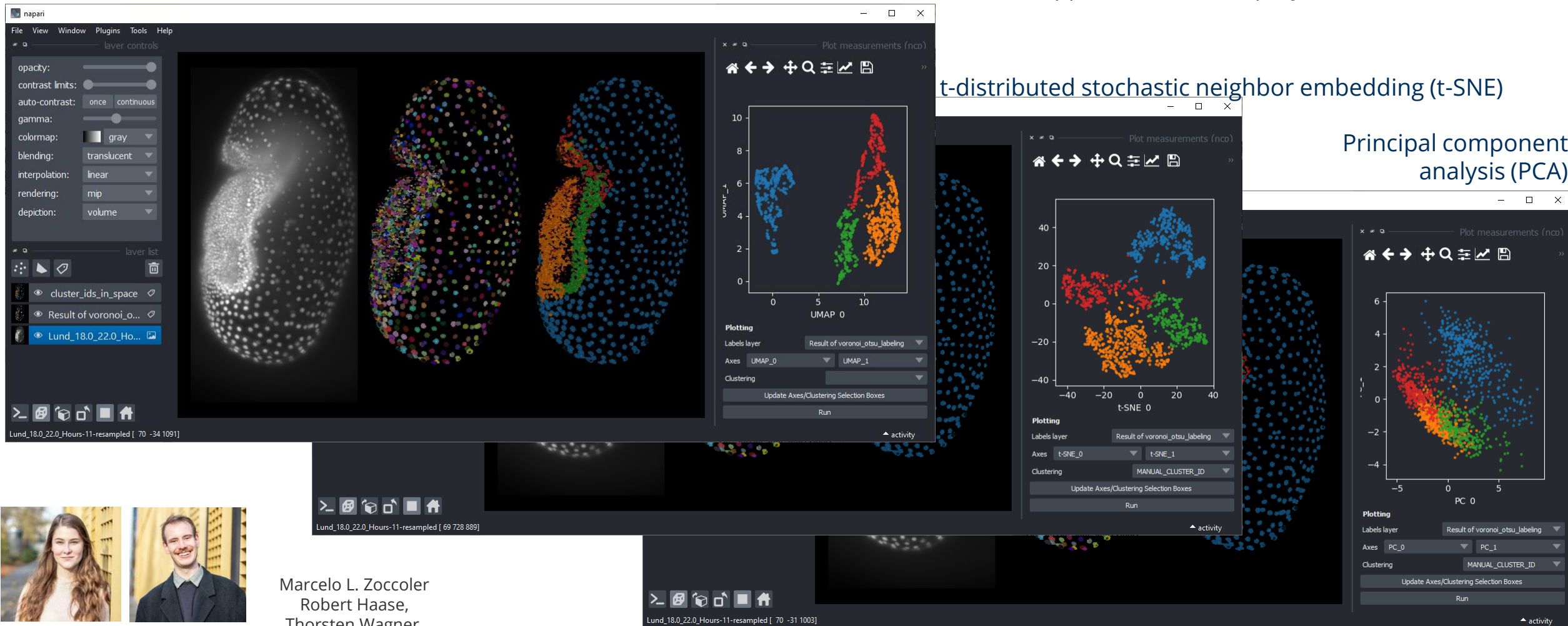


Dimensionality reduction

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

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

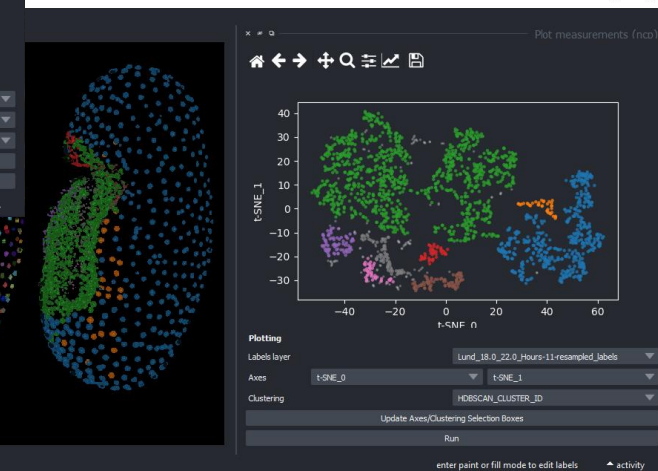
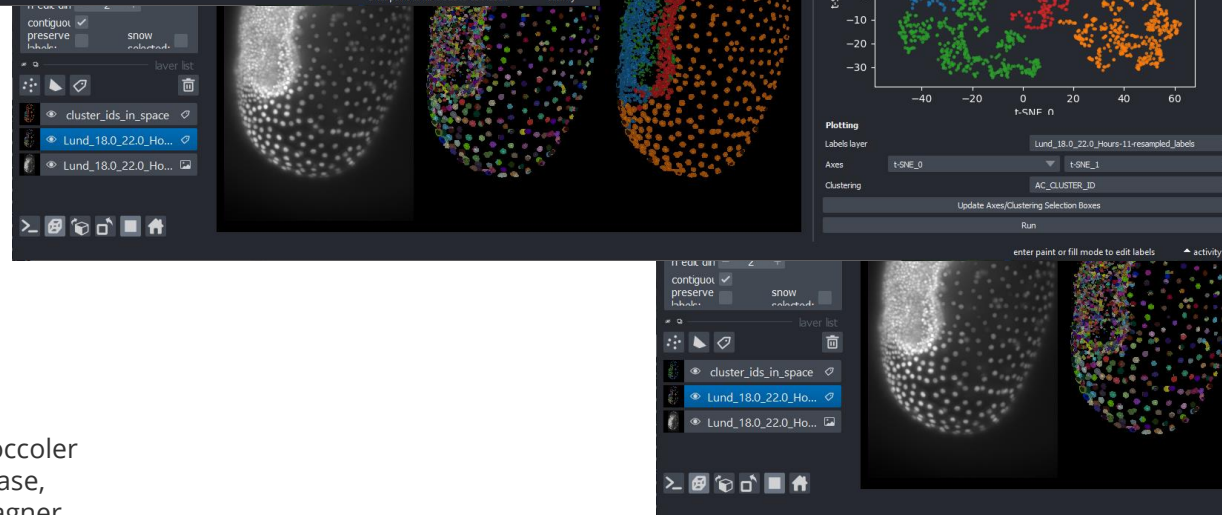
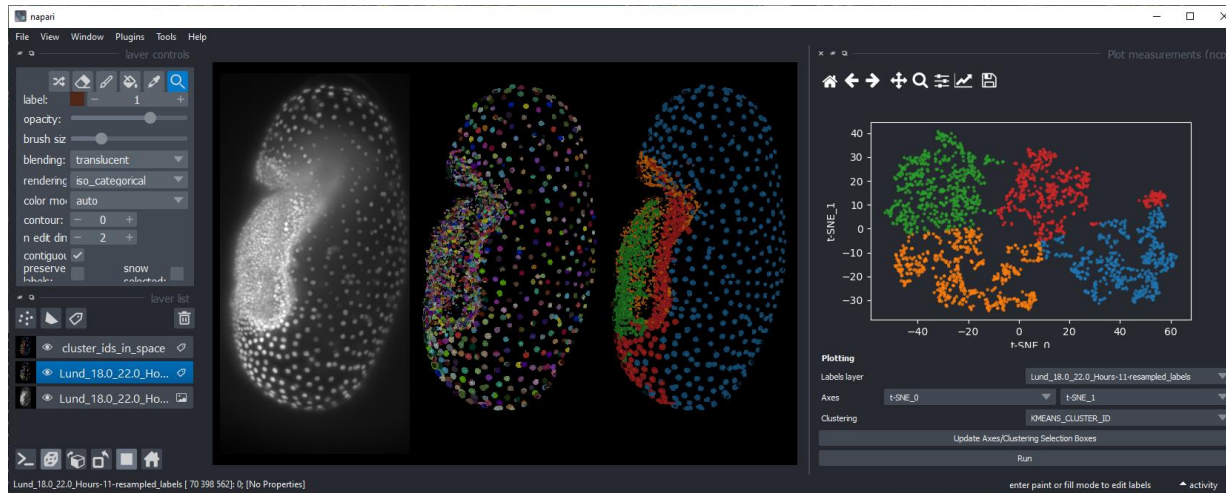
Image Data Source: Daniela Vorkel, Myers lab, MPI-CBG/CSBD Dresden

Clustering

K-means clustering

Agglomerative clustering

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



Laura Žigutyte
@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

Multi-layer
analysis
coming soon
in 0.9.0!

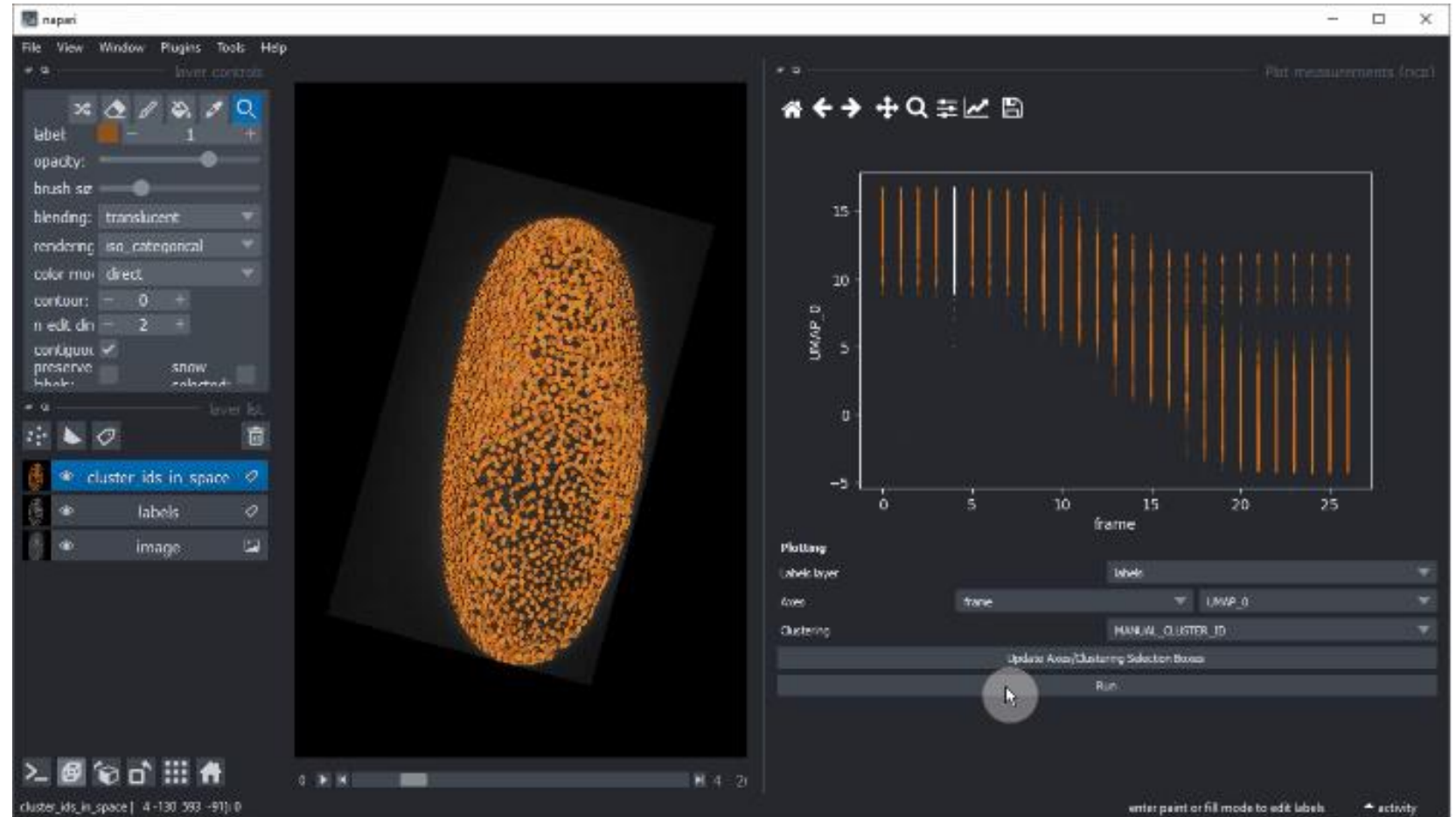


Laura Žigutytė
@zigutyte



Ryan Savill
@RyanSavill4

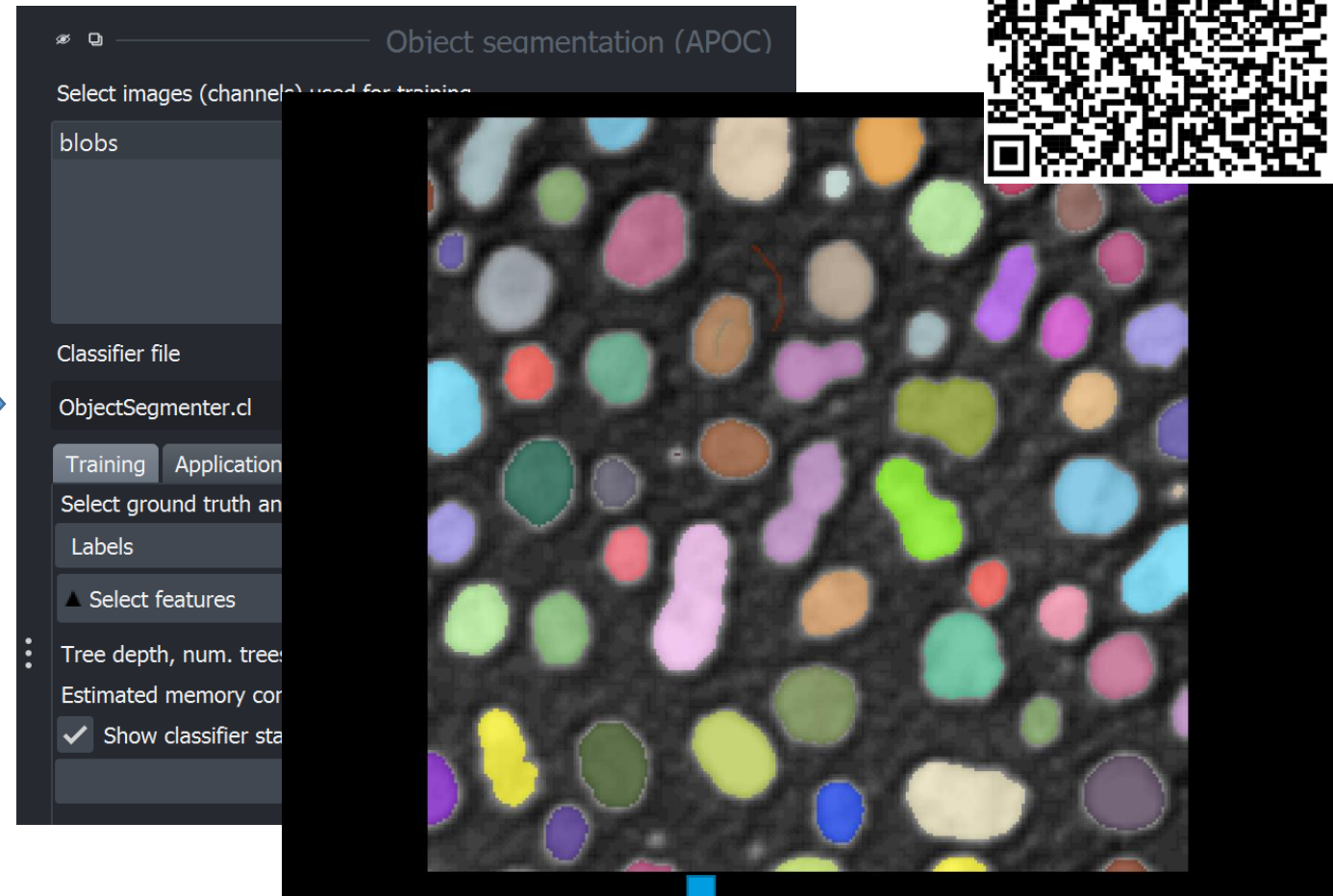
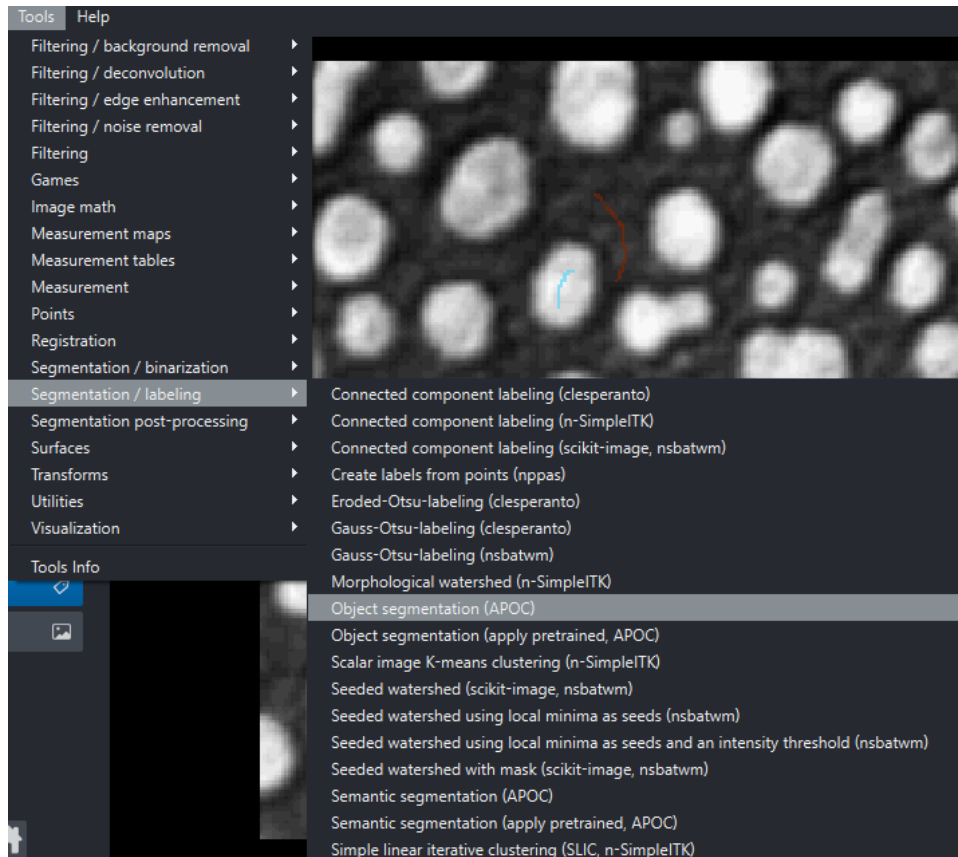
Marcelo L. Zoccoler
Robert Haase,
Thorsten Wagner,
Johannes Soltwedel



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

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

Exercise: Object segmentation



<https://imagej.nih.gov/ij/images/>

https://biapol.github.io/AMHCT_Bio_Image_Analysis_2025/interactive_pixel_classification/readme.html

1. Activate the environment and open napari
mamba activate amhct
napari
2. Open the blobs image
3. Perform object segmentation on blobs sample dataset

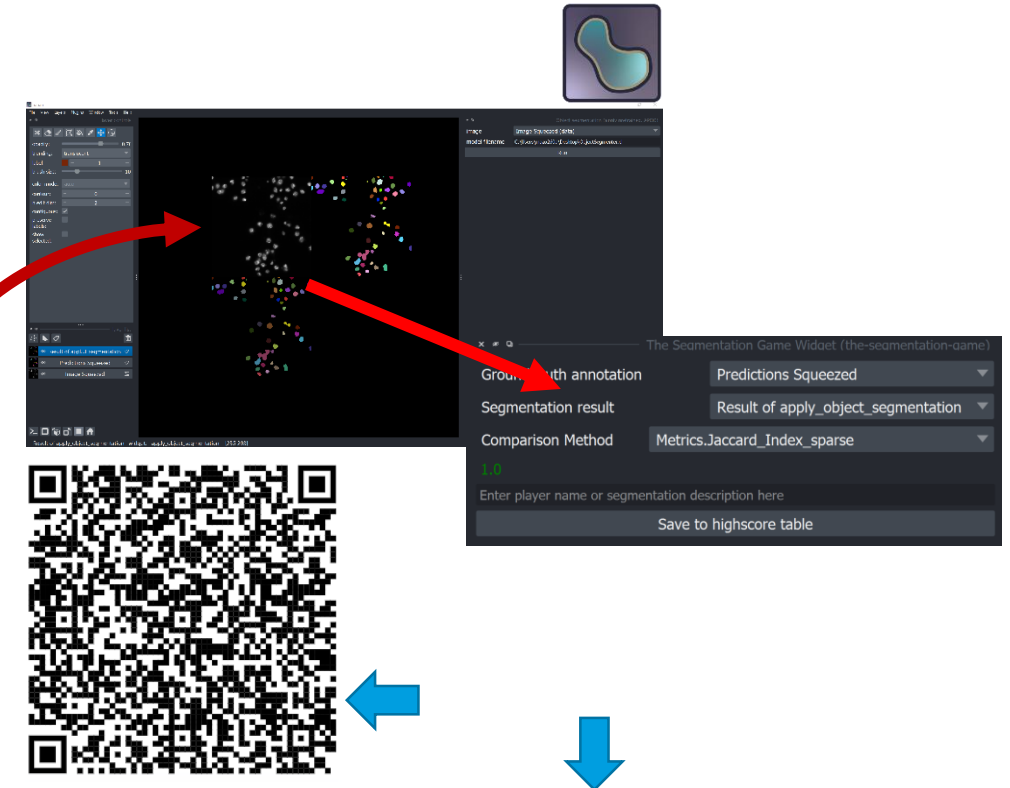
Exercise: Recording and Reproducing a Bio-Image Analysis Workflow

Creating a workflow:



https://biapol.github.io/AMHCT_Bio_Image_Analysis_2025/napari_apoc_omero_workflow/workflow_creation.html

Reproducing your colleague's workflow:



https://biapol.github.io/AMHCT_Bio_Image_Analysis_2025/napari_apoc_omero_workflow/workflow_reproduction.html

Discussion

- How many succeeded in **creating** the workflow?
- How many succeeded in **reproducing** the workflow?
- Which were key items to ensure reproducibility?
- Which part was the hardest?
- What could hamper reproducibility?

Further resources:

Napari-hub:

www.napari-hub.org

BiaPoL image analysis course materials:

https://github.com/BiAPoL/Bio-image_Analysis_with_Python

Bio-image analysis materials:

<https://bioimagebook.github.io/README.html>

<https://haesleinhuepf.github.io/BioImageAnalysisNotebooks/intro.html>

Bio-Image Workflow Tools:

<https://galaxyproject.org/>

<https://github.com/nextflow-io/nextflow>

Image Science Community Forum:

<https://forum.image.sc/>

Acknowledgements

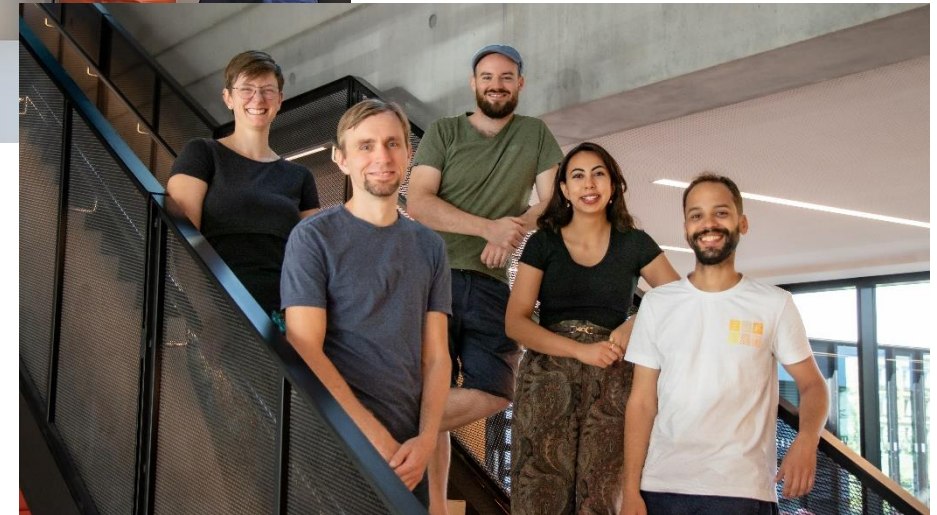


BiAPoL team

- Marcelo Zoccoler
- Johannes Soltwedel
- Stefan Hahmann

Former lab members:

- Robert Haase
- Allyson Ryan
- Till Korten
- Mara Lampert
- Svetlana Iarovenko
- Ryan George Savill
- Laura Zigutyte
- Somashekhar Kulkarni
- Maleeha Hassan
- Tina Smejka



Networks



Funding

