Whole-Slide Image Analysis with open-source software QuPath

Thierry Pécot
Research Engineer
FAIIA
Biosit SFR UMS CNRS 3480 – Inserm 018

Whole-Slide Image Size

- A **slide size** is generally **25 mm x 75 mm**

Whole-Slide Image Size

- A **slide size** is generally **25 mm x 75 mm**
- A **20X** magnification gives a pixel width of **0.45 µm**

WHOLE-SLIDE IMAGE SIZE

- A **slide size** is generally **25 mm x 75 mm**
- A **20X** magnification gives a pixel width of **0.45 µm**
- If the **whole-slide image** is **scanned**:
    - (25e-3 * 75e-3) / (0.45e-6)^2 = **9 259 259 259 pixels**

Whole-Slide Image Size

- A **slide size** is generally **25 mm x 75 mm**
- A **20X** magnification gives a pixel width of **0.45 µm**
- If the **whole-slide image** is **scanned**:
  - (25e-3 * 75e-3) / (0.45e-6)^2 = **9 259 259 259 pixels**
- For a **whole-slide image** with **1 channel** of fluorescence:
  - 9 259 259 259 * 1 byte = **9.26 GB**
- For a **H&E(S) whole-slide image** with 3 channels (RGB):
  - 9 259 259 259 * 1 byte * 3 = **27.78 GB**

Whole-Slide Image Size

- A **slide size** is generally **25 mm x 75 mm**
- A **20X** magnification gives a pixel width of **0.45 µm**
- If the **whole-slide image** is **scanned**:
    - (25e-3 * 75e-3) / (0.45e-6)^2 = **9 259 259 259 pixels**
- For a **whole-slide image** with **1 channel** of fluorescence:
    - 9 259 259 259 * 1 byte = **9.26 GB**
- For a **H&E(S) whole-slide image** with 3 channels (RGB):
    - 9 259 259 259 * 1 byte * 3 = **27.78 GB**

⚠ **not possible** to load the whole data in the **RAM memory**

WHOLE-SLIDE IMAGE SIZE

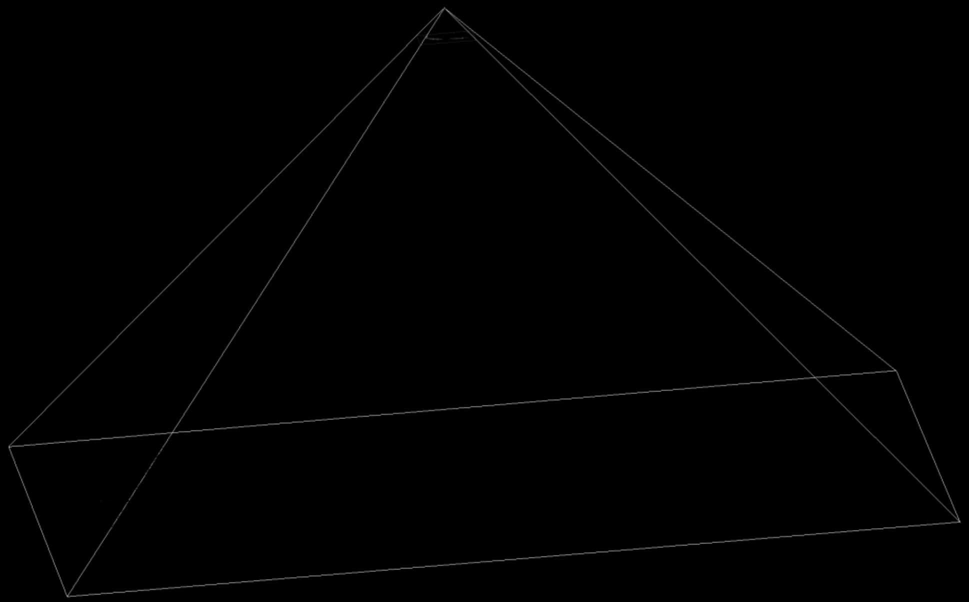
- A **slide size** is generally **25 mm x 75 mm**
- A **20X** magnification gives a pixel width of **0.45 µm**
- If the **whole-slide image** is **scanned**:
    - (25e-3 * 75e-3) / (0.45e-6)^2 = **9 259 259 259 pixels**
- For a **whole-slide image** with **1 channel** of fluorescence:
    - 9 259 259 259 * 1 byte = **9.26 GB**
- For a **H&E(S) whole-slide image** with 3 channels (RGB):
    - 9 259 259 259 * 1 byte * 3 = **27.78 GB**

⚠ **not possible** to load the whole data in the **RAM memory**

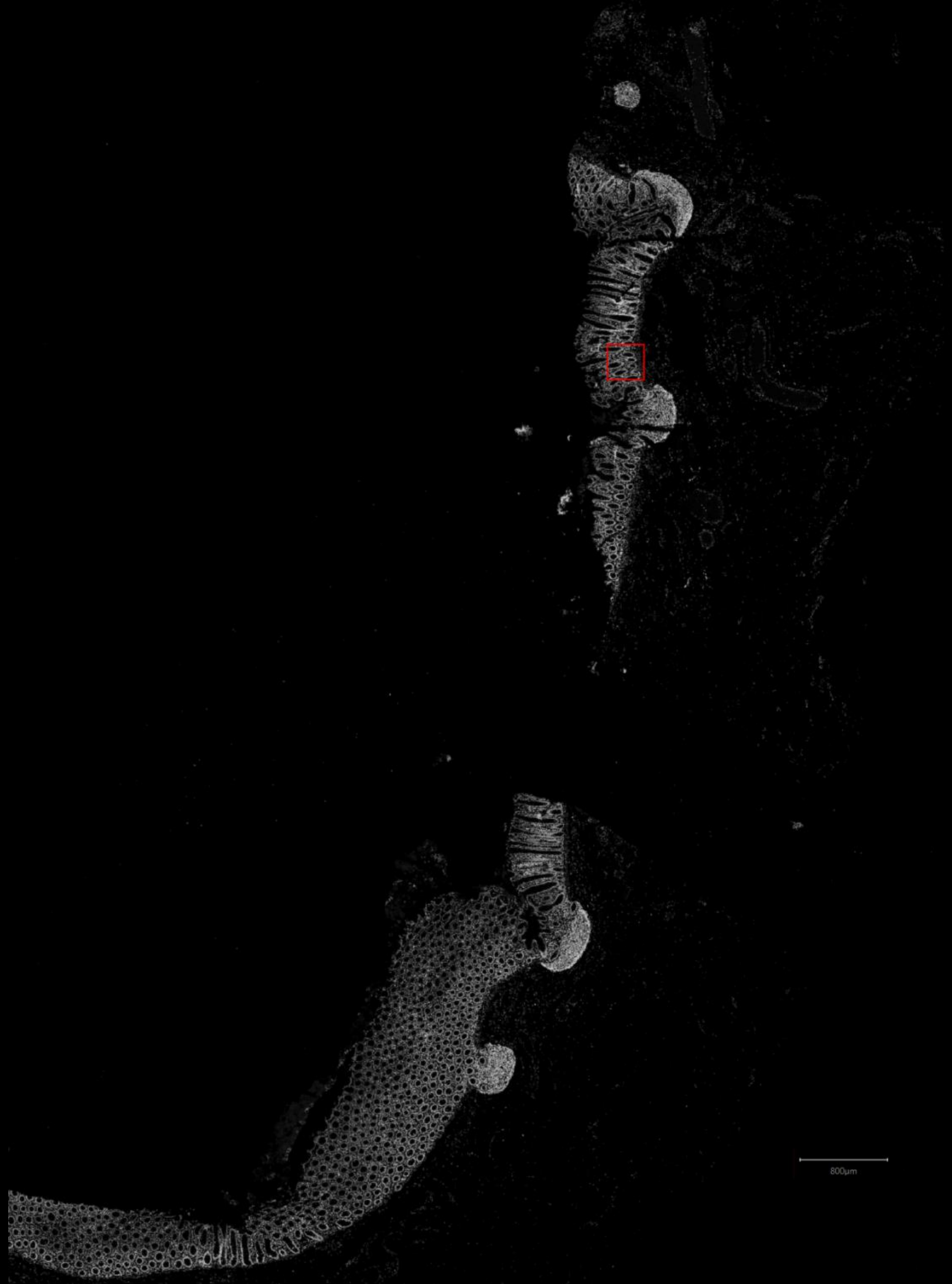
💡 **only** load the **required information** in the RAM memory:

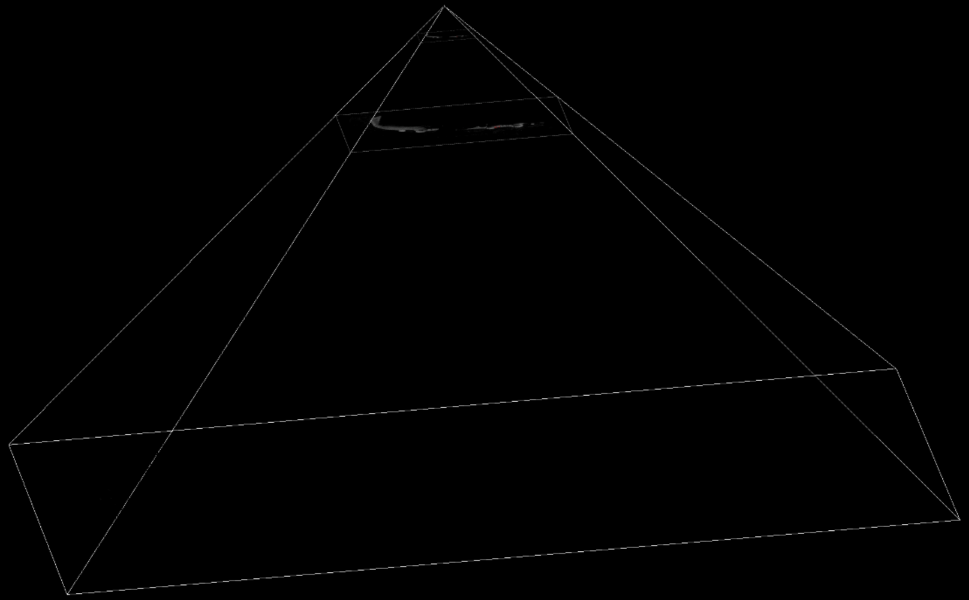
- Define **several resolutions** to create a **pyramidal representation**

Whole Slide Image – Pyramidal Representation

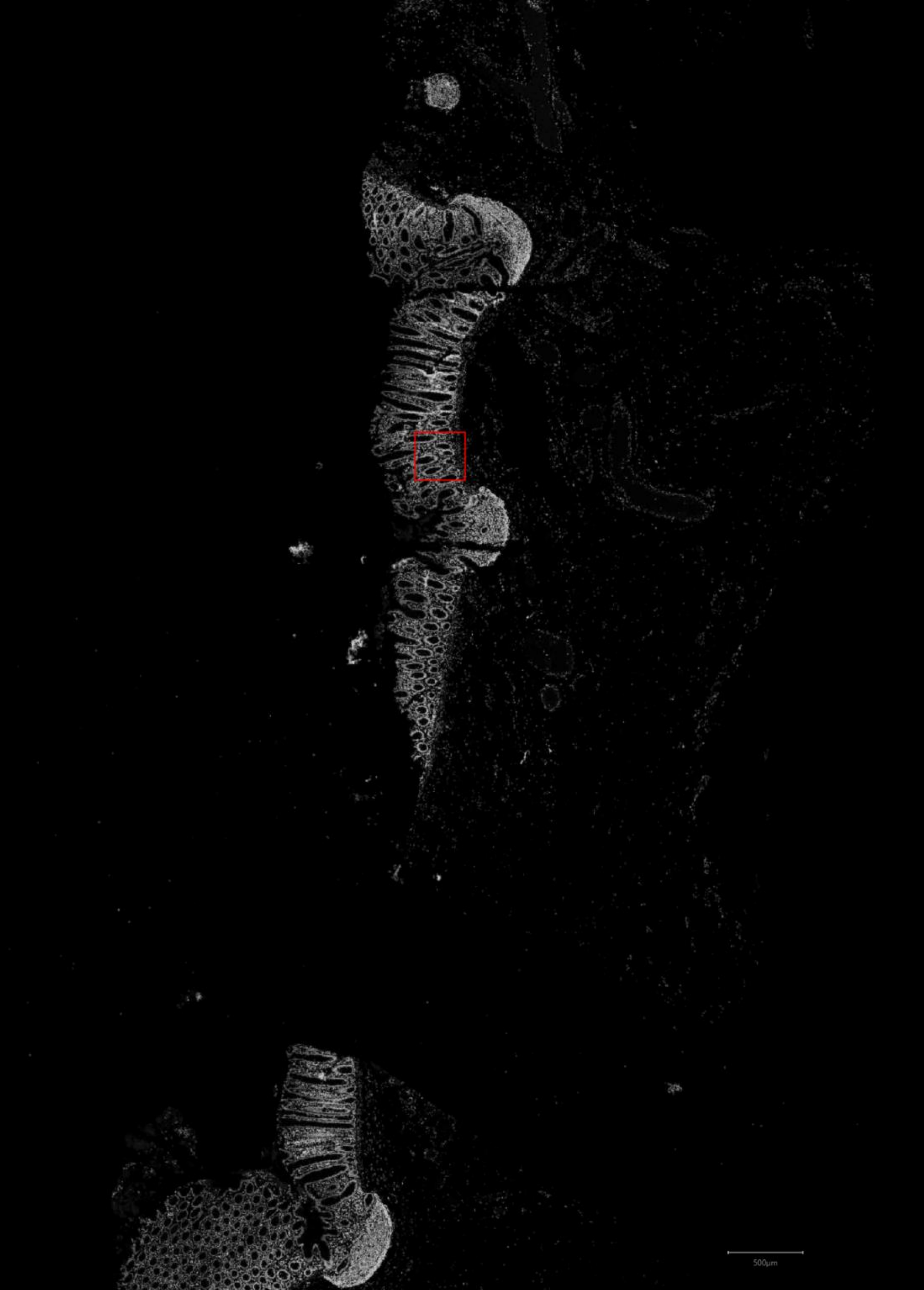


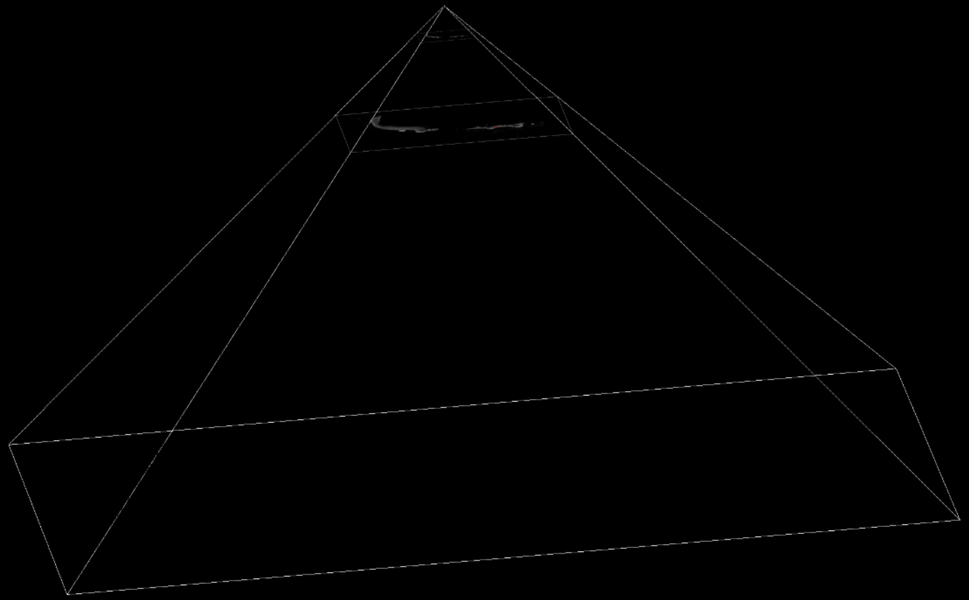
Whole Slide Image – Pyramidal Representation



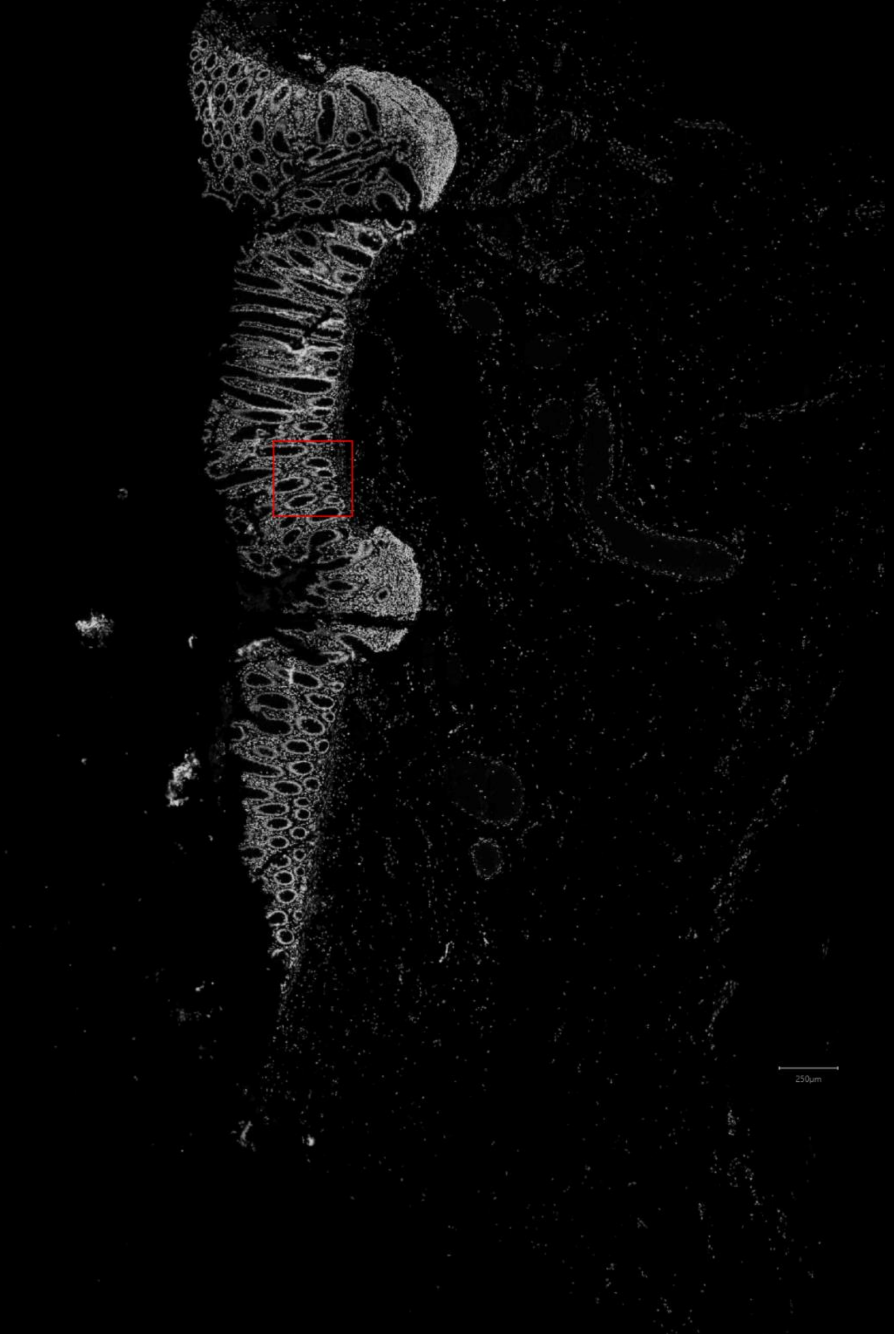


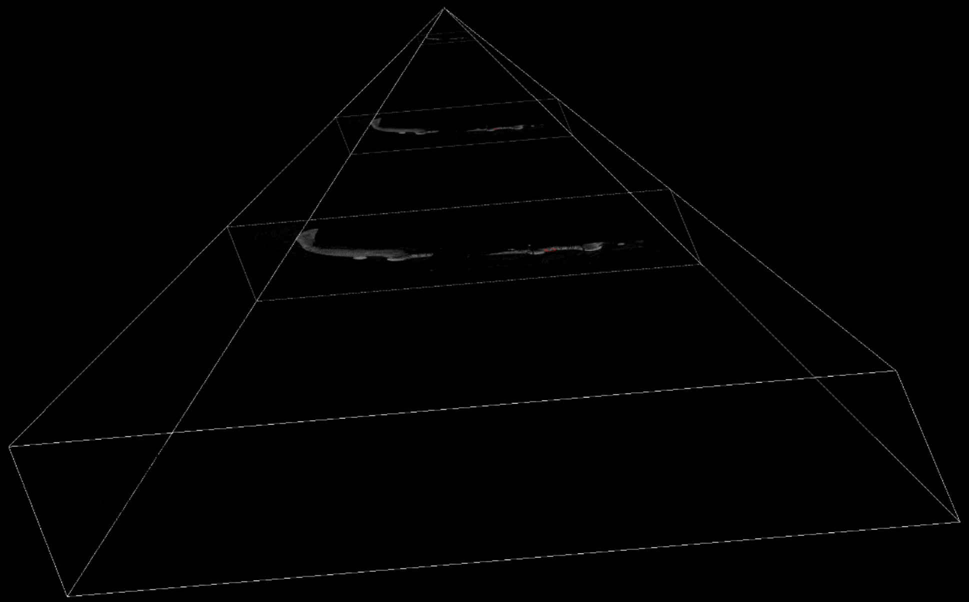
Whole Slide Image – Pyramidal Representation



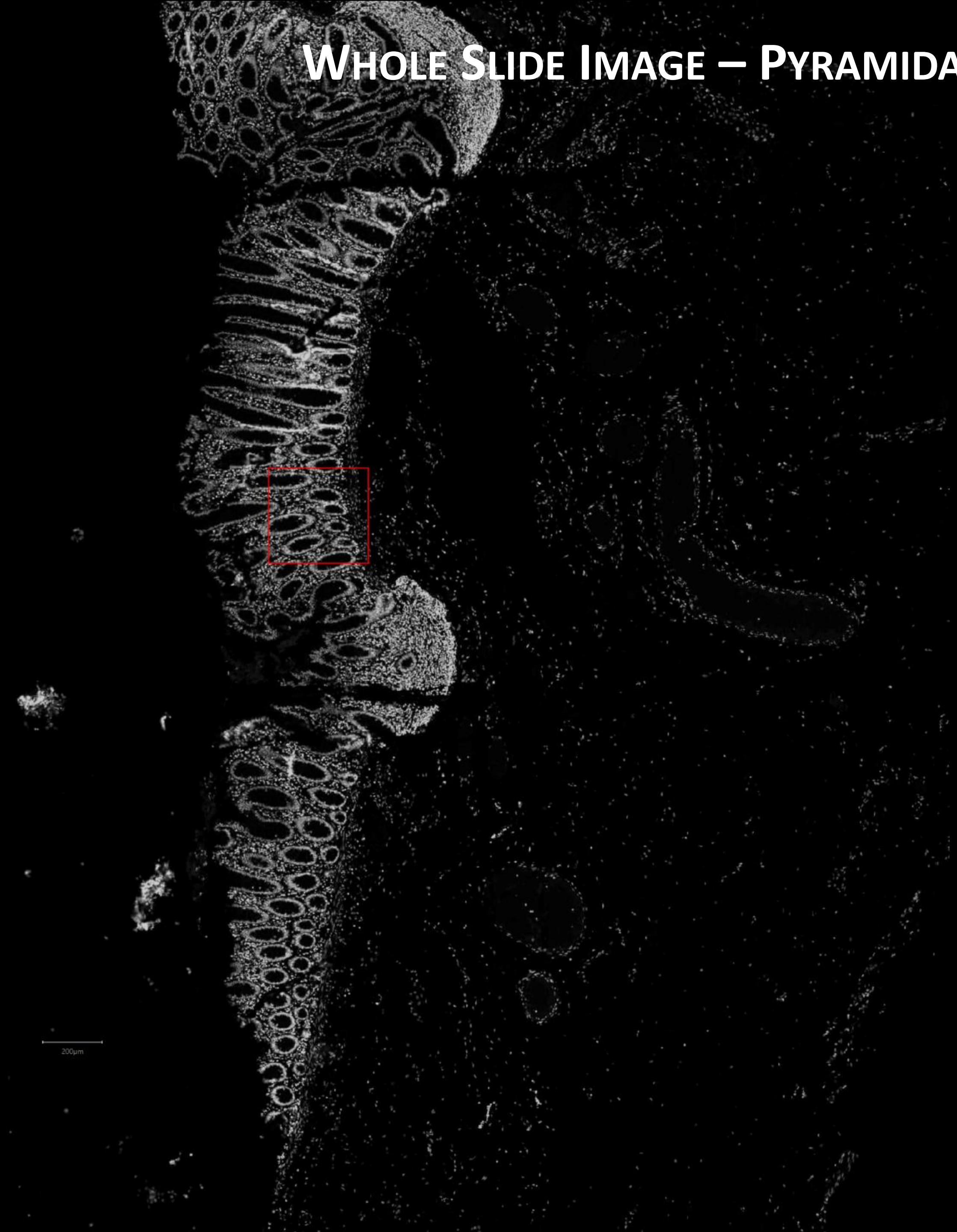


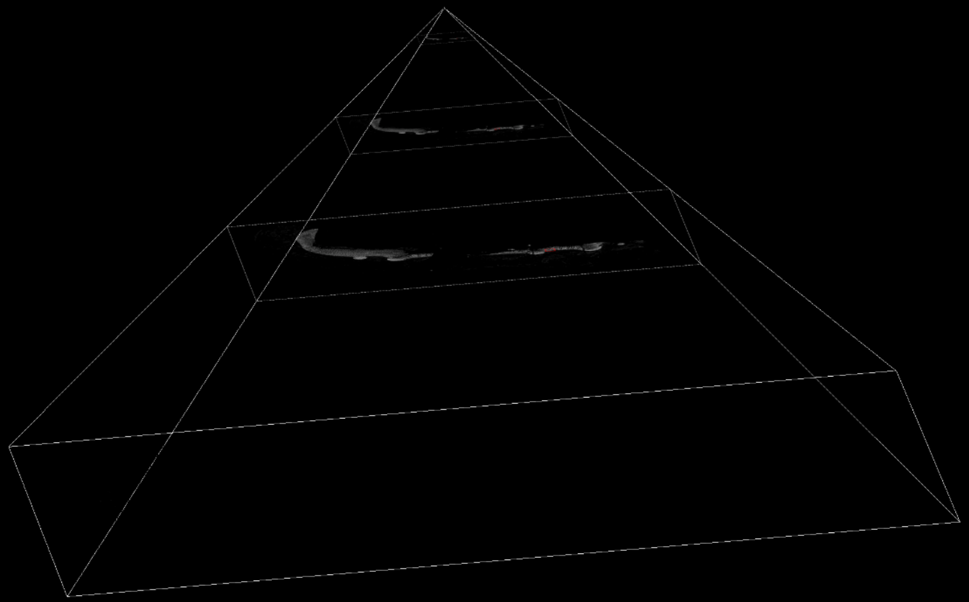
Whole Slide Image – Pyramidal Representation



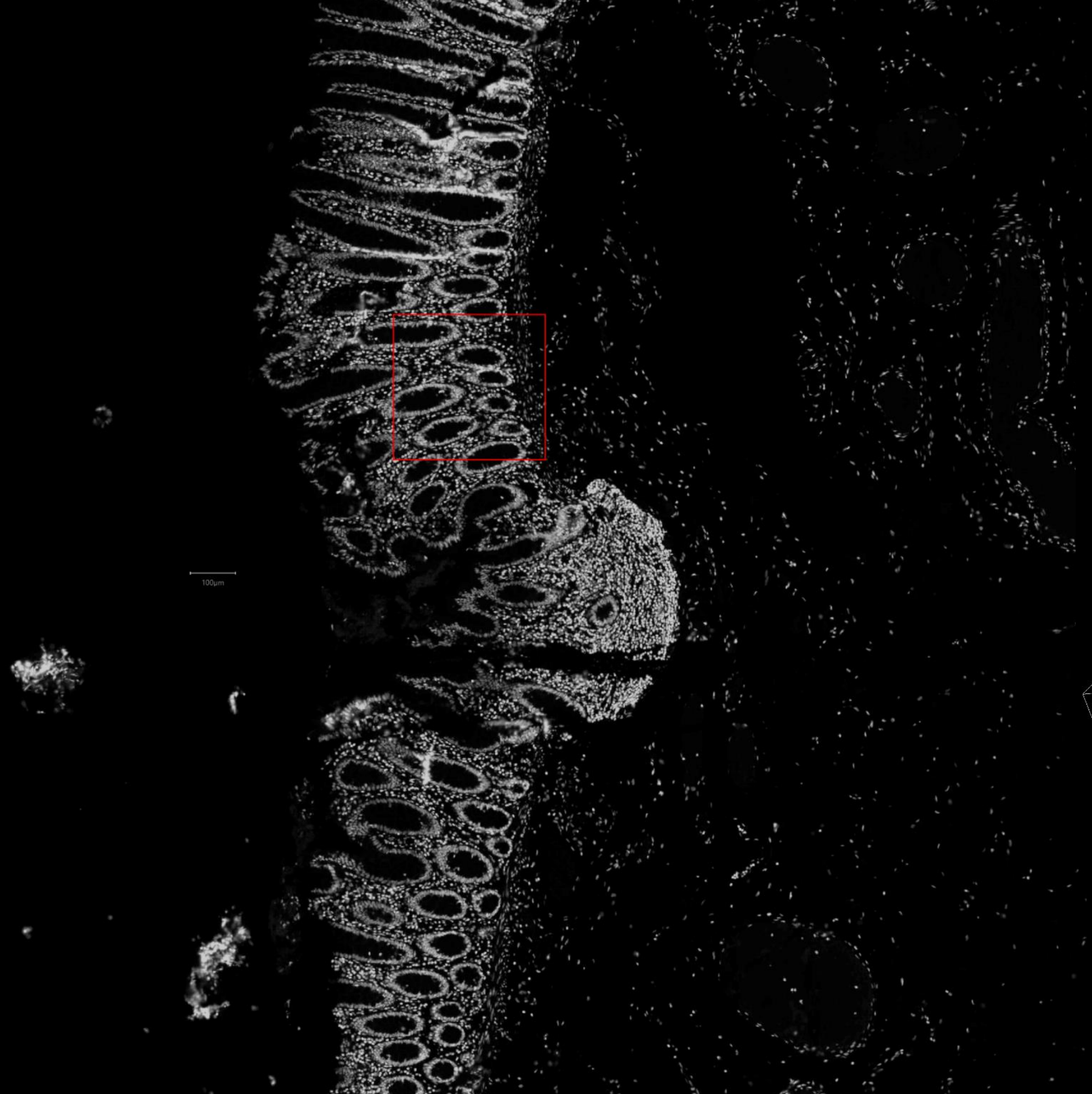


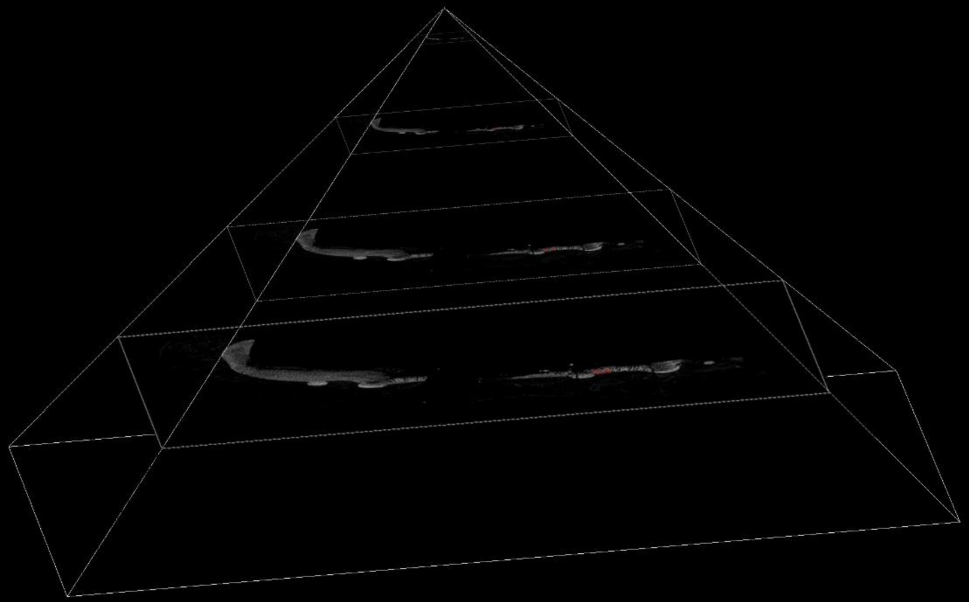
Whole Slide Image – Pyramidal Representation



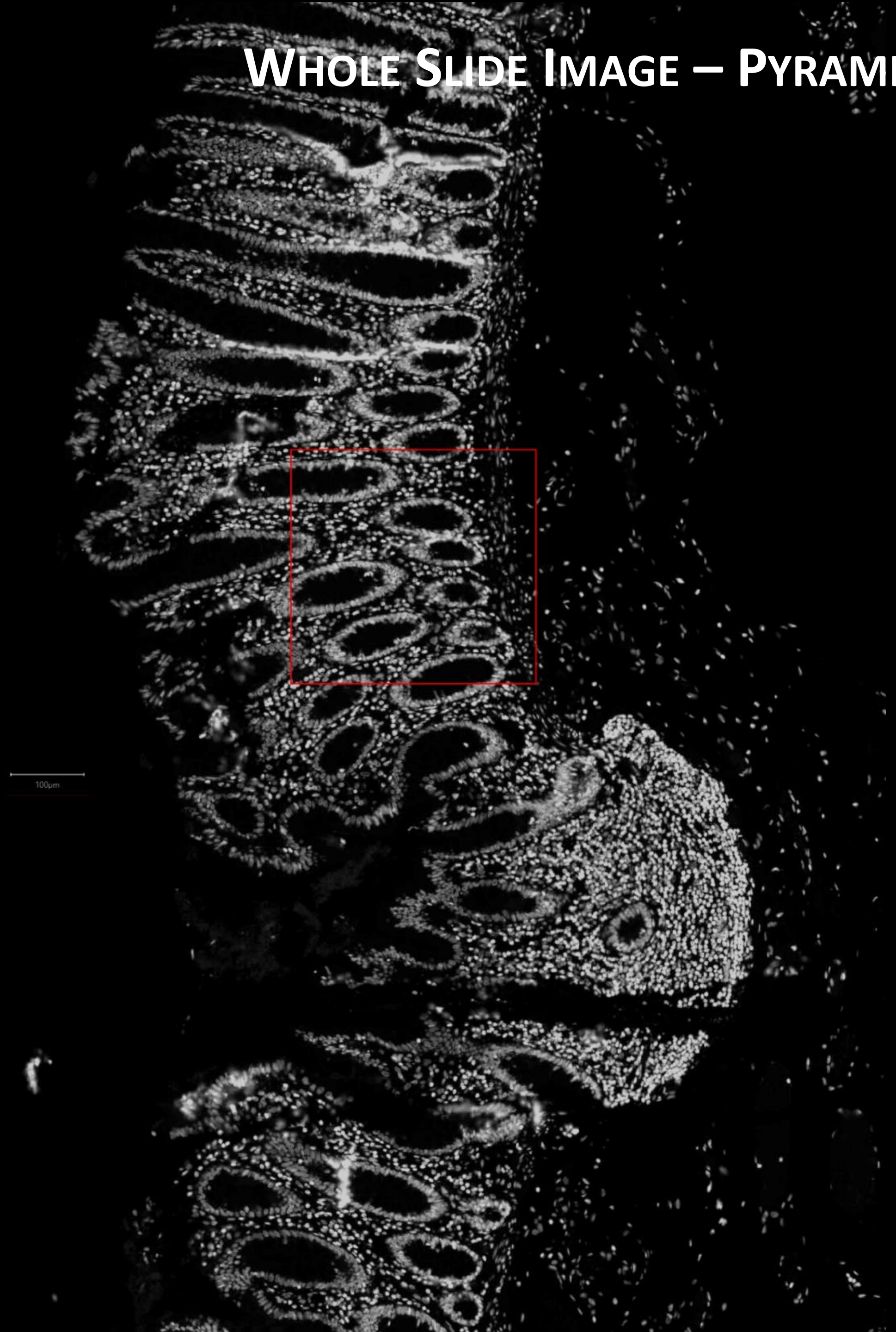


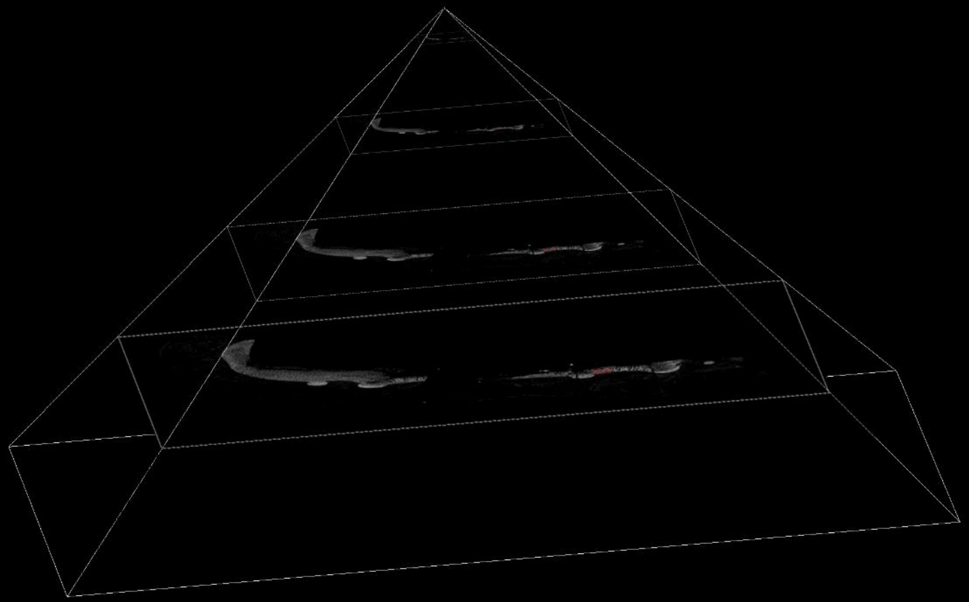
Whole Slide Image – Pyramidal Representation



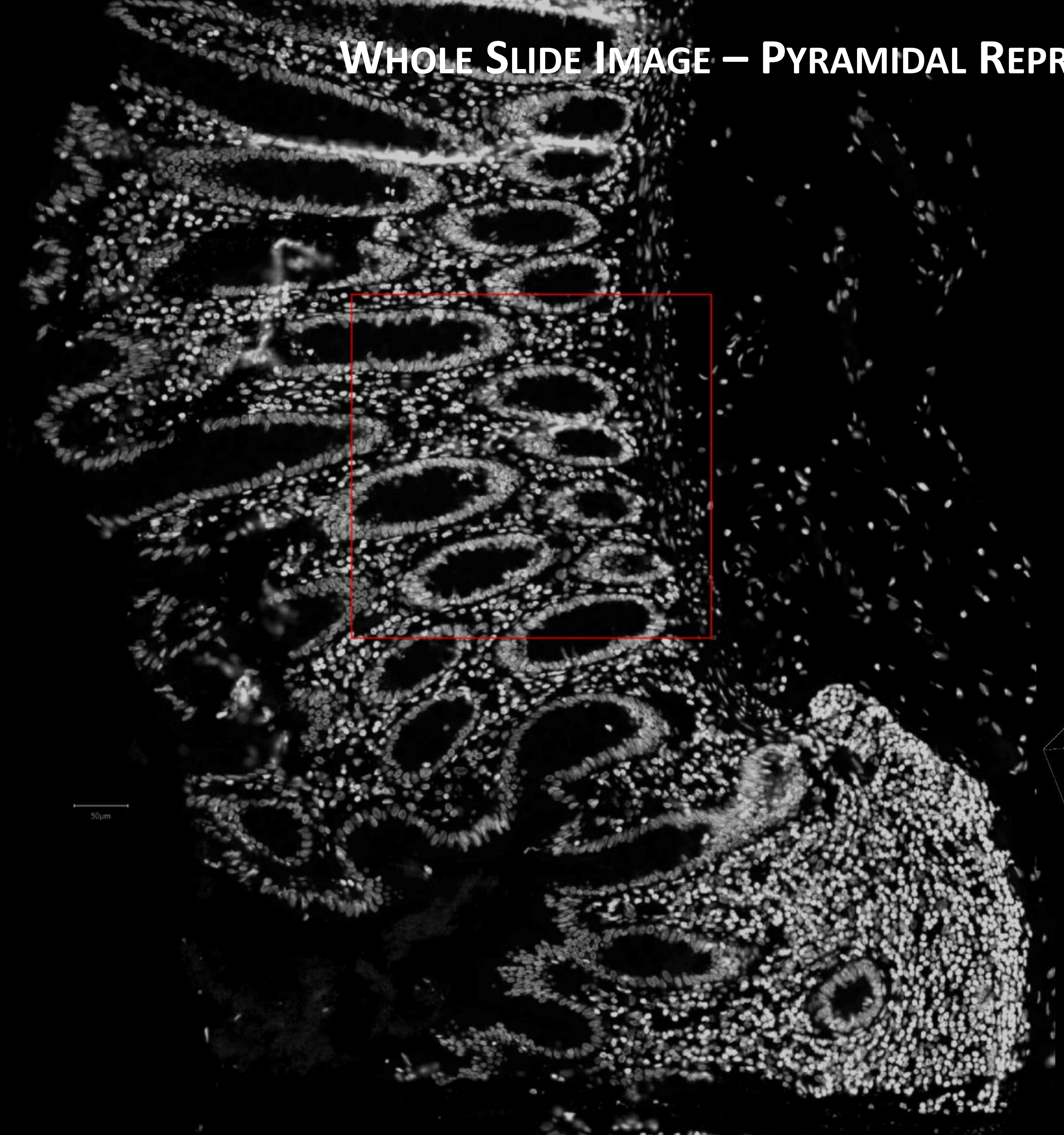


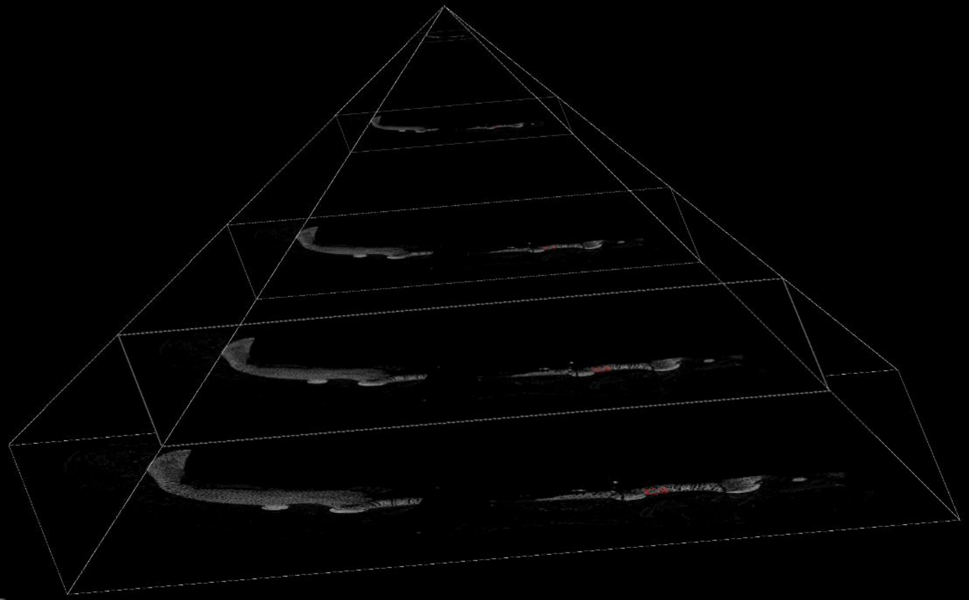
WHOLE SLIDE IMAGE – PYRAMIDAL REPRESENTATION



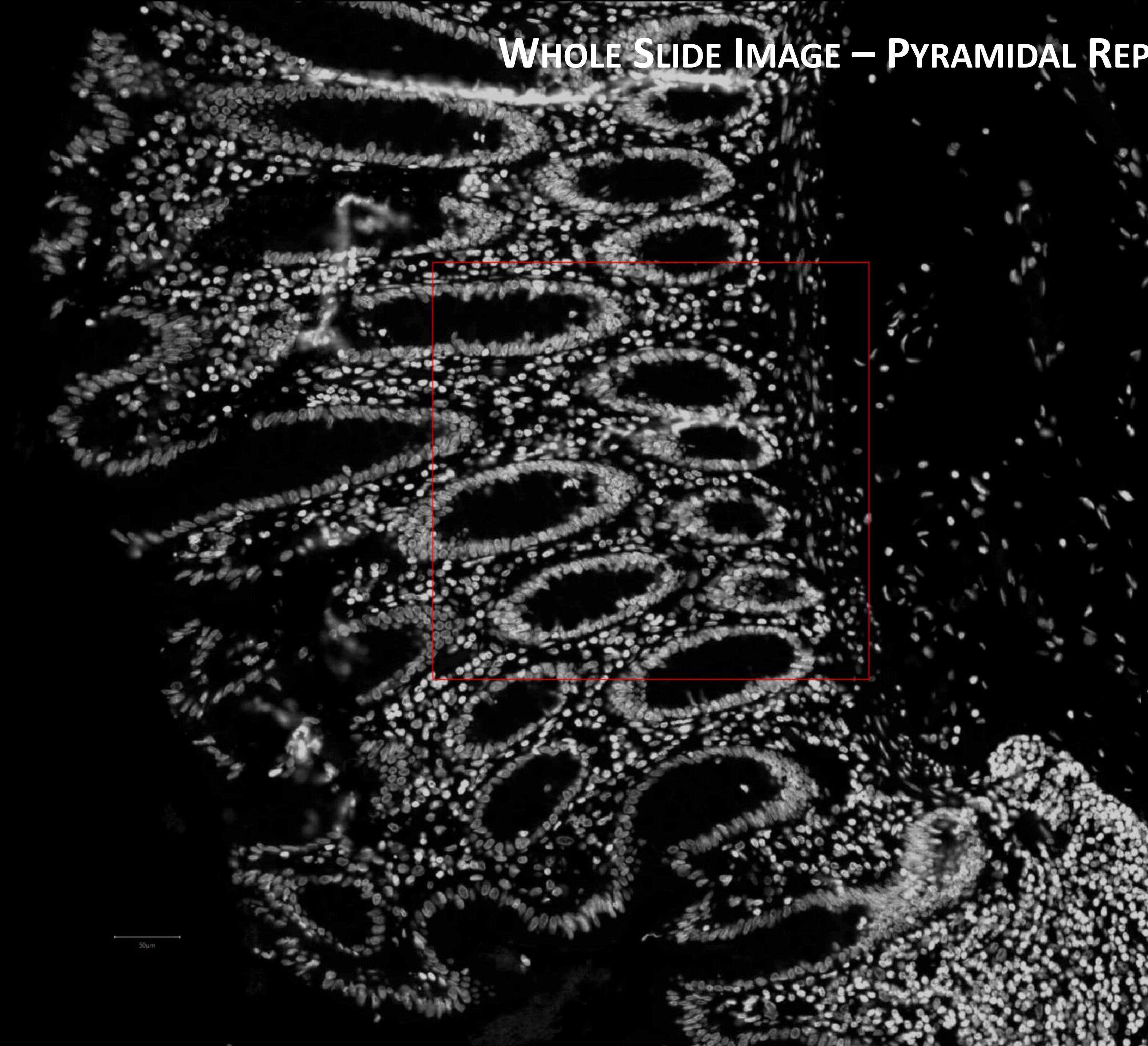


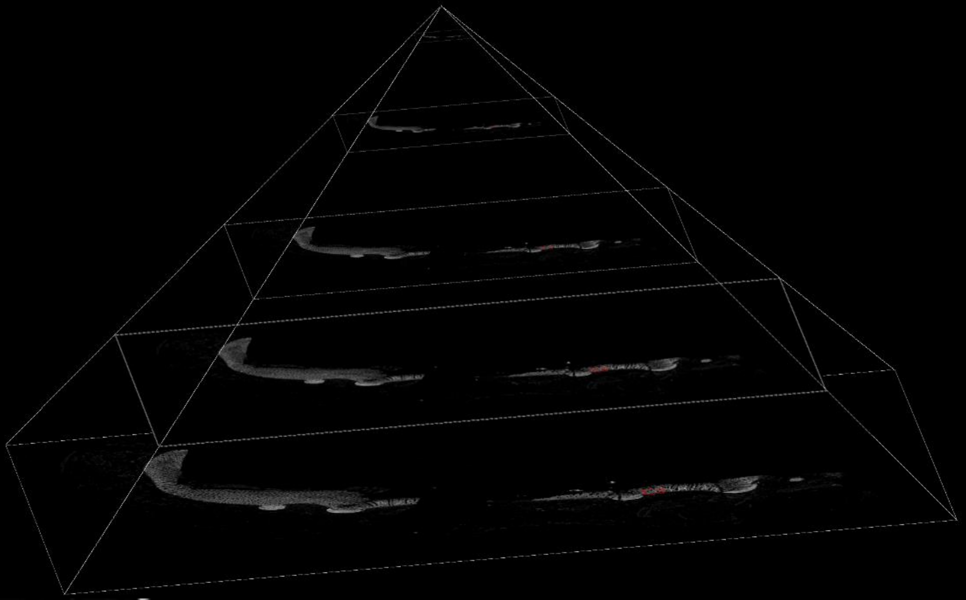
Whole Slide Image – Pyramidal Representation



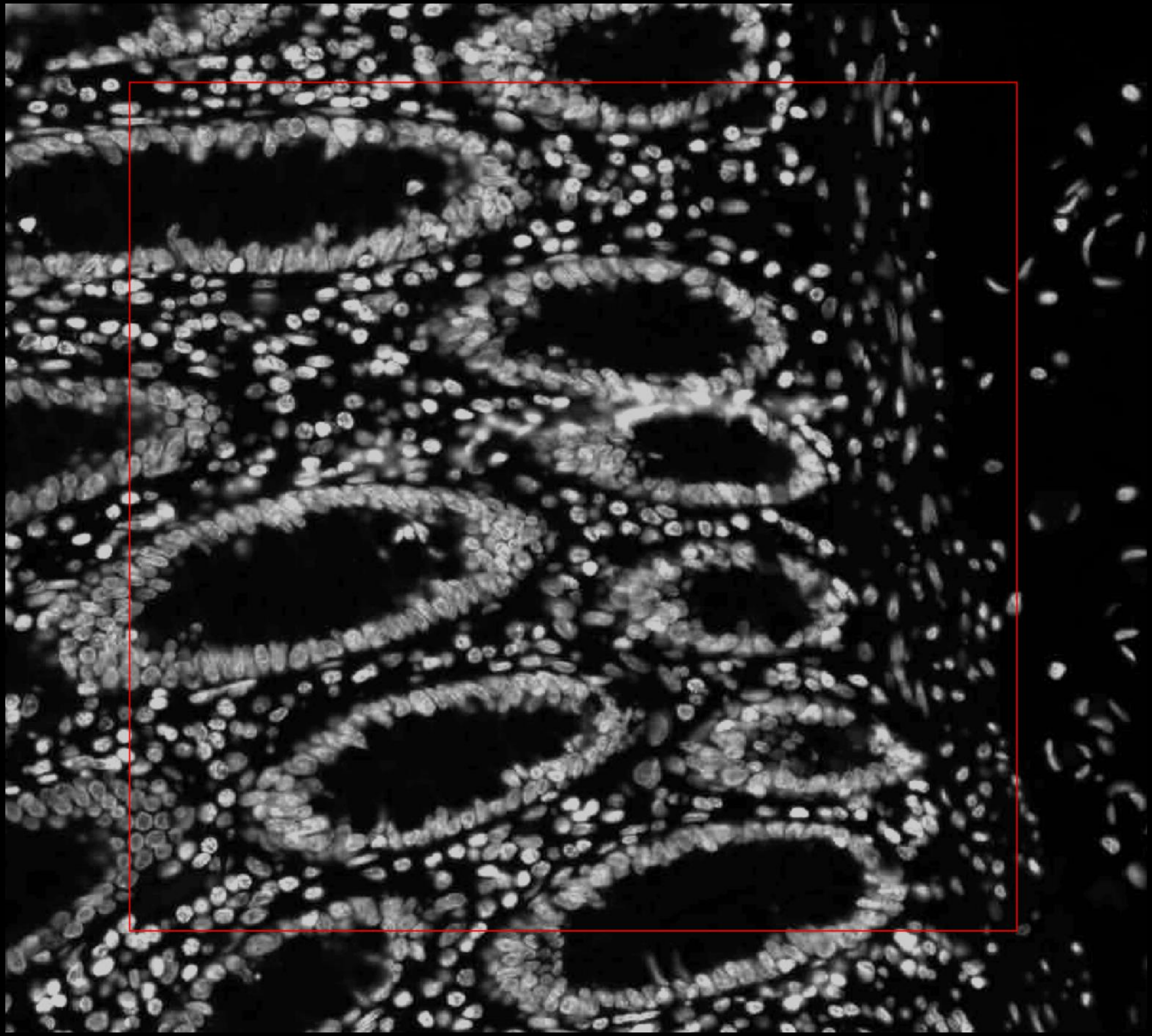


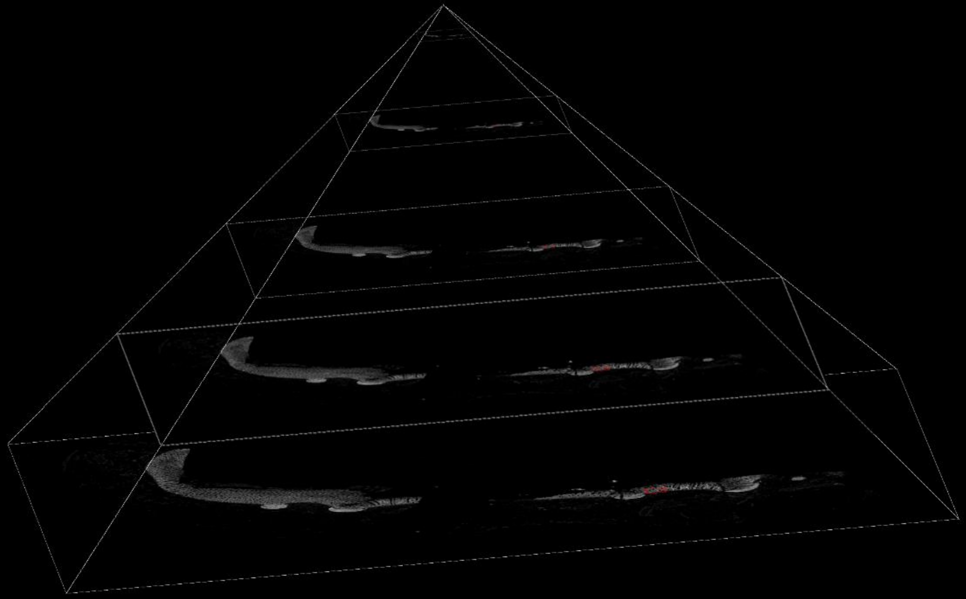
Whole Slide Image – Pyramidal Representation





WHOLE SLIDE IMAGE – PYRAMIDAL REPRESENTATION

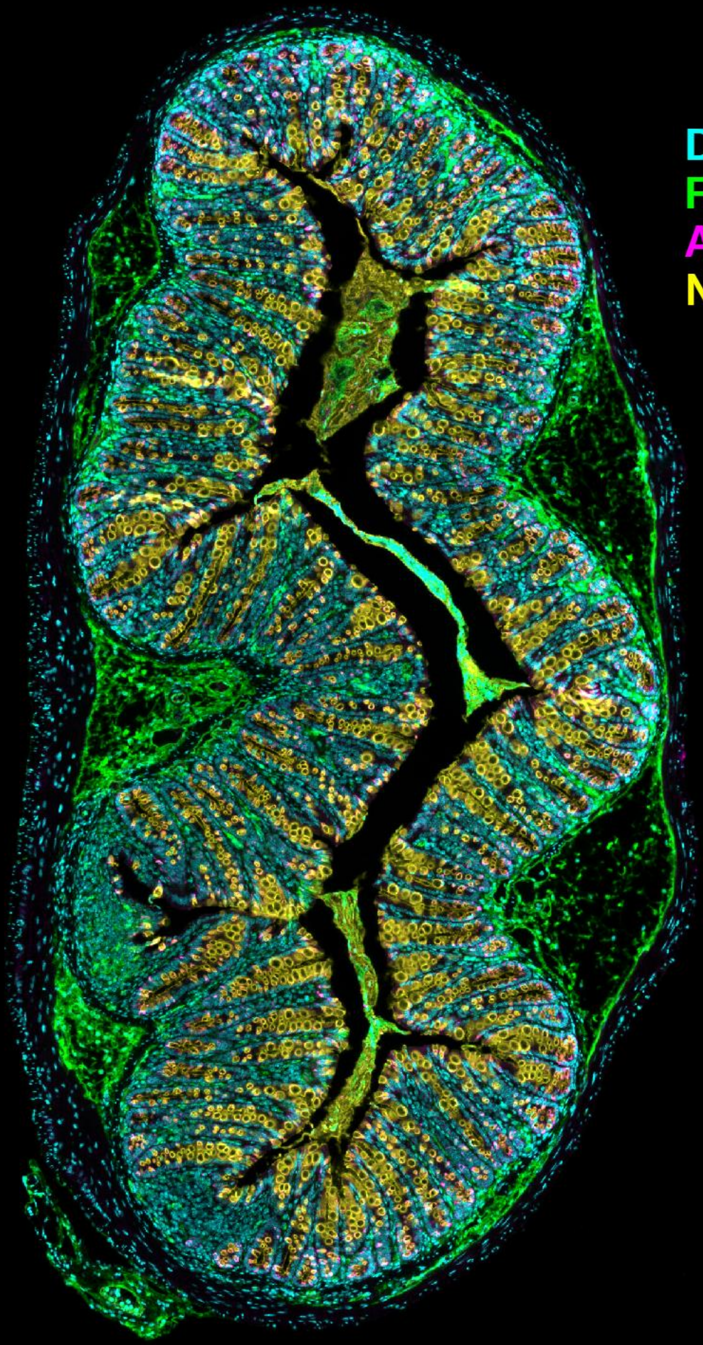




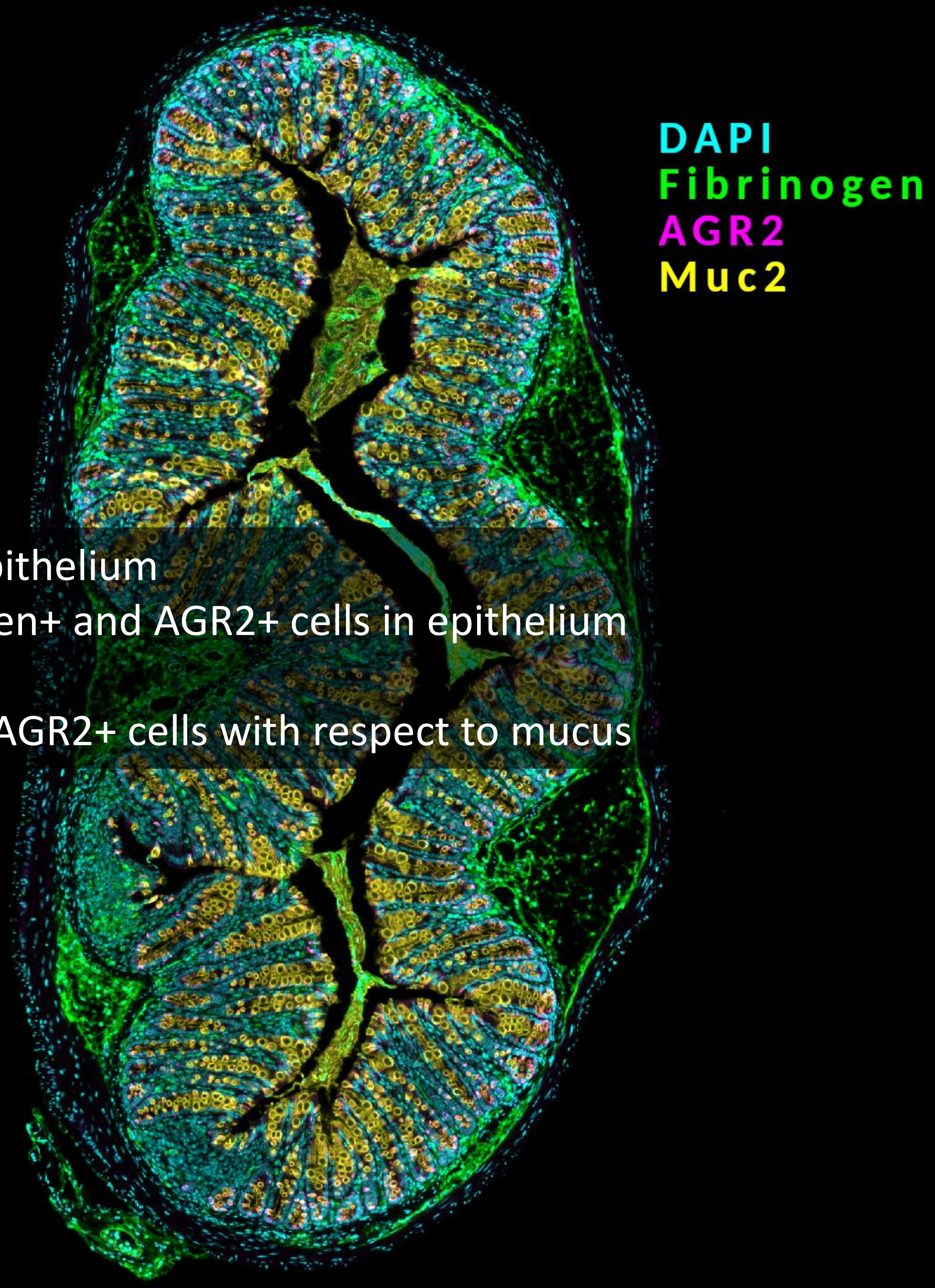
Whole-Slide Image Size

- A **full H&E(S)** whole-slide image would be:
  - 9 259 259 259 * 1 byte * 3 = **27.78 GB** for resolution 1
  - 9 259 259 259 * 1 byte * 3 = **6.94 GB** for resolution 2
  - 9 259 259 259 * 1 byte * 3 = **1.74 GB** for resolution 4
  - 9 259 259 259 * 1 byte * 3 = **434 MB** for resolution 8
  - 9 259 259 259 * 1 byte * 3 = **109 MB** for resolution 16
  - 9 259 259 259 * 1 byte * 3 = **27 MB** for resolution 32
  - 9 259 259 259 * 1 byte * 3 = **7 MB** for resolution 64
  - 9 259 259 259 * 1 byte * 3 = **1.7 MB** for resolution 128
  - 9 259 259 259 * 1 byte * 3 = **424 KB** for resolution 256

➡ Total of **37 GB uncompressed data**



Mouse colon, Marine Seffals, H2P2



- Muc2 tissue ratio in epithelium
- Proportion of Fibrinogen+ and AGR2+ cells in epithelium and in the stroma
- Spatial distribution of AGR2+ cells with respect to mucus

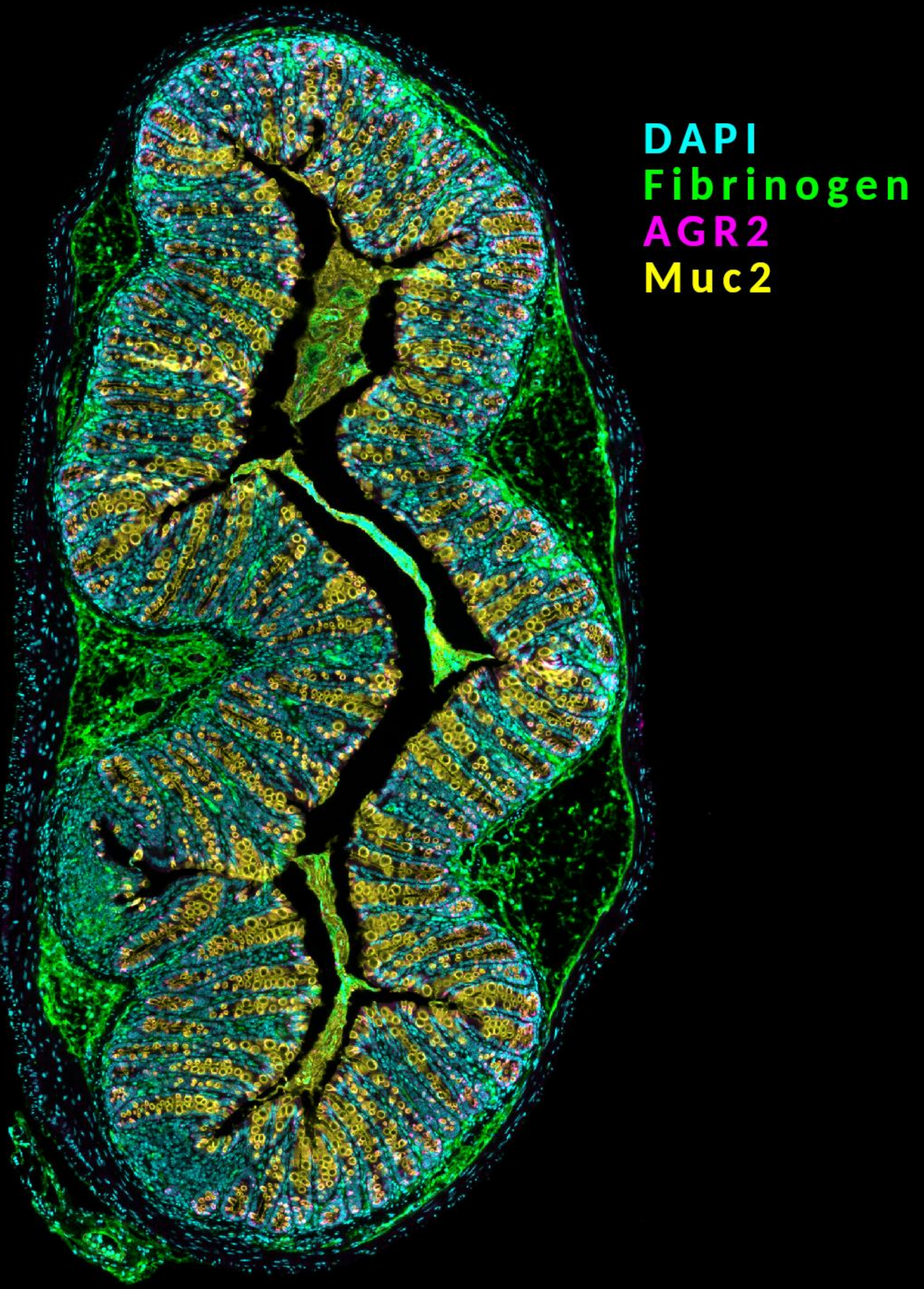
Mouse colon, Marine Seffals, H2P2

ANNOTATIONS

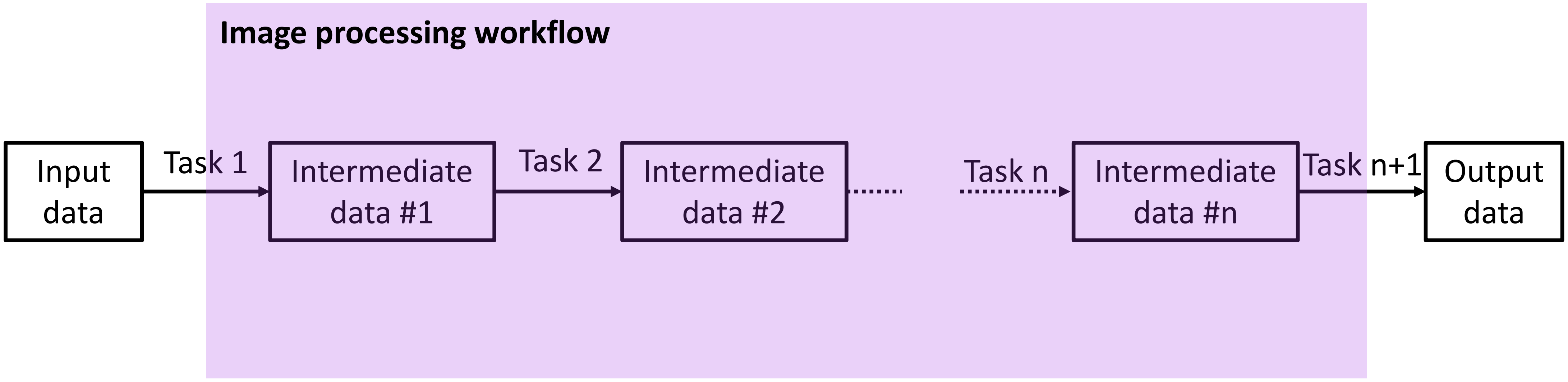
- Allow to **add information** to specific regions or entire images
- Lots of **features/measurements** can then be extracted from these regions
- **Powerful** and **storage-efficient** way to process images
- Can be **manually** defined or **automatically** estimated
- Can be enriched by **adding classes**

ANNOTATIONS

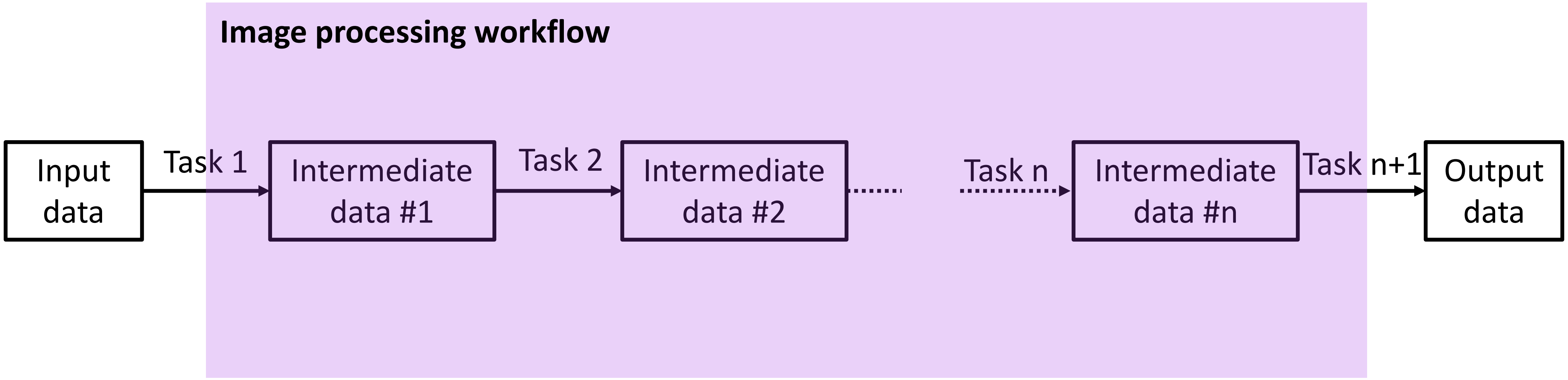
- **Open** Colon_1.ome.tif
- Create **different types** of annotations
- Play with **resolution**
- Look at the **measurements** for each **type of annotation**

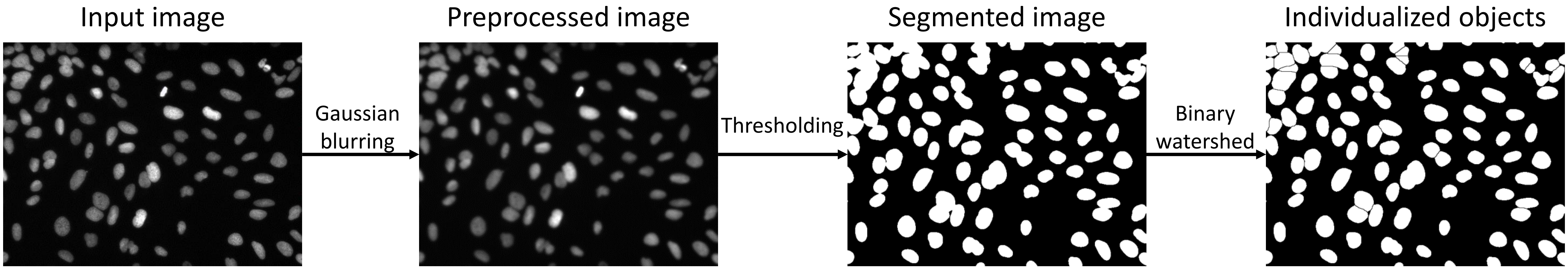


EXPLICIT PROGRAMMING

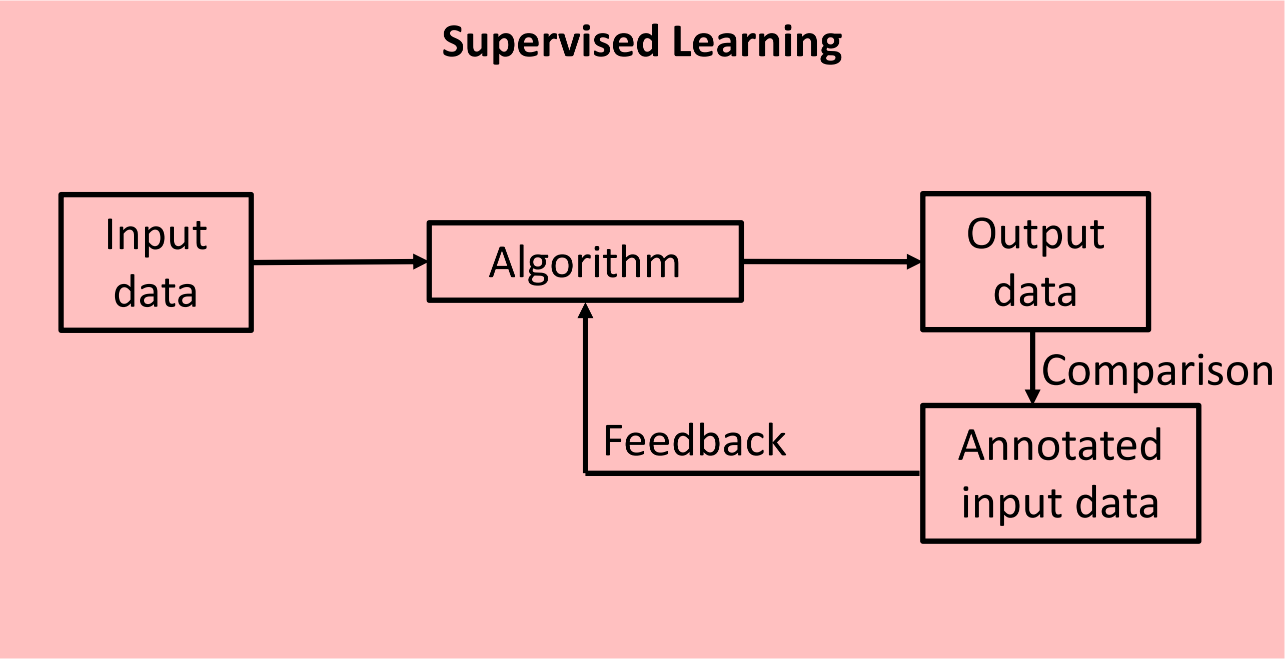


EXPLICIT PROGRAMMING

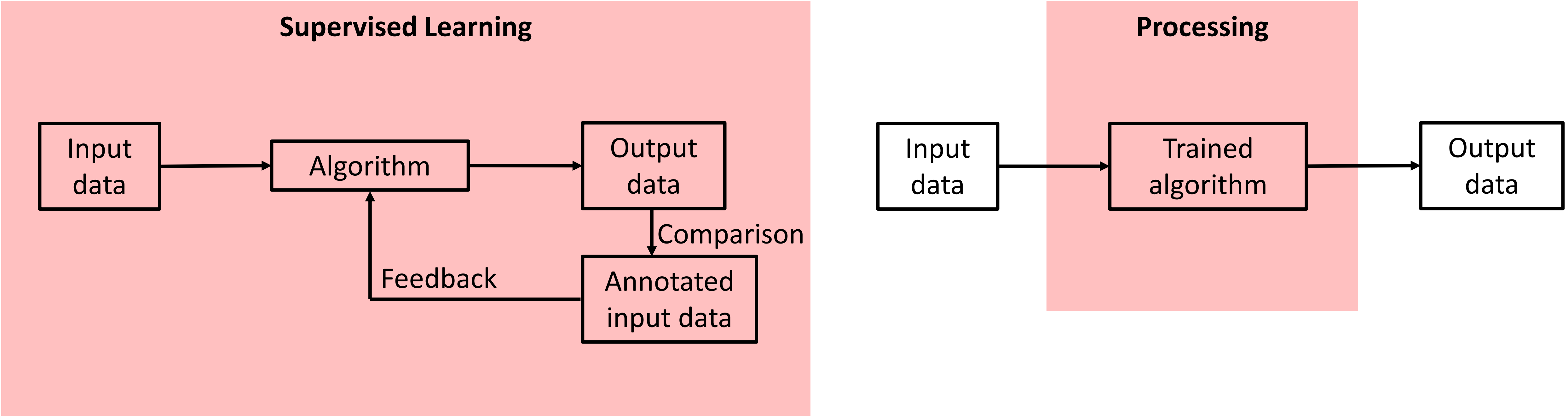




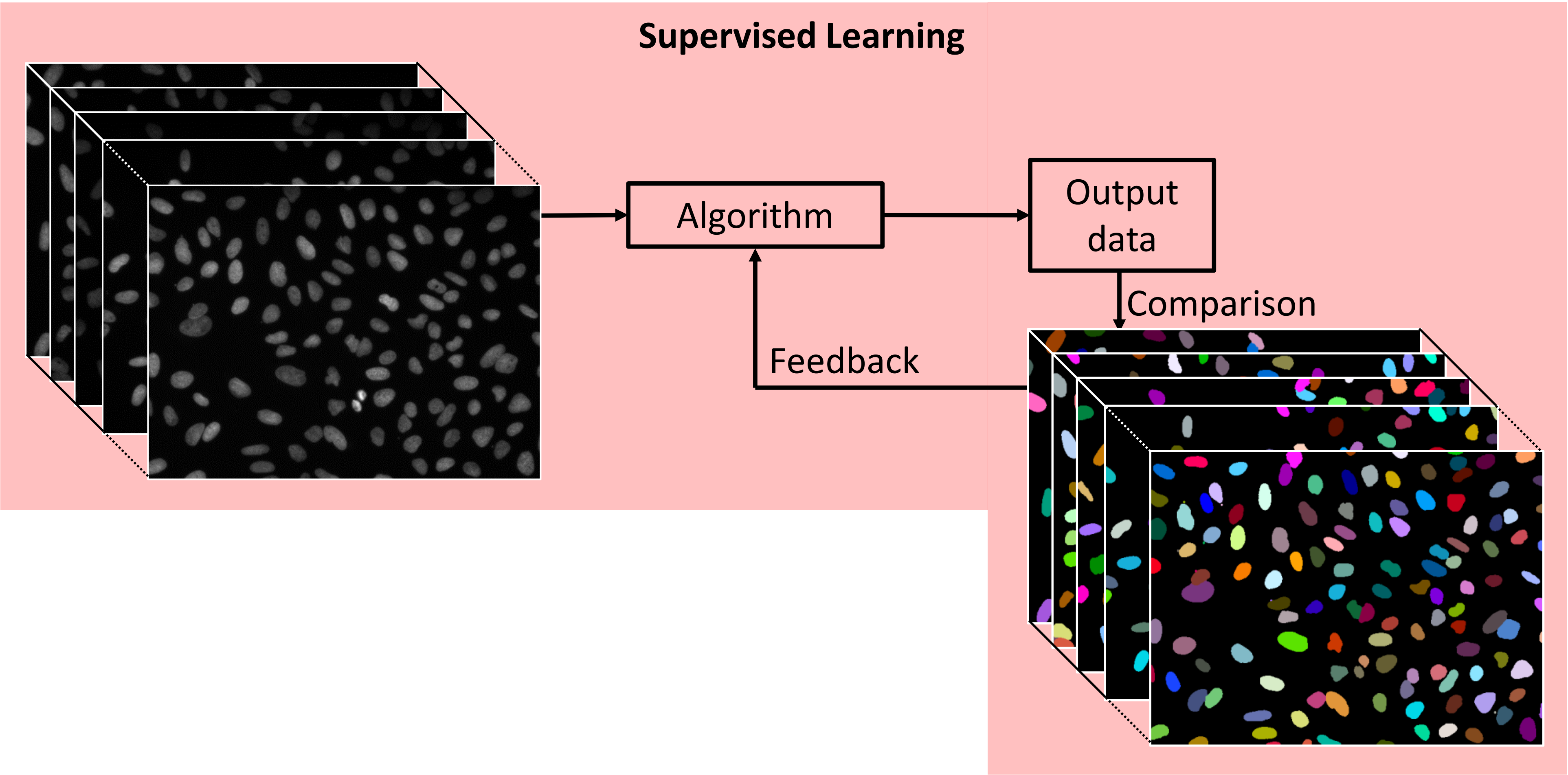
SUPERVISED MACHINE LEARNING



SUPERVISED MACHINE LEARNING



SUPERVISED MACHINE LEARNING



Supervised Learning

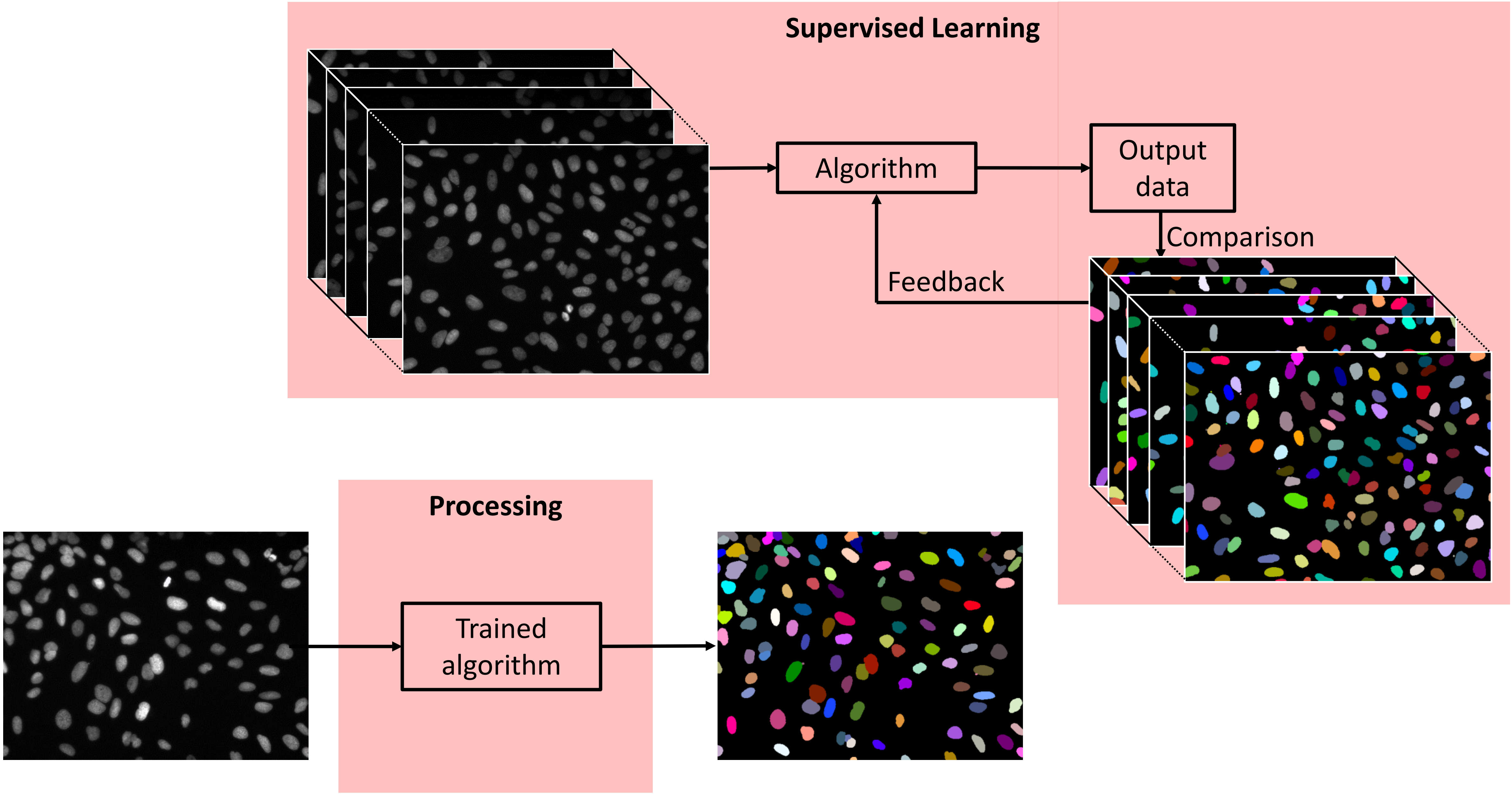
Algorithm

Output data

Comparison

Feedback

Supervised Machine Learning

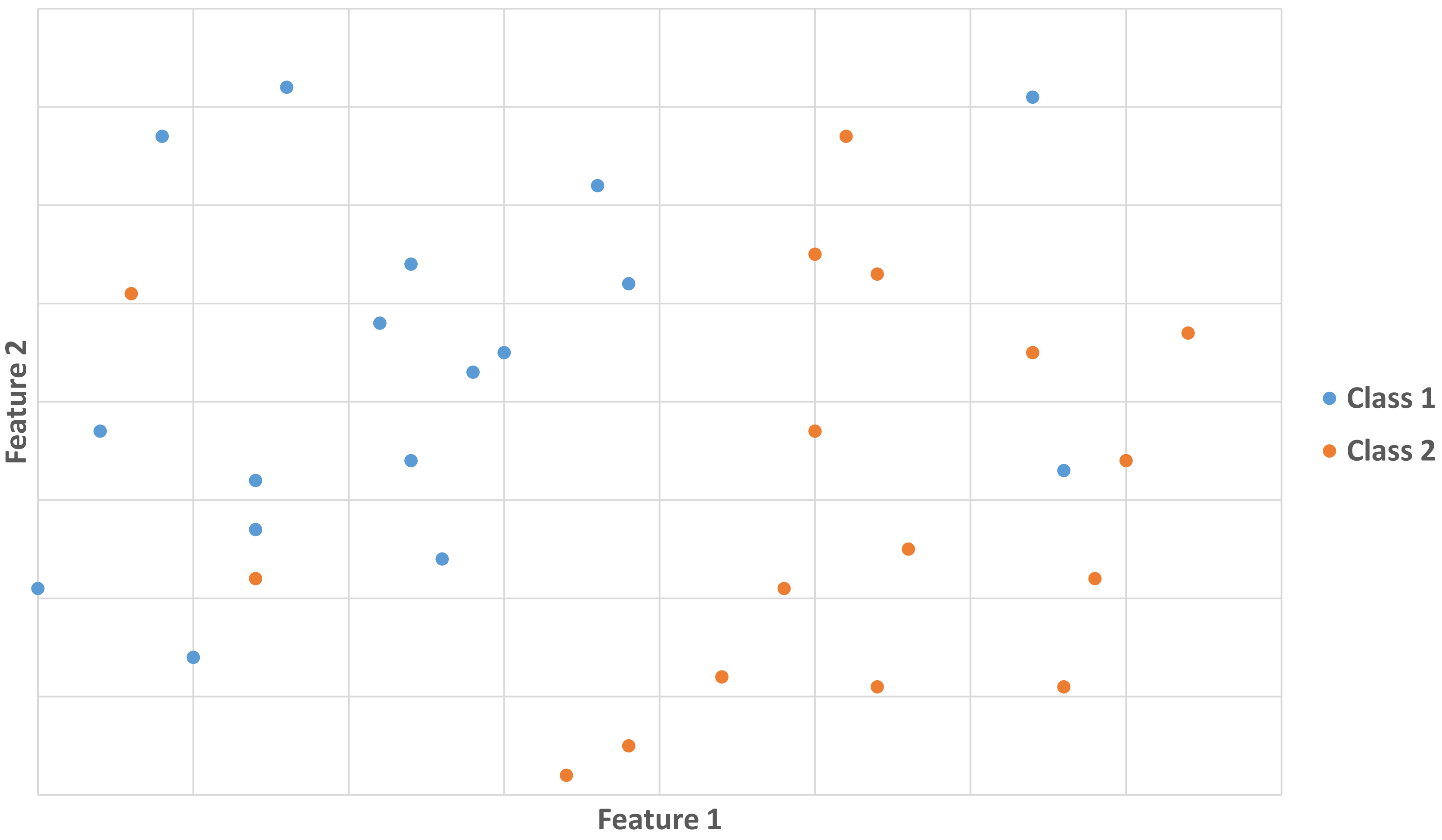


Shallow Machine Learning for Pixel Classification

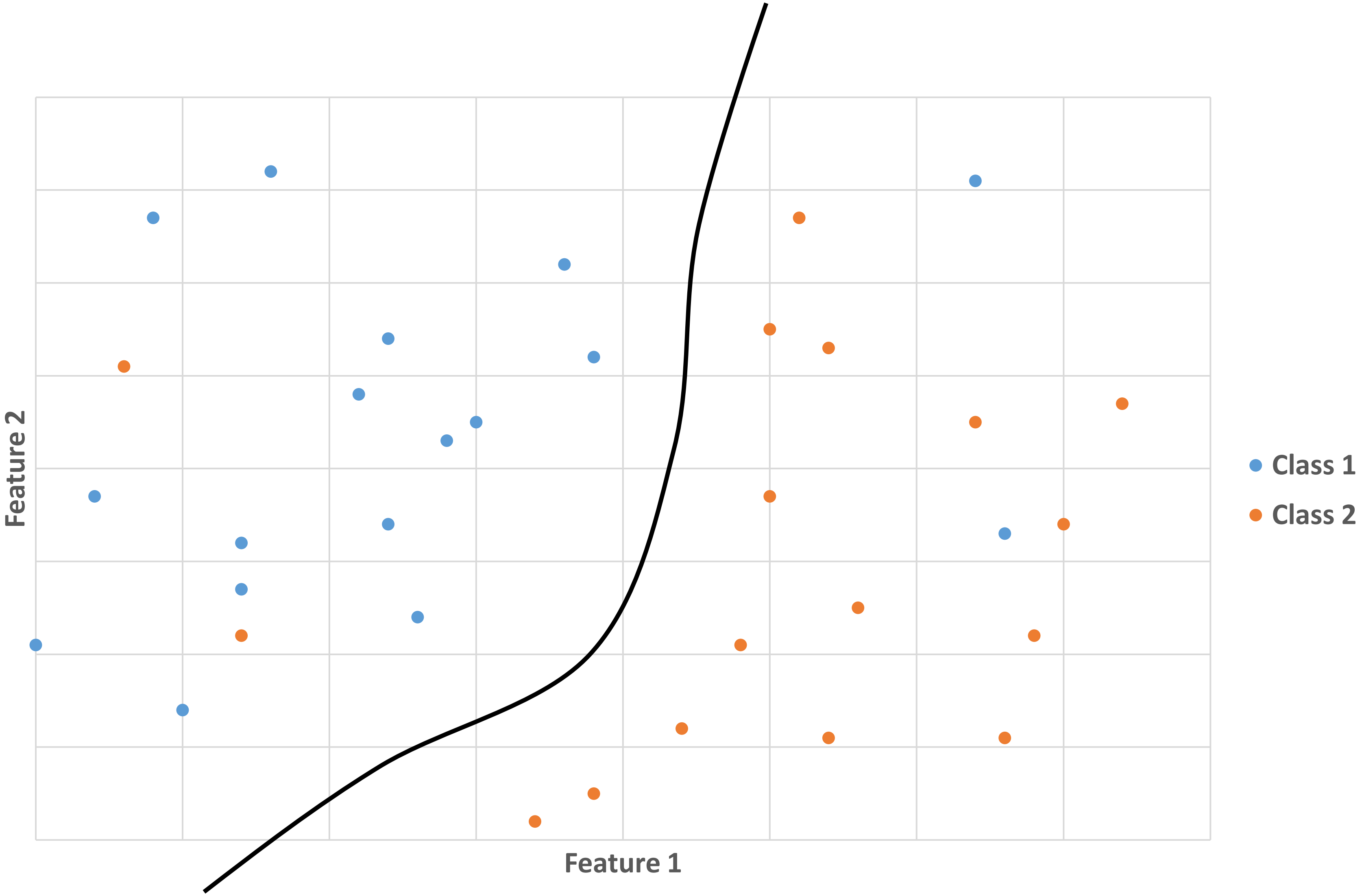
Supervised classification:

- **Examples** of classes are **manually** defined by the user
- A **classifier** is **trained** by using **defined features** with these examples
- Data is then **automatically classified** by using the trained classifier

Shallow Machine Learning for Pixel Classification

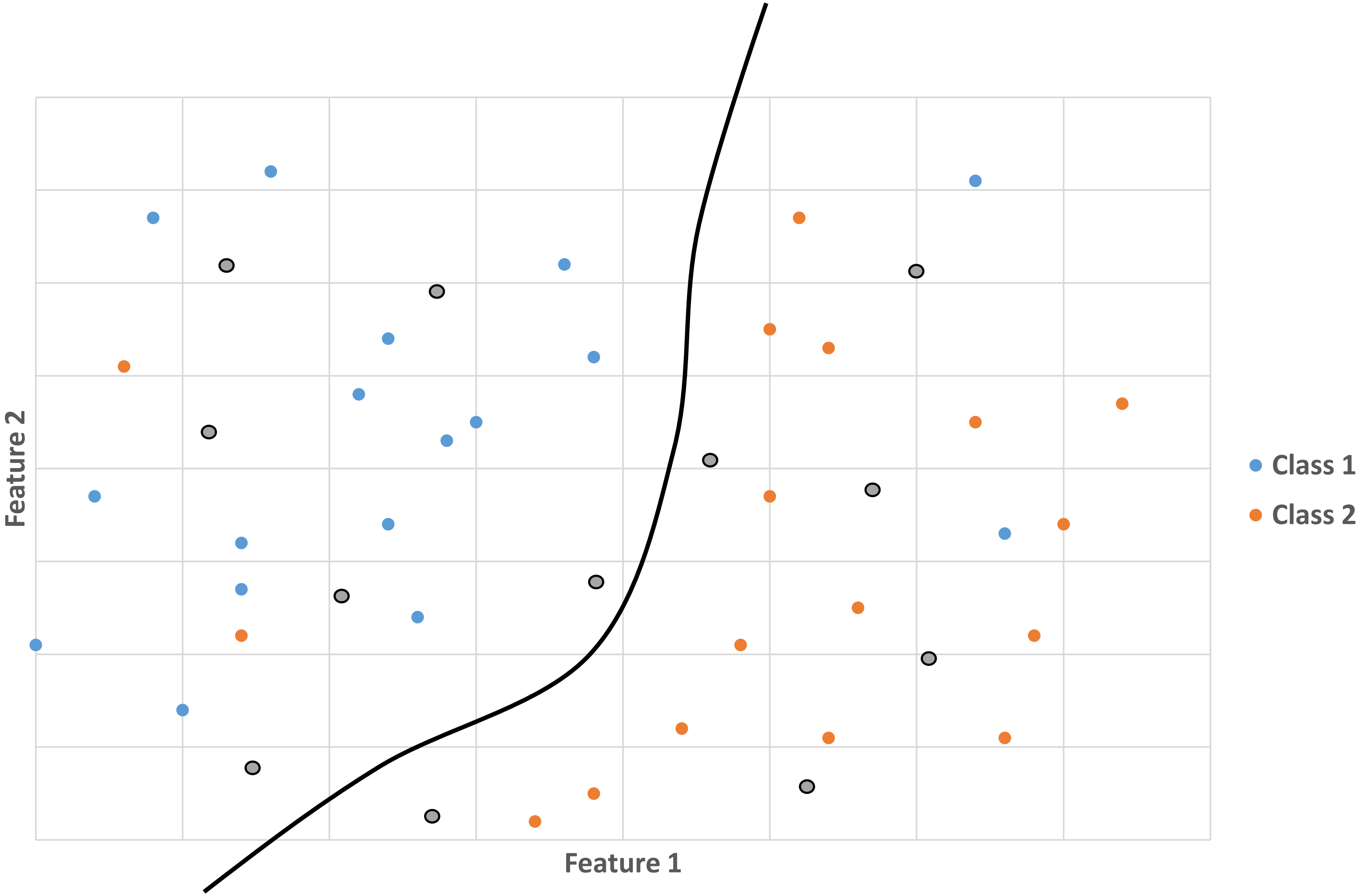


Shallow Machine Learning for Pixel Classification



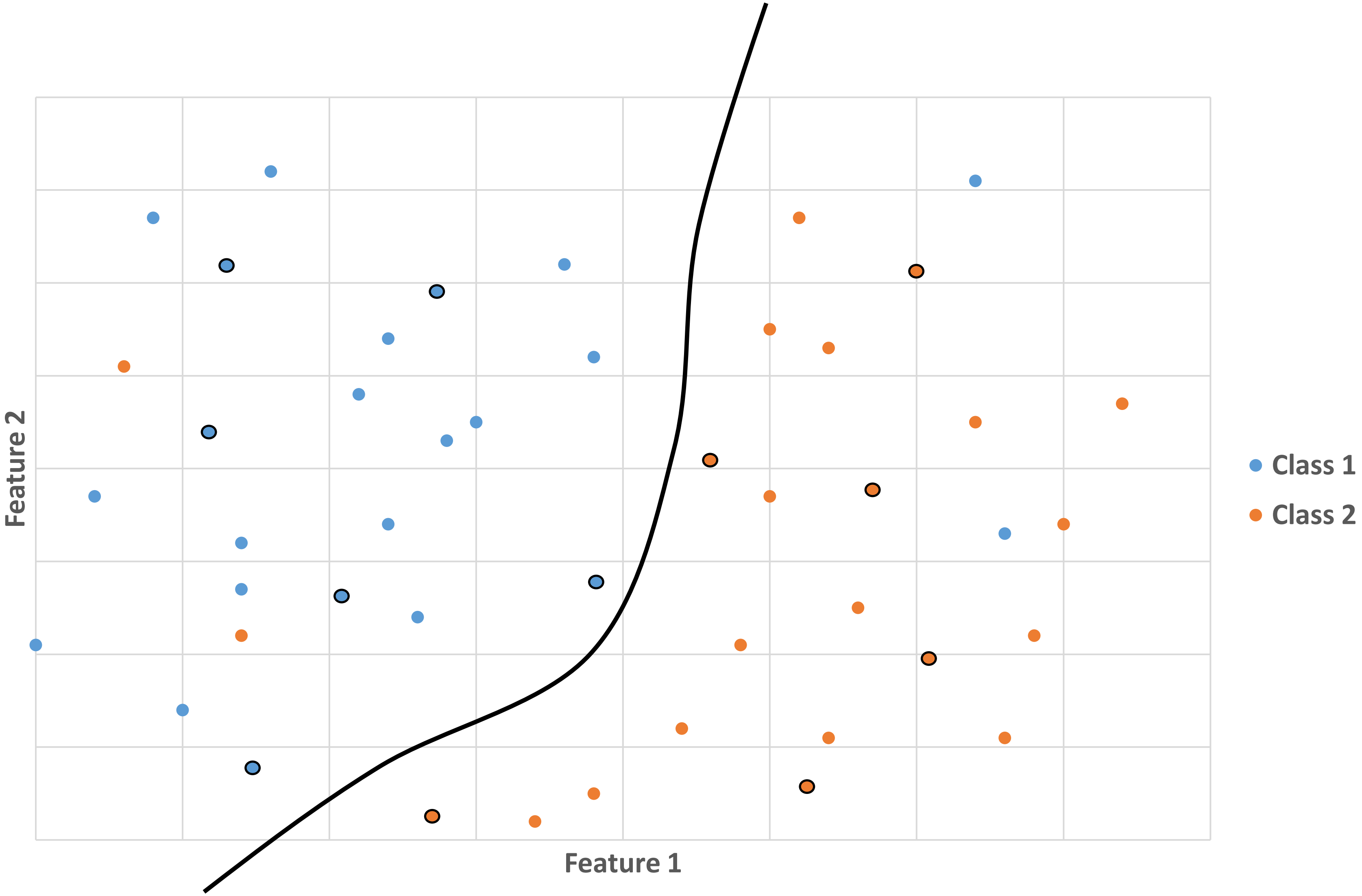
Training a classifier consists in **estimating** the **"best" separation** between classes

Shallow Machine Learning for Pixel Classification



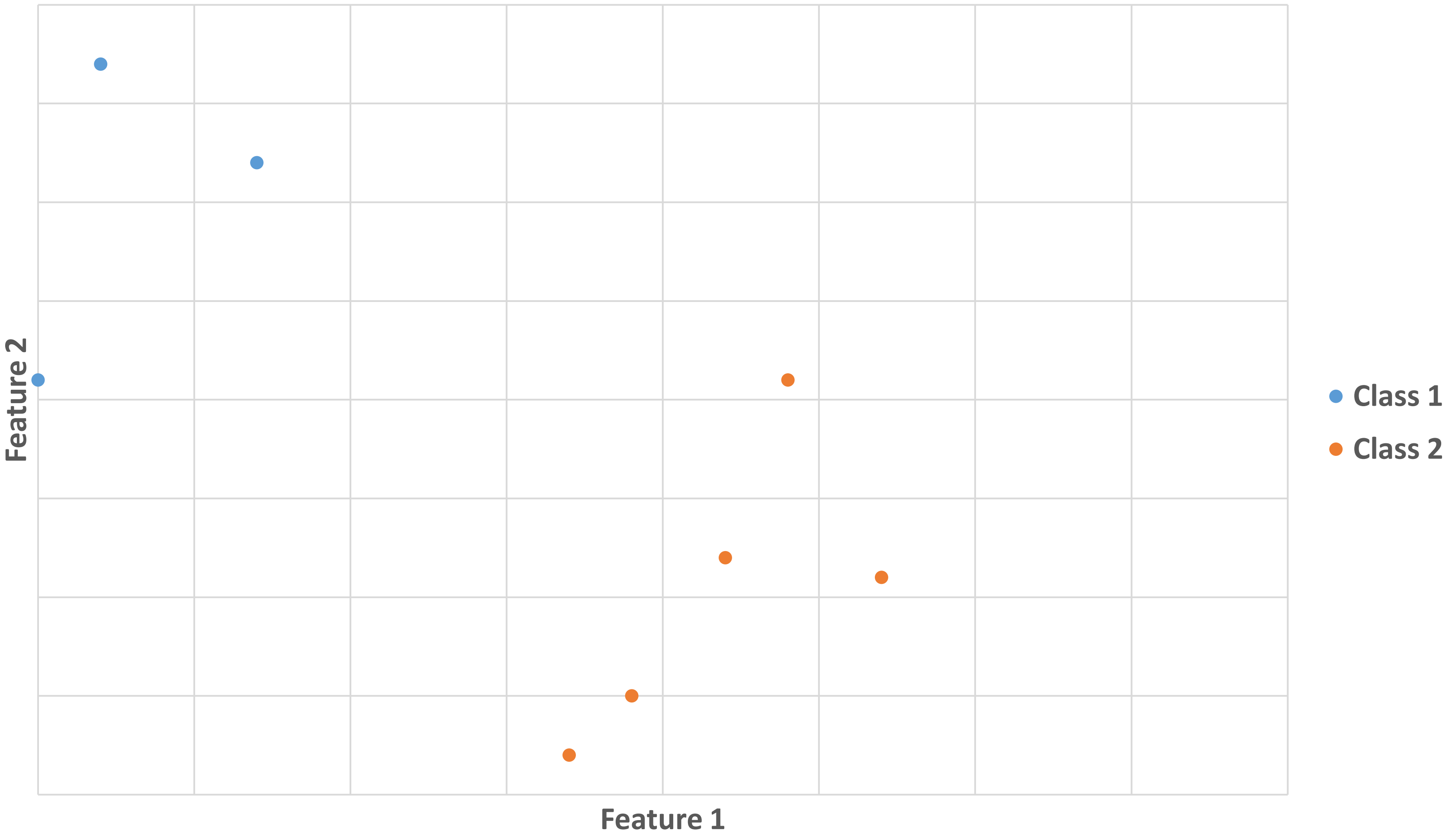
The **estimated class** for new data will be given by the **trained classifier**

SHALLOW MACHINE LEARNING FOR PIXEL CLASSIFICATION



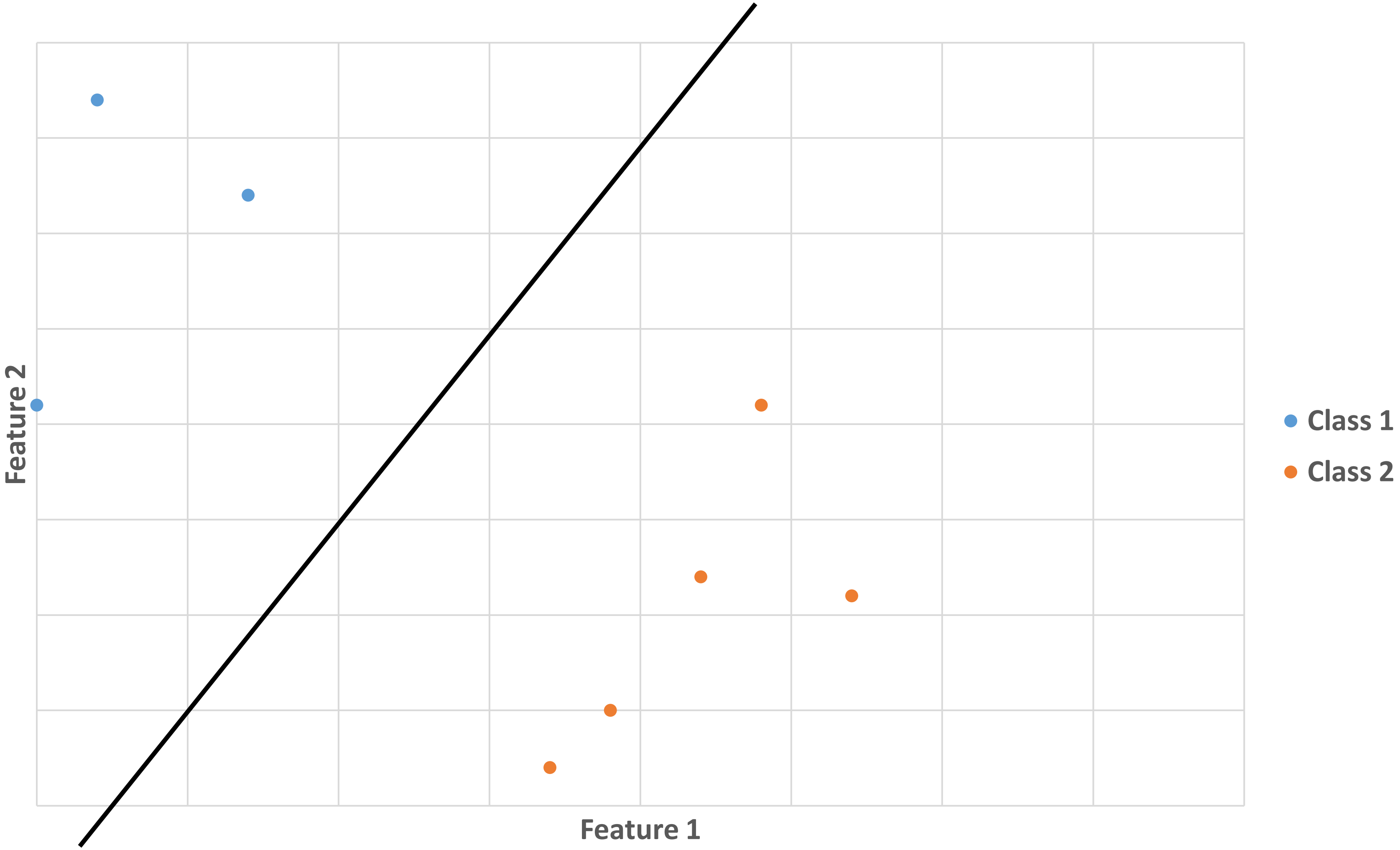
The **estimated class** for new data will be given by the **trained classifier**

Shallow Machine Learning for Pixel Classification



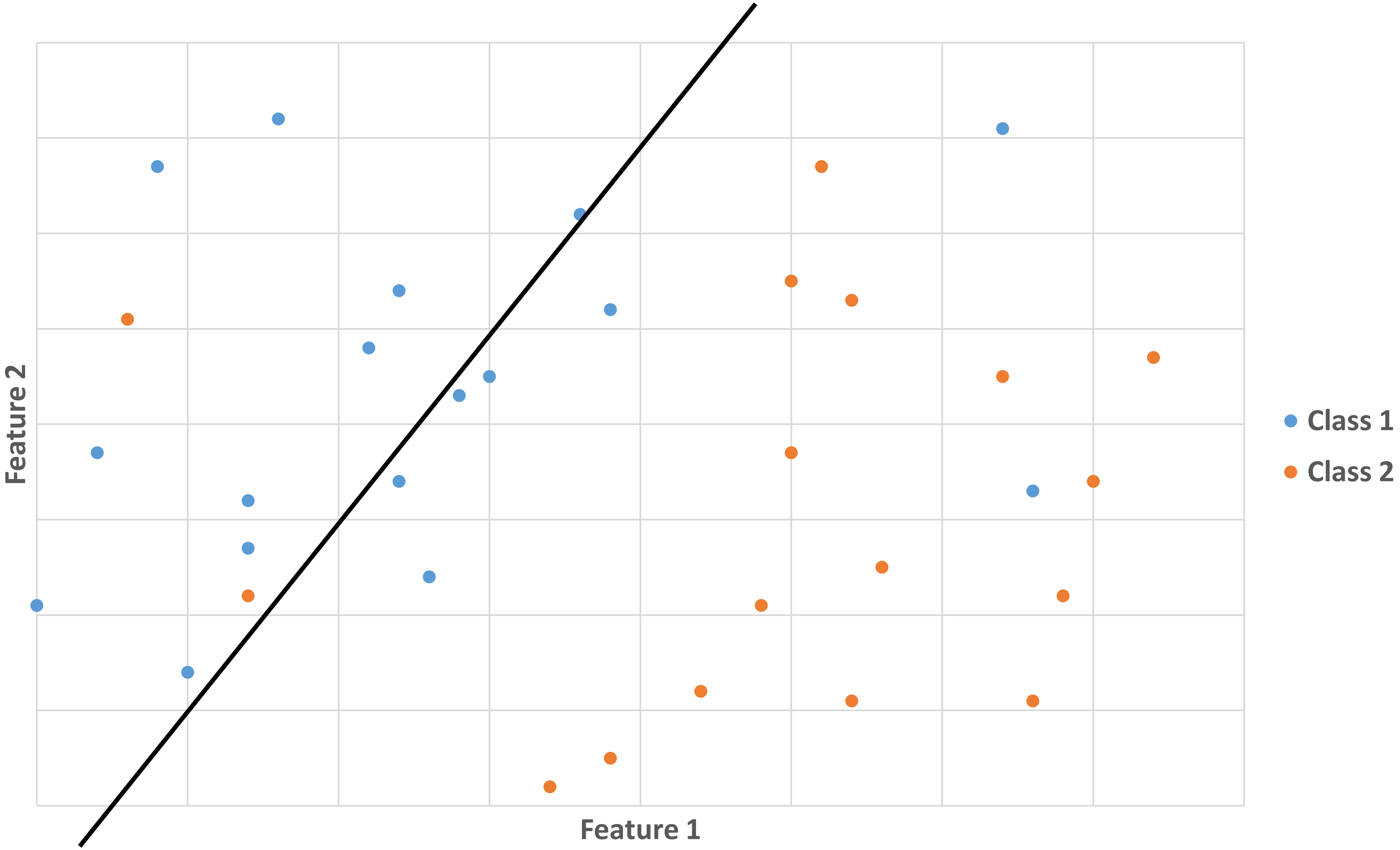
Training will be easier if **enough** and **representative** data is used

Shallow Machine Learning for Pixel Classification



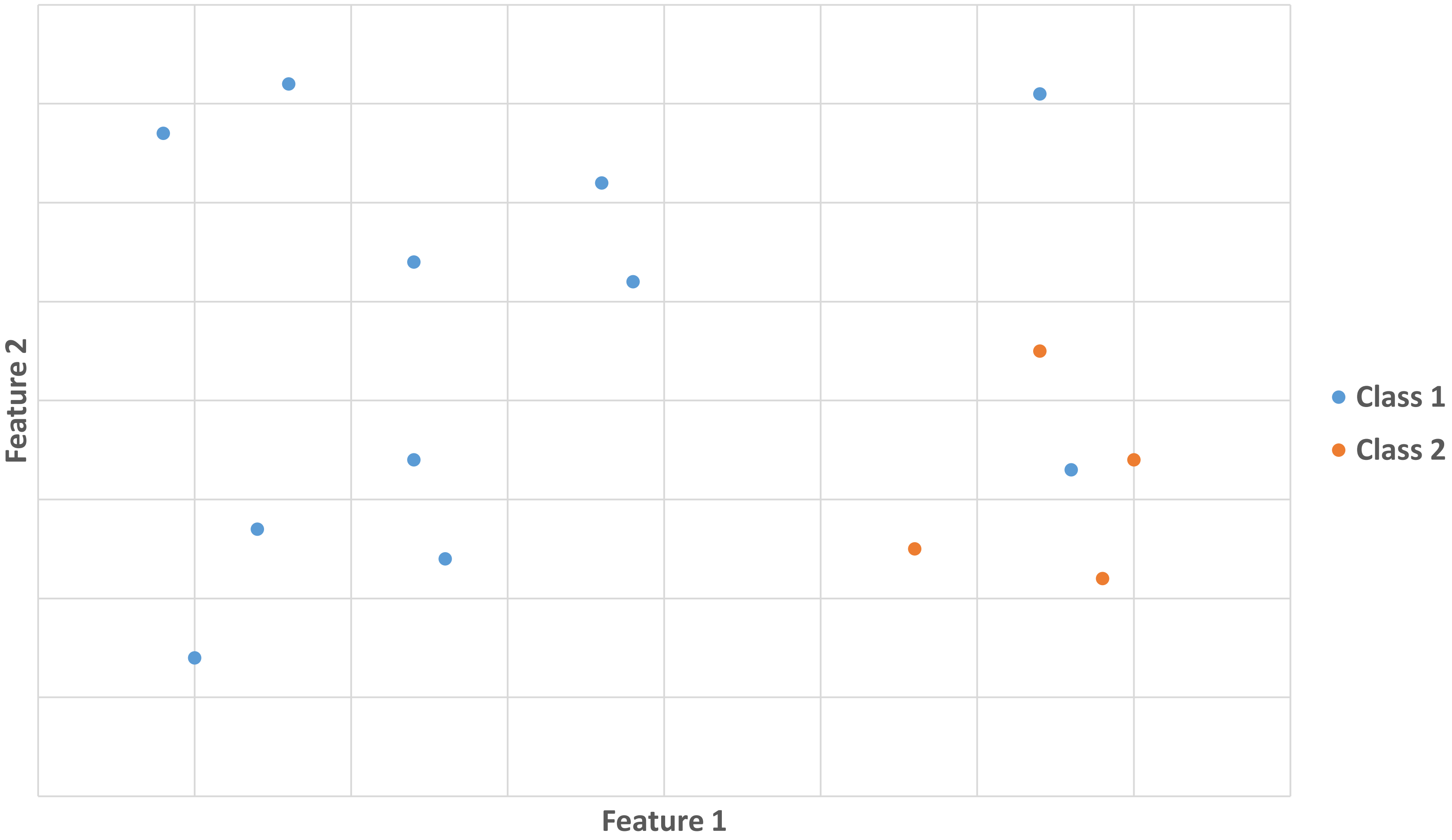
Training will be easier if **enough** and **representative** data is used

Shallow Machine Learning for Pixel Classification



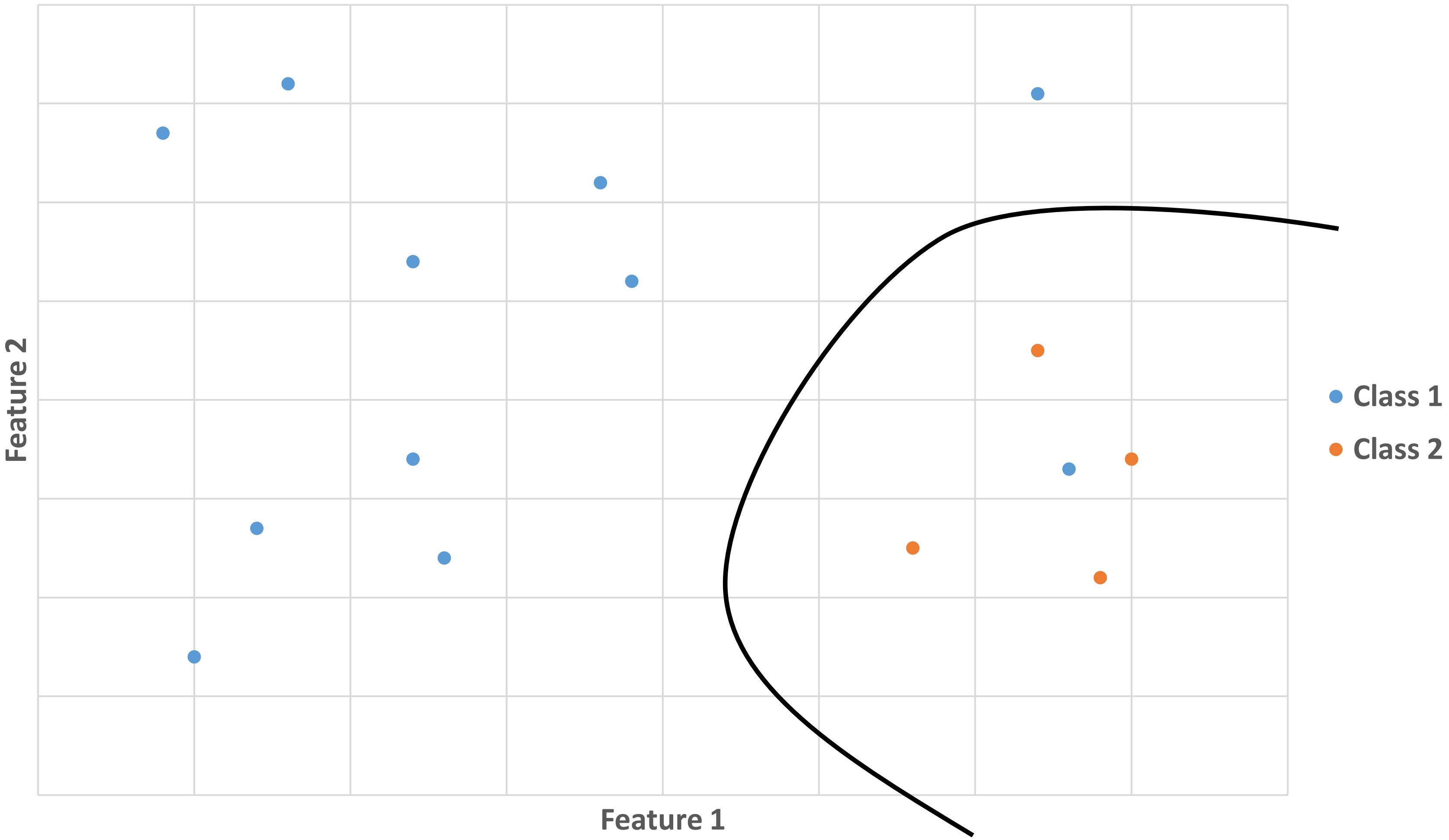
Training will be easier if **enough** and **representative** data is used

Shallow Machine Learning for Pixel Classification



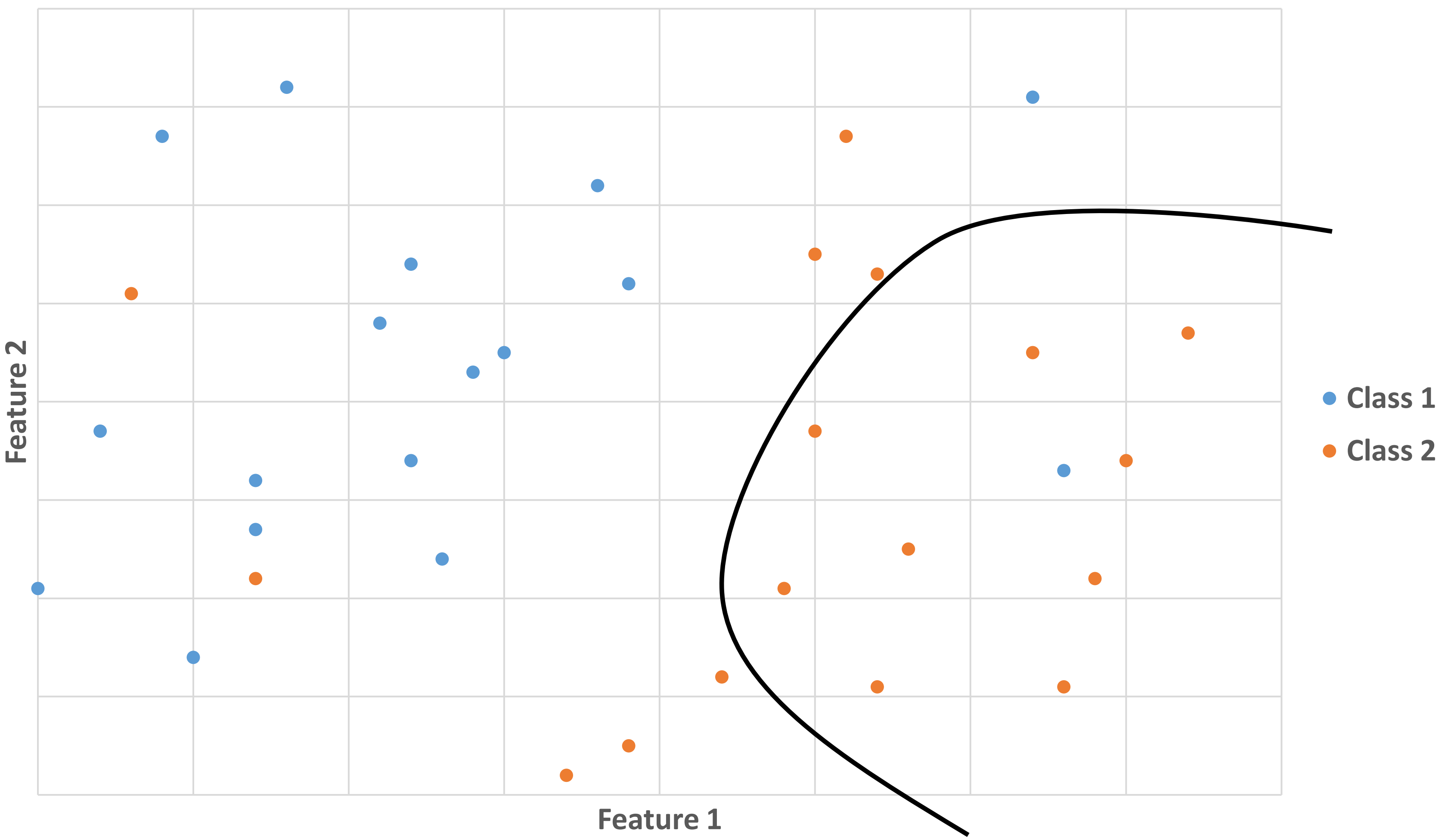
Class imbalance makes the training **more difficult**

Shallow Machine Learning for Pixel Classification



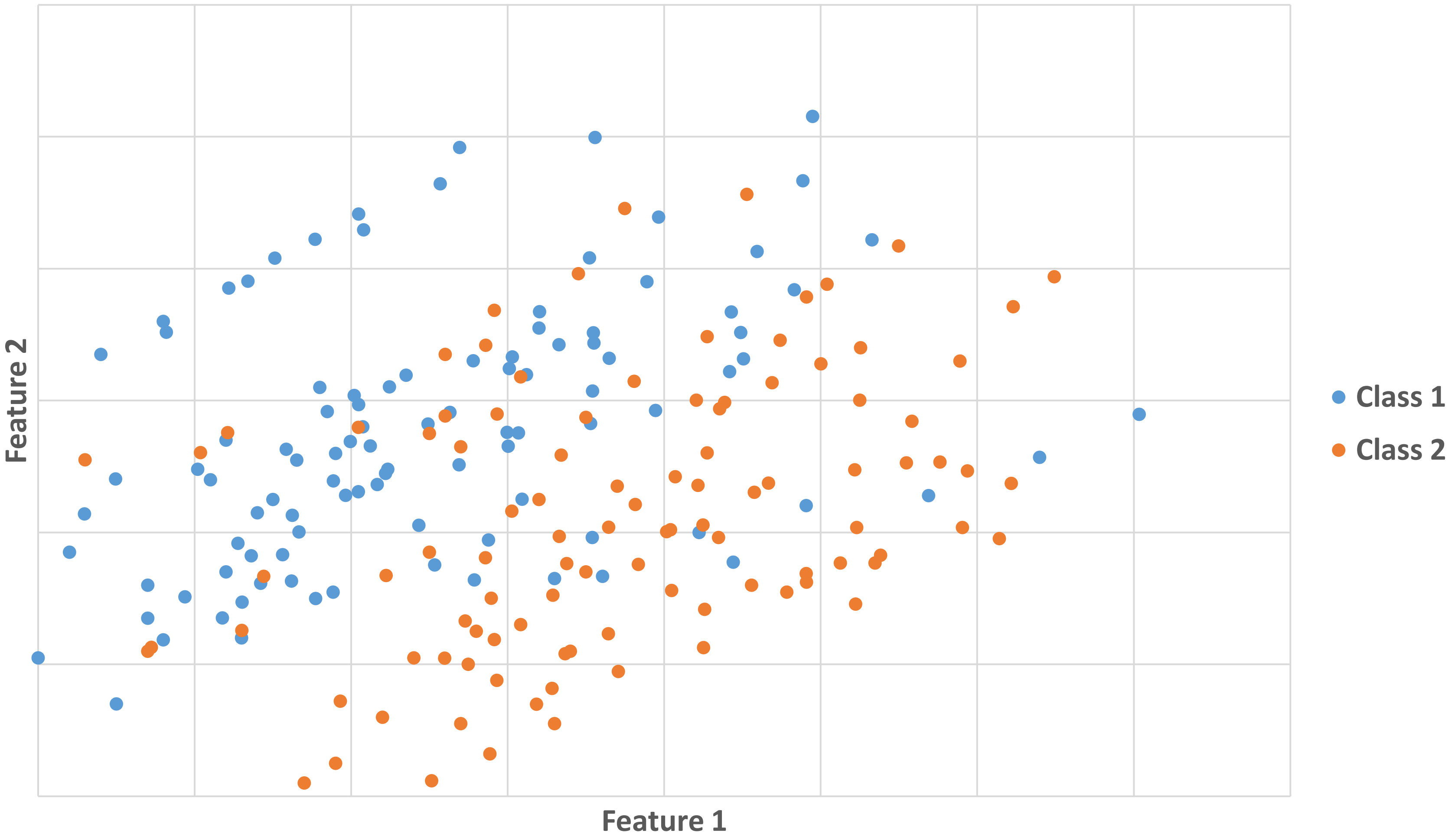
Class imbalance makes the training **more difficult**

Shallow Machine Learning for Pixel Classification



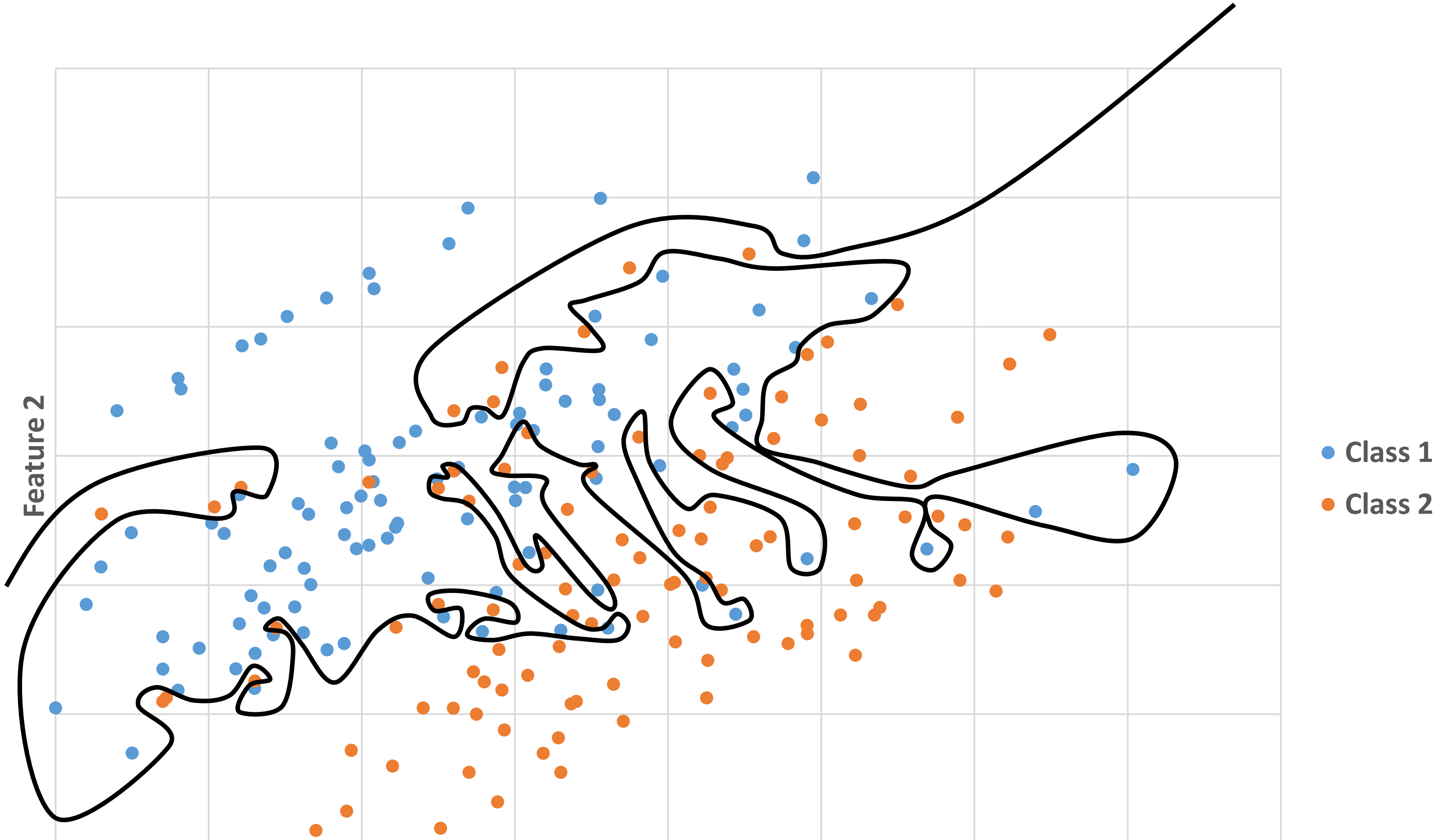
Class imbalance makes the training **more difficult**

Shallow Machine Learning for Pixel Classification



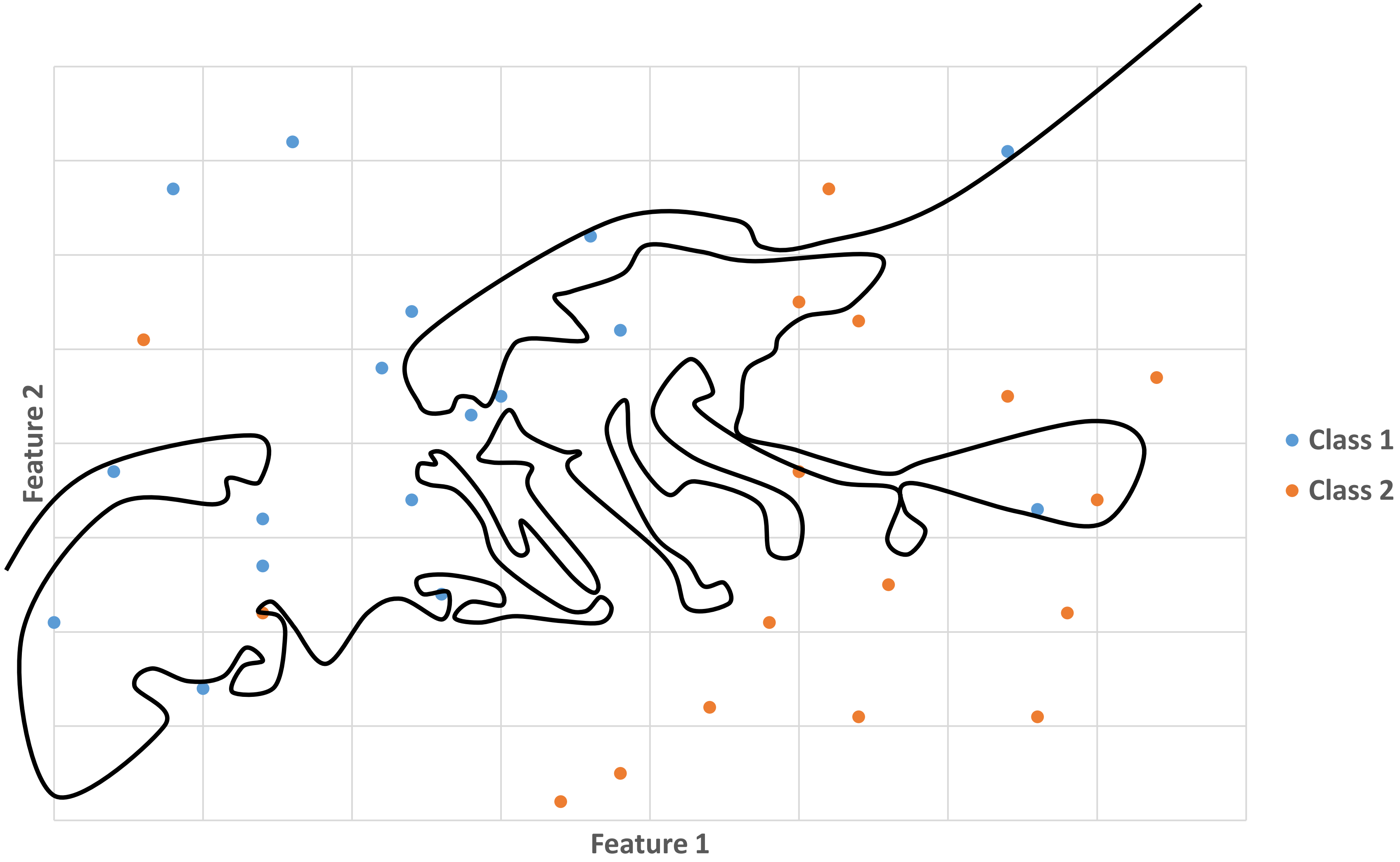
Too many annotations can lead to **over-fitting**: works great on training data but poorly on new data

Shallow Machine Learning for Pixel Classification



Too many annotations can lead to **over-fitting**: works great on training data but poorly on new data

Shallow Machine Learning for Pixel Classification



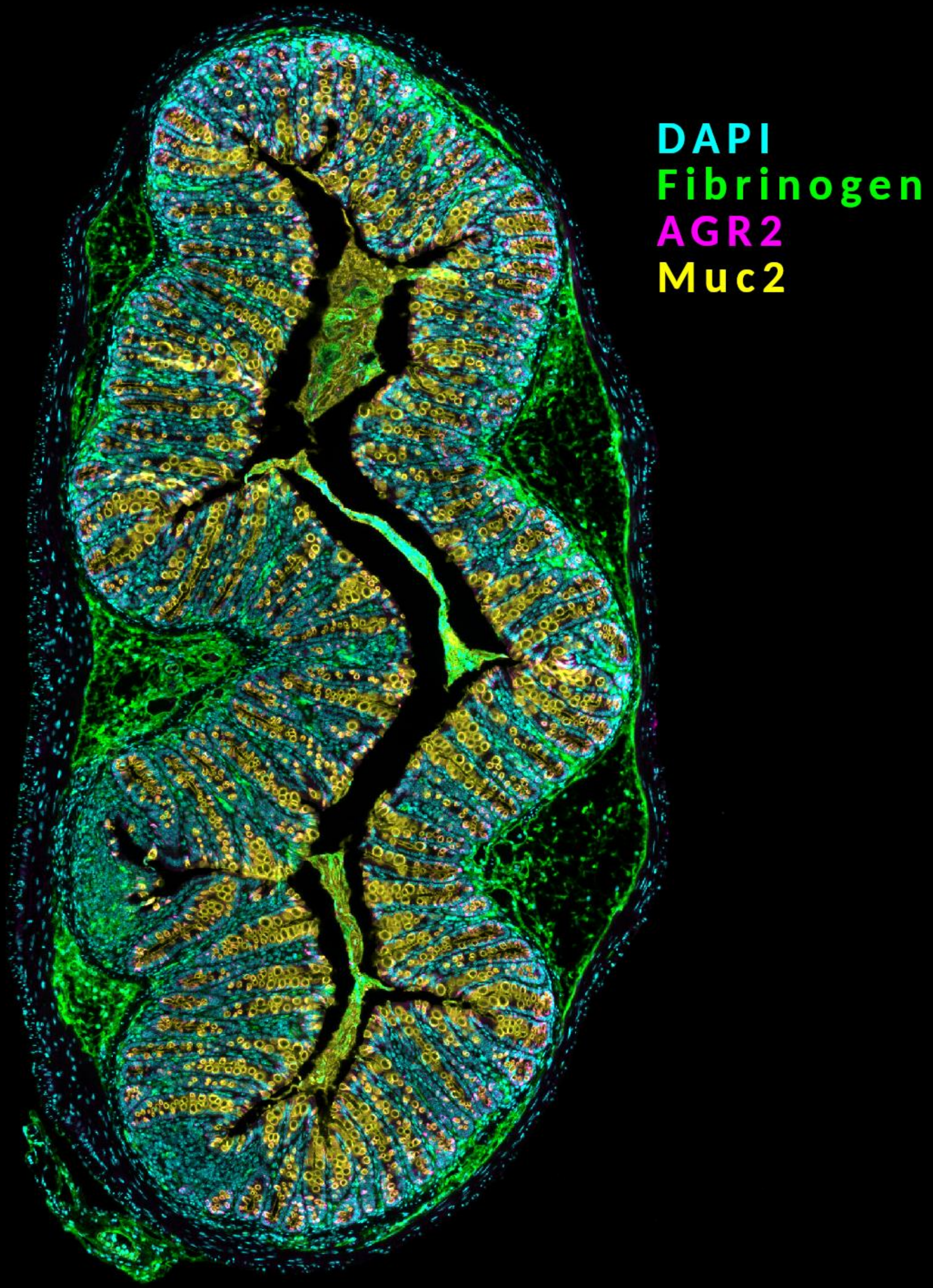
Too many annotations can lead to **over-fitting**: works great on training data but poorly on new data

Shallow Machine Learning for Pixel Classification

- Select **image features appropriate** to the classification problem
- Manually annotate regions/objects that are **representative** of what is seen in images
- Use **V** tool for annotations to **avoid over-representation**
- Define roughly the **same amount** of annotations for **each class**
- **Do not** manually annotate an **entire region of slide** to avoid over-fitting

Pixel Classification

- **Open** Colon_1.ome.tif and Colon_2.ome.tif
- Create **annotations** in each image that recapitulate the **diversity** of the tissue
- Create **regions annotations**
- Open **"Pixel classifier"**
- **Annotate** pixels belonging to **Mucus, Epithelium without Mucus** and **everything else**
- **Save classifier** and apply it to **each image** with a **script** (workflow tab)
- Get **proportions** of **tissues**



Cell Detection with Star-convex Polygons

Uwe Schmidt[1,⋆], Martin Weigert[1,⋆], Coleman Broaddus[1], and Gene Myers[1,2]

[1] Max Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany
Center for Systems Biology Dresden, Germany
[2] Faculty of Computer Science, Technical University Dresden, Germany

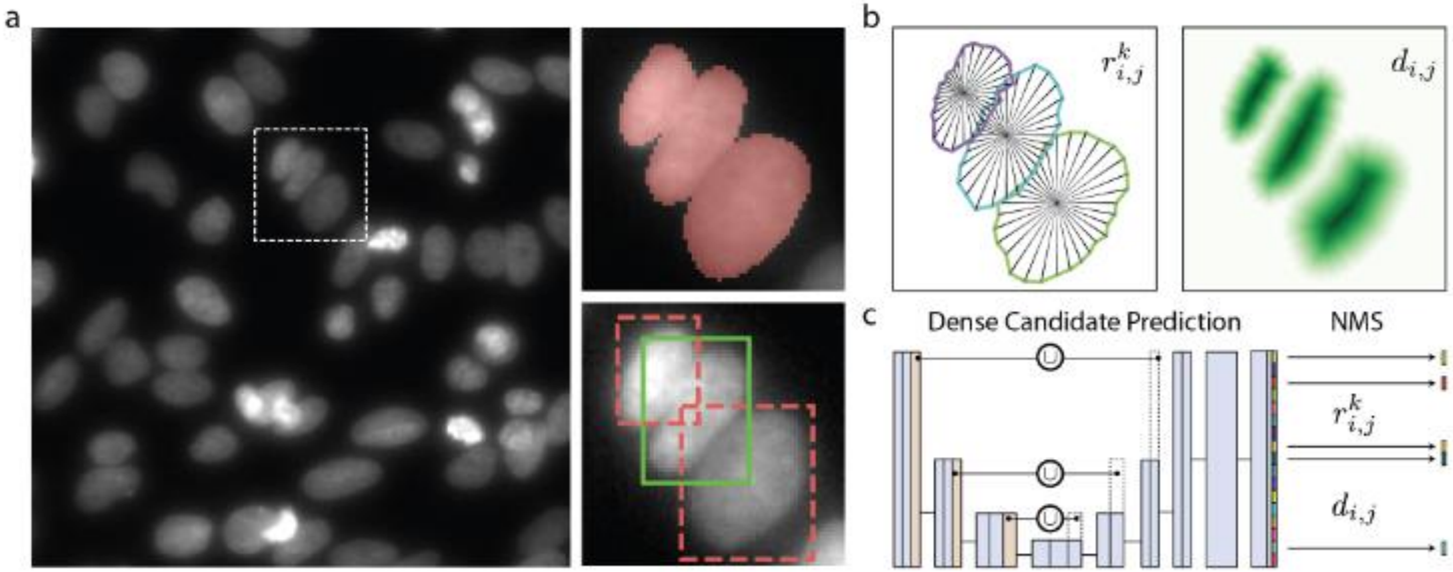
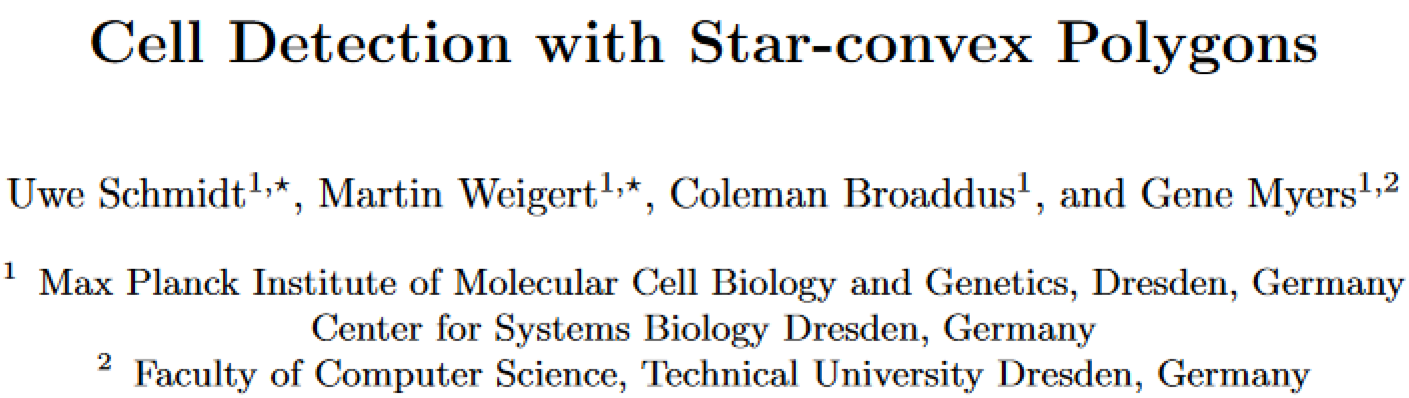


Fig. 1: *(a)* Potential segmentation errors for images with crowded nuclei: Merging of touching cells (upper right) or suppression of valid cell instances due to large overlap of bounding box localization (lower right). *(b)* The proposed StarDist method predicts object probabilities $d_{i,j}$ and star-convex polygons parameterized by the radial distances $r_{i,j}^k$. *(c)* We densely predict $r_{i,j}^k$ and $d_{i,j}$ using a simple U-Net architecture [15] and then select the final instances via non-maximum suppression (NMS).



Cell Detection with Star-convex Polygons

Uwe Schmidt[1,*], Martin Weigert[1,*], Coleman Broaddus[1], and Gene Myers[1,2]

[1] Max Planck Institute of Molecular Cell Biology and Genetics, Dresden, Germany
Center for Systems Biology Dresden, Germany
[2] Faculty of Computer Science, Technical University Dresden, Germany

For the workshop, a Stardist model was trained with data coming from 3 articles:

- **Whole-cell segmentation of tissue images with human-level performance using large-scale data annotation and deep learning.** *Nature Biotechnology (2022).*
- **A deep learning segmentation strategy that minimizes the amount of manually annotated images.** *F1000 Research (2022).*
- **Deep learning tools and modeling to estimate the temporal expression of cell cycle proteins from 2D still images.** *PLOS Computational Biology (2022).*

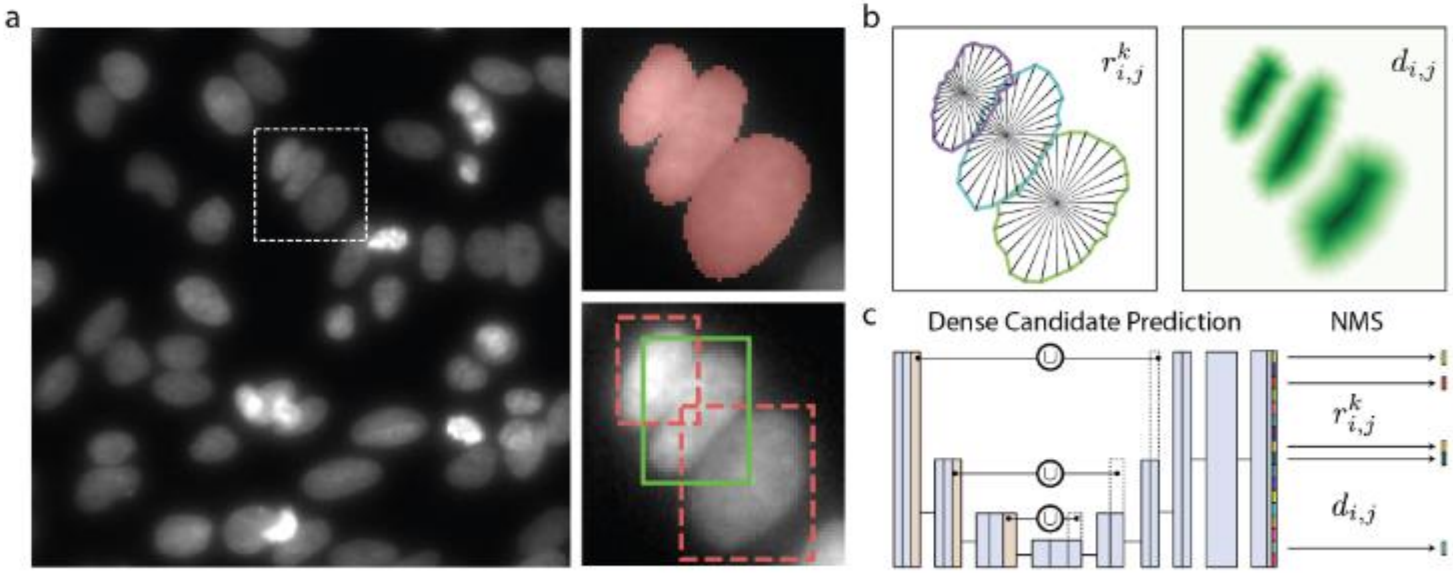
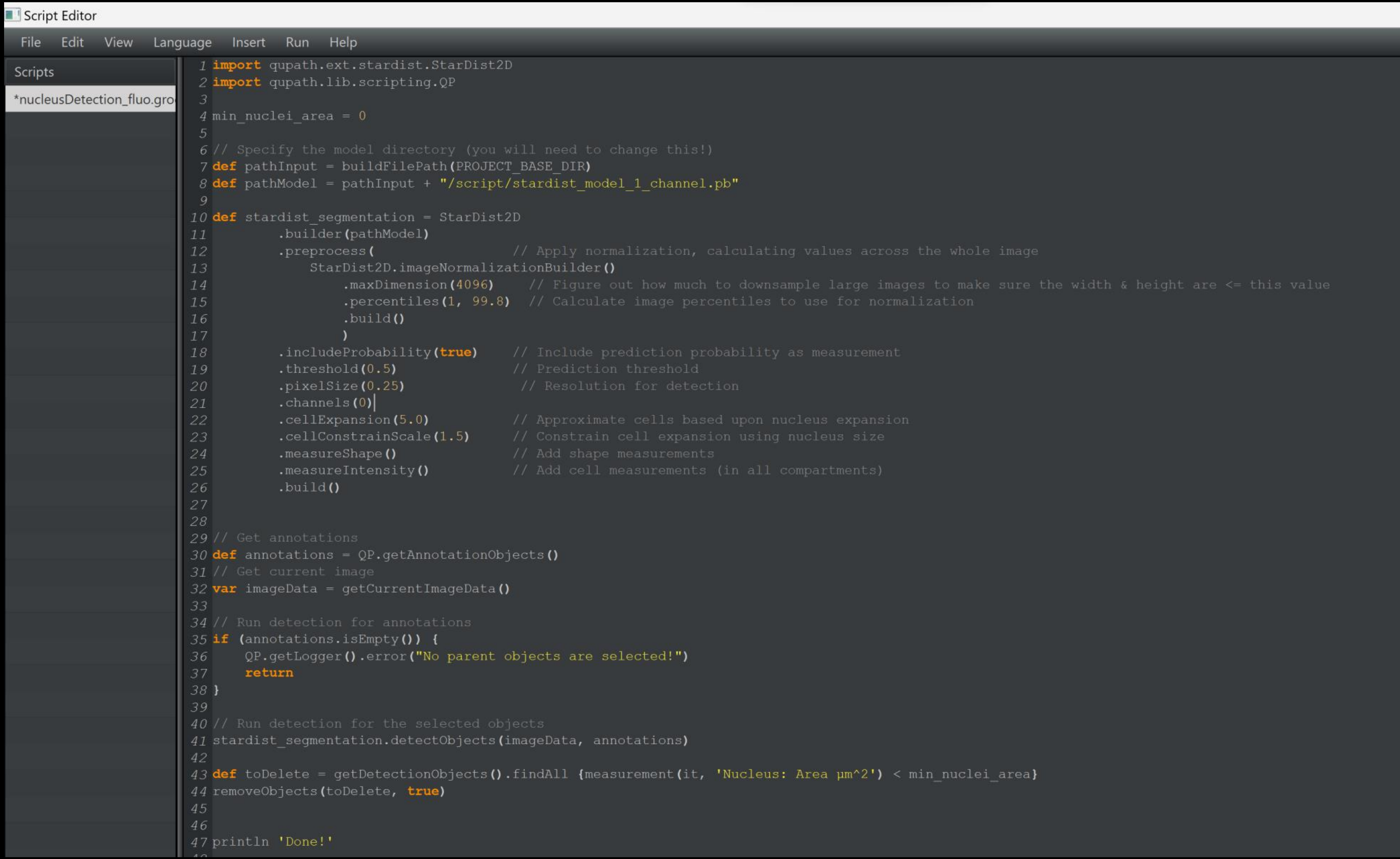


Fig. 1: *(a)* Potential segmentation errors for images with crowded nuclei: Merging of touching cells (upper right) or suppression of valid cell instances due to large overlap of bounding box localization (lower right). *(b)* The proposed StarDist method predicts object probabilities $d_{i,j}$ and star-convex polygons parameterized by the radial distances $r^k_{i,j}$. *(c)* We densely predict $r^k_{i,j}$ and $d_{i,j}$ using a simple U-Net architecture [15] and then select the final instances via non-maximum suppression (NMS).

SEGMENTATION WITH STARDIST

- **Copy** script folder into QuPath project
- **Open** nucleusDetection_fluo.groovy



```
import qupath.ext.stardist.StarDist2D
import qupath.lib.scripting.QP

min_nuclei_area = 0

// Specify the model directory (you will need to change this!)
def pathInput = buildFilePath(PROJECT_BASE_DIR)
def pathModel = pathInput + "/script/stardist_model_1_channel.pb"

def stardist_segmentation = StarDist2D
        .builder(pathModel)
        .preprocess(                 // Apply normalization, calculating values across the whole image
            StarDist2D.imageNormalizationBuilder()
                .maxDimension(4096)    // Figure out how much to downsample large images to make sure the width & height are <= this value
                .percentiles(1, 99.8)  // Calculate image percentiles to use for normalization
                .build()
                )
        .includeProbability(true)    // Include prediction probability as measurement
        .threshold(0.5)              // Prediction threshold
        .pixelSize(0.25)              // Resolution for detection
        .channels(0)
        .cellExpansion(5.0)          // Approximate cells based upon nucleus expansion
        .cellConstrainScale(1.5)     // Constrain cell expansion using nucleus size
        .measureShape()              // Add shape measurements
        .measureIntensity()          // Add cell measurements (in all compartments)
        .build()


// Get annotations
def annotations = QP.getAnnotationObjects()
// Get current image
var imageData = getCurrentImageData()

// Run detection for annotations
if (annotations.isEmpty()) {
    QP.getLogger().error("No parent objects are selected!")
    return
}

// Run detection for the selected objects
stardist_segmentation.detectObjects(imageData, annotations)

def toDelete = getDetectionObjects().findAll {measurement(it, 'Nucleus: Area µm^2') < min_nuclei_area}
removeObjects(toDelete, true)


println 'Done!'
```

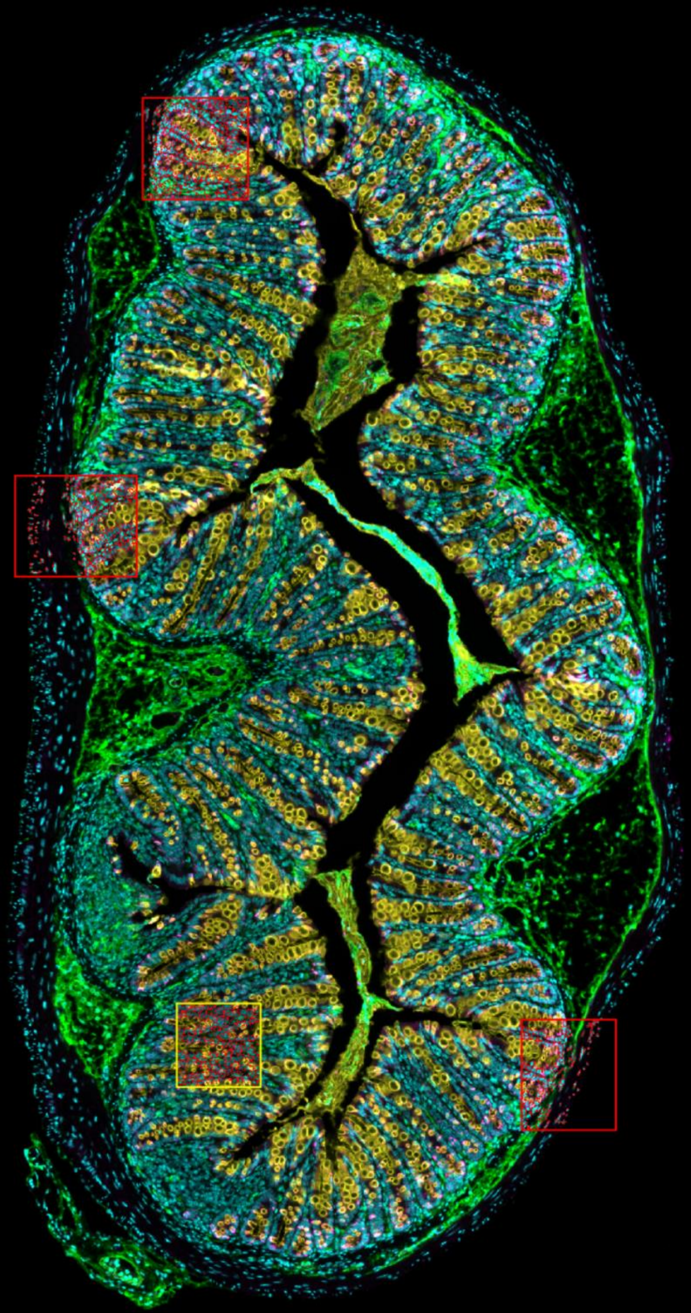
SEGMENTATION WITH STARDIST

- **Copy** script folder into QuPath project

- **Open** nucleusDetection_fluo.groovy

- Define a **small rectangle annotation** to test the **Stardist parameters**

- **Test** on **few small other** rectangle annotations



Shallow Machine Learning for Object Classification

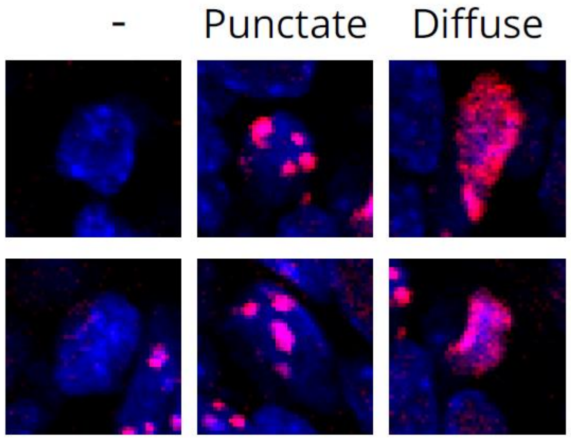
As for pixel classification:

- **Examples** of classes are **manually** defined by the user
- A **classifier** is **trained** with these examples
- Data is then **automatically classified** by using the trained classifier

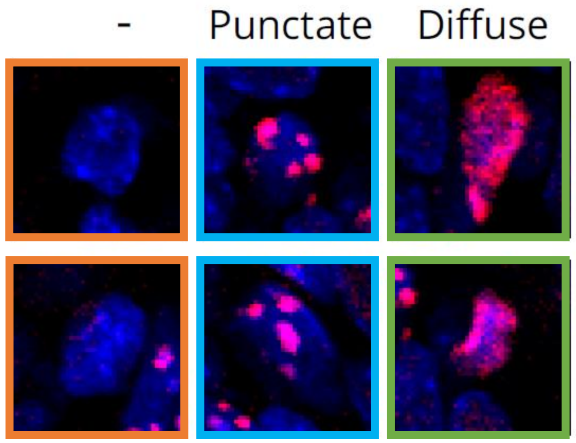
But this time, features are **measurements associated to detections** (most often cells or nuclei) such as:

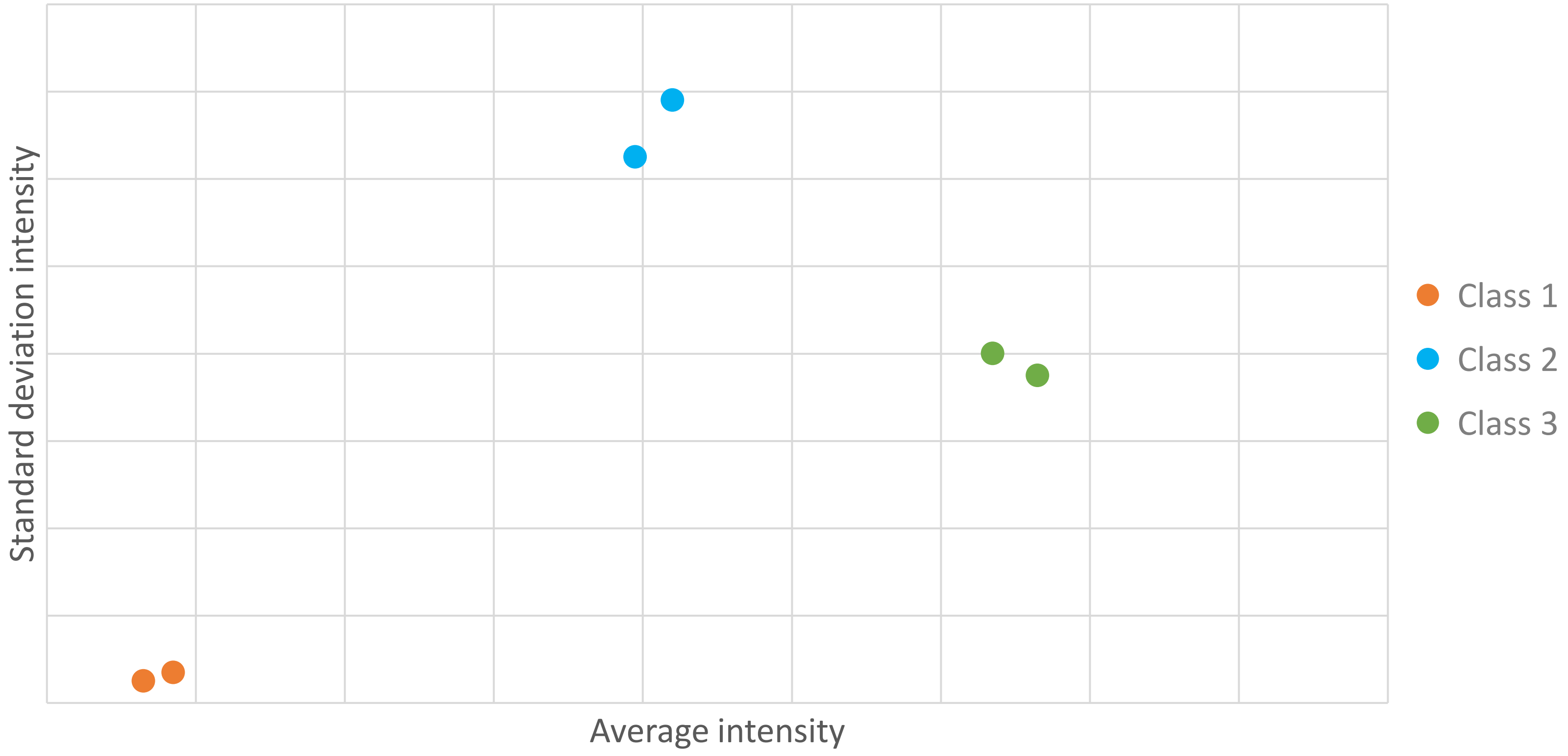
- **Average** intensity
- **Median** intensity
- **Standard deviation** of intensity
- **Minimum/Maximum** intensity
- Object **area**
- Object **Circularity**
- …

Shallow Machine Learning for Object Classification

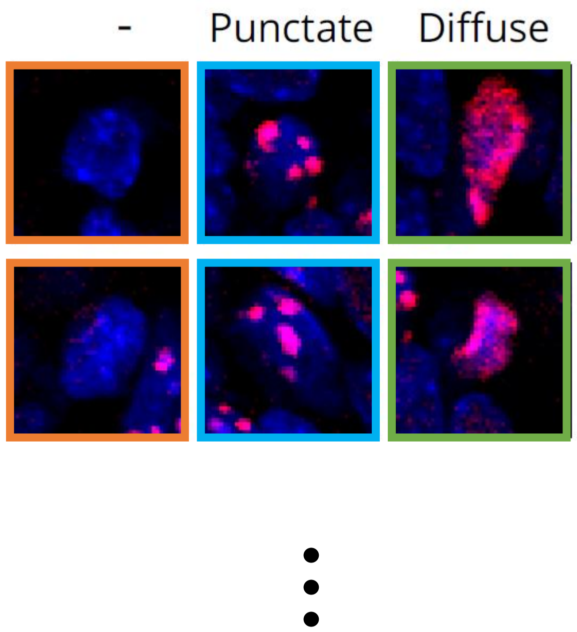


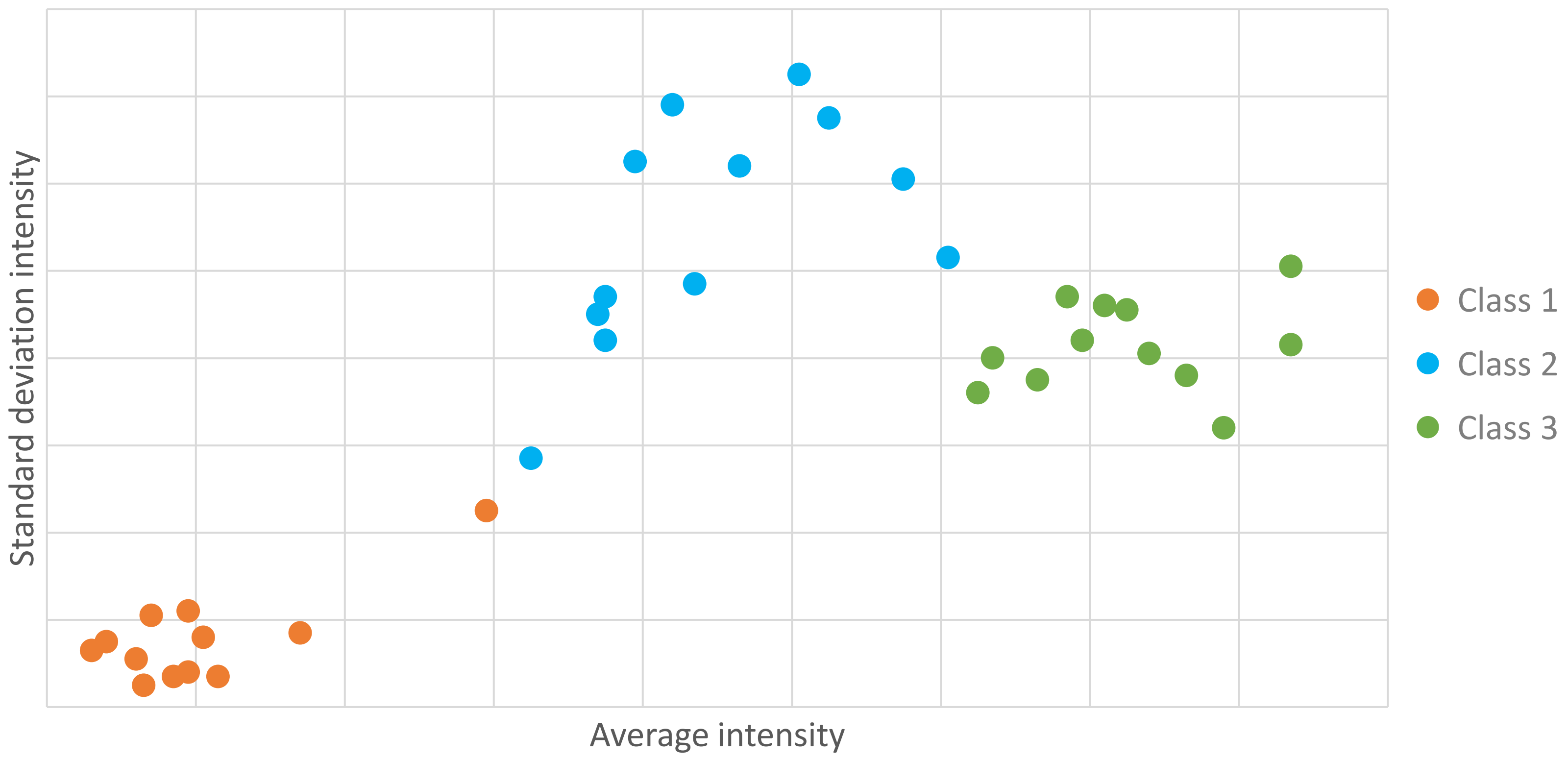
Shallow Machine Learning for Object Classification



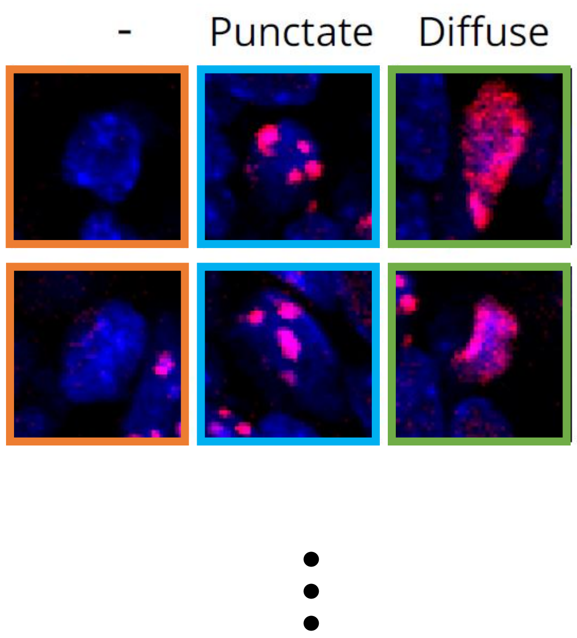


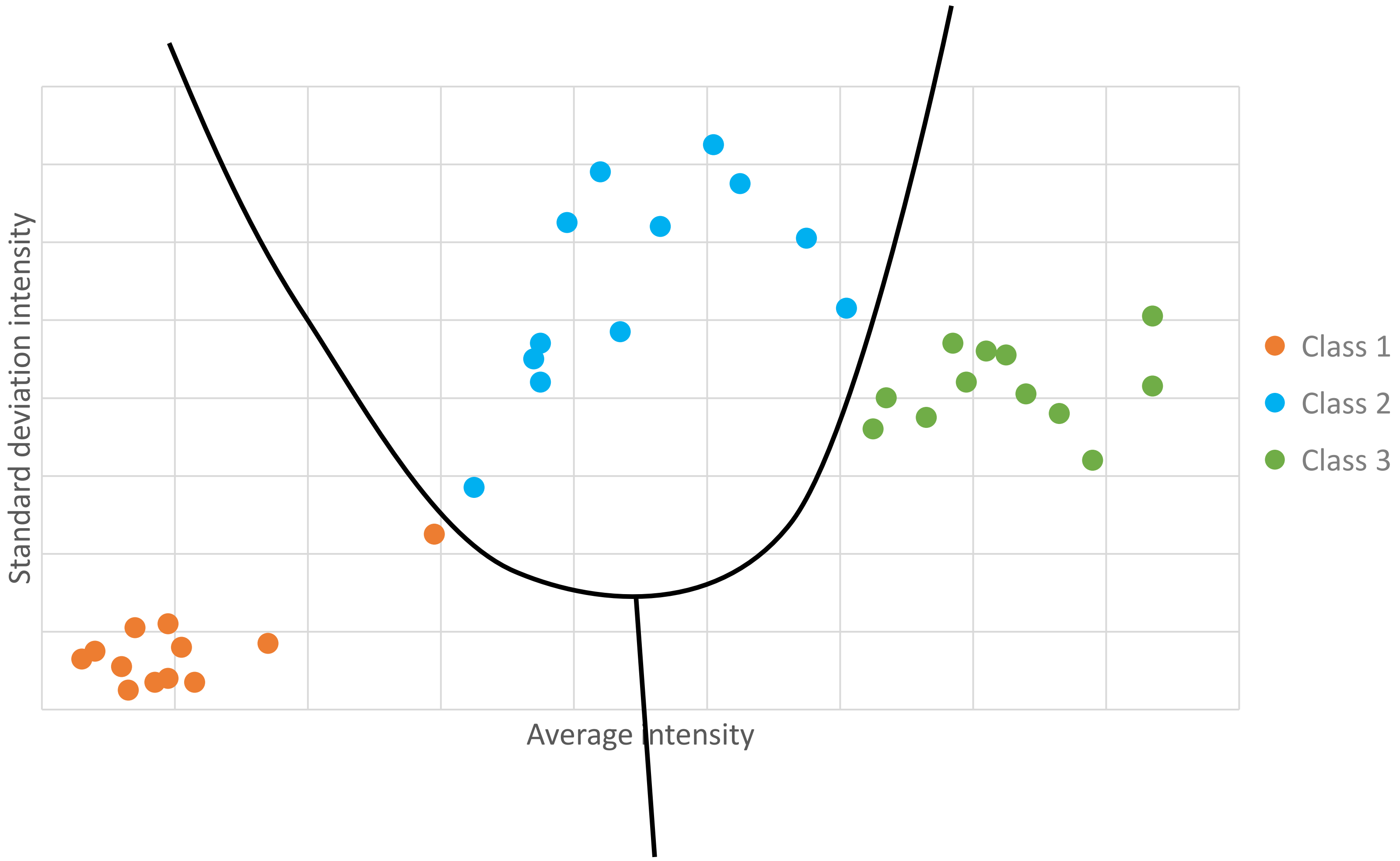
Shallow Machine Learning for Object Classification





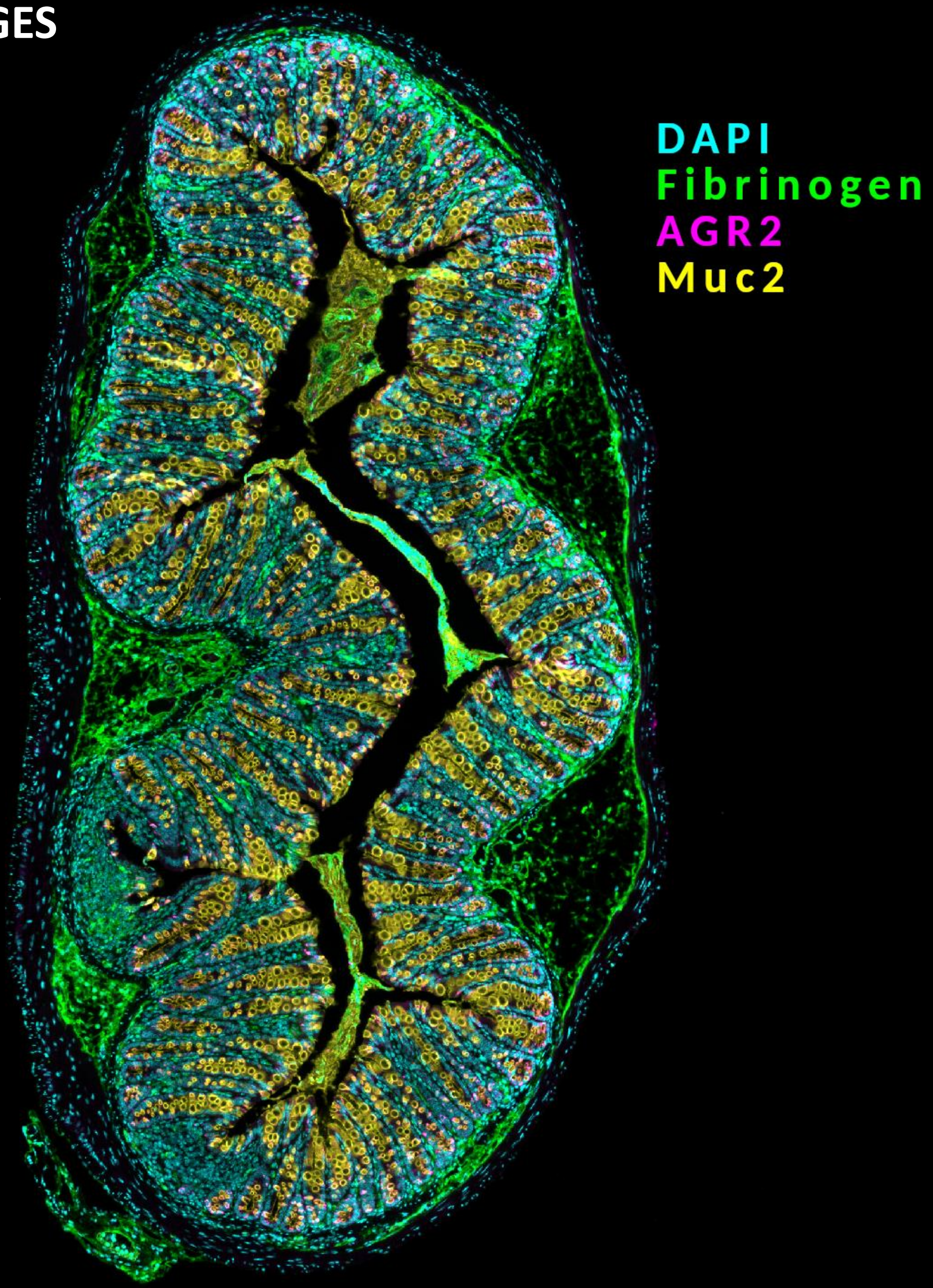
Shallow Machine Learning for Object Classification





Multi/Hyper-plexed Images

- Create **annotations** in each image that recapitulate the **diversity** of the tissue
- Create **regions annotations**
- Apply **Stardist**
- **Train** an **object** classifier to identify positive cells for Fibrinogen and AGR2
- Apply **Stardist** on the Colon_1, **classify cells** and apply **pixel classifier**
- Use the **workflow tab** to create a **script** that applies these **3 tasks** to Colon_2
- Compute **distances** between AGR2 positive cells and mucus
- **Export measurements** for both images



Citations

- P. Bankhead *et al.* **QuPath: Open source software for digital pathology image analysis**. *Scientific Reports* (2017). https://doi.org/10.1038/s41598-017-17204-5

- U. Schmidt *et al.* **Cell Detection with Star-convex Polygons.** *International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)* (2018). https://arxiv.org/abs/1806.03535

- N.F. Greenwald *et al.* **Whole-cell segmentation of tissue images with human-level performance using large-scale data annotation and deep learning.** *Nature Biotechnology (2021)*. https://doi.org/10.1038/s41587-021-01094-0

- T. Pécot *et al.* **A deep learning segmentation strategy that minimizes the amount of manually annotated images.** *F1000 Research (2022)* https://doi.org/10.12688/f1000research.52026.2

- T. Pécot *et al.* **Deep learning tools and modeling to estimate the temporal expression of cell cycle proteins from 2D still images.** *PLOS Computational Biology (2022)*

VIDEO TUTORIALS

- QuPath installation, data and script downloading
- Project creation and annotations
- Stain deconvolution
- Pixel classification (epithelium/stroma for H&E images)
- Nuclei segmentation (watershed) and DAB positive cells
- Nuclei segmentation (stardist) and DAB positive cells (thresholding)
- Visualization of fluorescence images
- Pixel classification (epithelium/stroma for fluorescence images)
- Nuclei segmentation (stardist)
- Object classification for marker identification