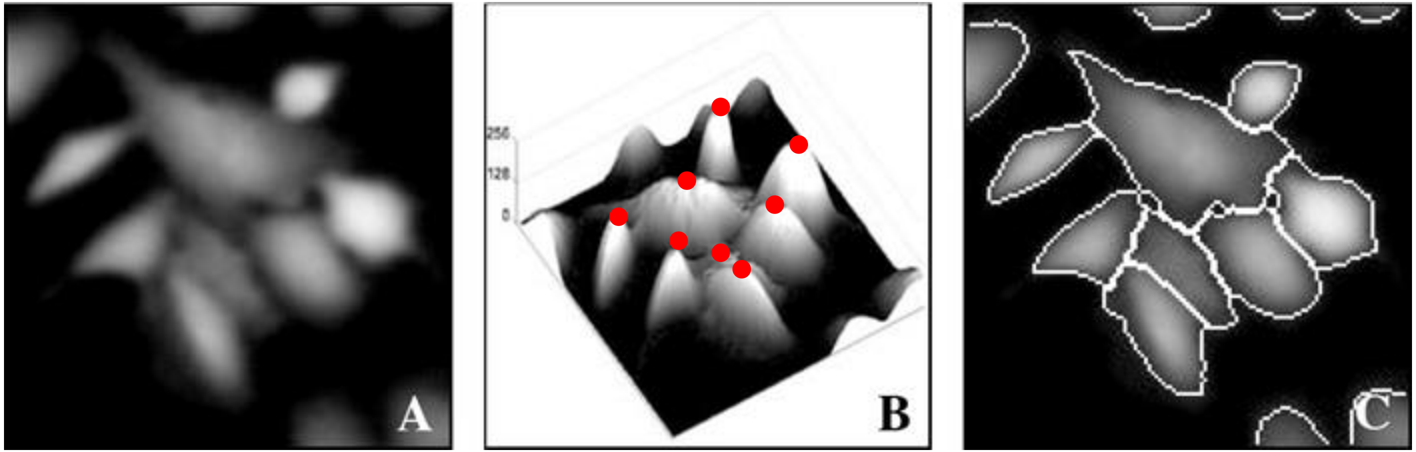
Introduction to Instance Segmentation

Beth Cimini, PhD
Associate Director, Bioimage Analysis
Imaging Platform, Broad Institute of MIT and Harvard

Slides available at broad.io/InstanceDL_2024

Instance segmentation

- We next need to distinguish multiple objects contained in the same “clump”



Images from Carolina Wahlby

Image Processing Modules

- All classical image analysis segmentation works off of the principle- “my image should have my objects of interest bright and everywhere else dark”
- ***<u>For segmentation purposes</u>***, basically any transformation you can think of to achieve this is “legal”
  - ***<u>Emphatically</u>***, this does not apply to images you are actively measuring, just images to be fed into a segmentation
- To achieve this, you may need to enhance dim signal that really is there and/or mask out bright signal that truly is (but is not supposed to be segmented at the current time)
- The more transformations you do before segmentation, the more critical it is to check your segmentations against your raw images

Classical Image Processing Options

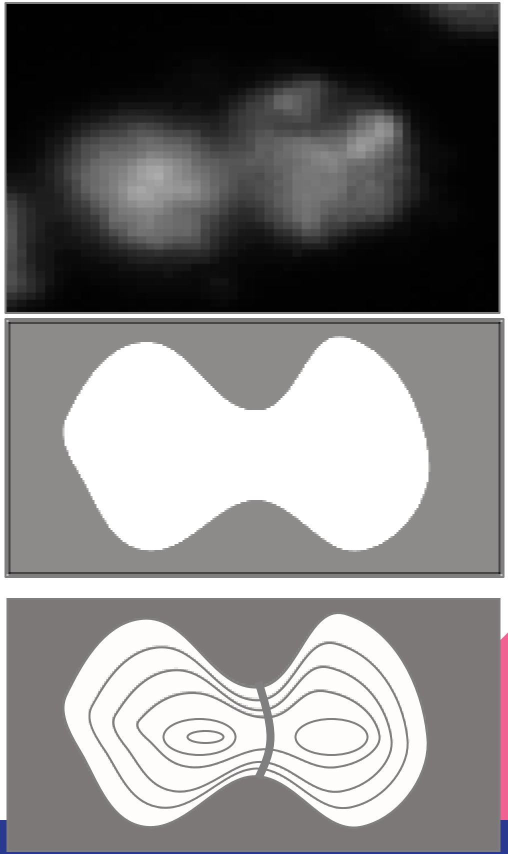
(Not exhaustive, just inspirational)

- EnhanceEdges
- EnhanceOrSuppressFeatures- works on tubes, speckles, and/or dark holes
- Smooth
- Crop
- ColorToGray / GrayToColor
- UnmixColors
- GaussianFilter
- MedialAxis
- RemoveHoles
- MatchTemplate

- PLUS, classical (such as Random Forest based) semantic segmenters

Object identification

- Once the images are loaded, how do you find objects of interest?

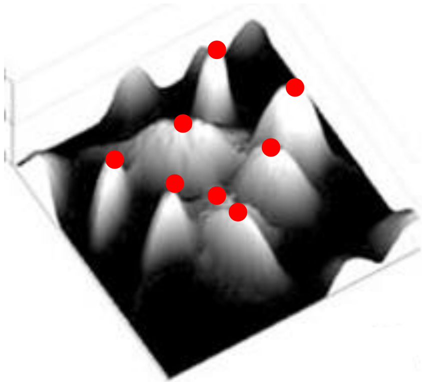


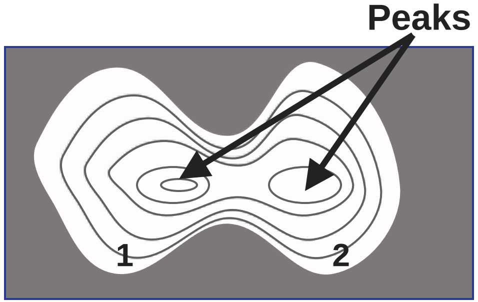
- **Step 1:** Distinguish the foreground from the background (thresholding)
- **Step 2:** Split/merge "objects" properly

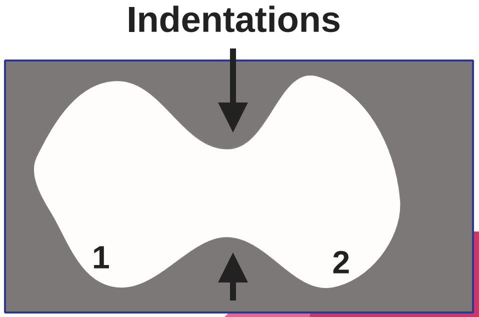
Separating touching objects – clump identification

- **Intensity:** Works best if objects are brighter at center, dimmer at edges

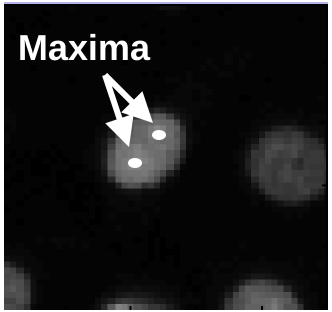
- **Shape:** Works best if objects have indentations where clumps touch (esp. if objects are round)



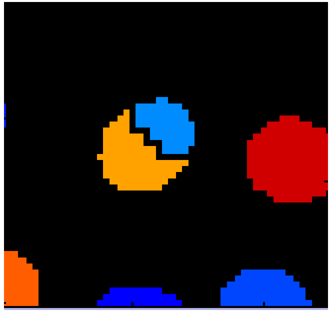




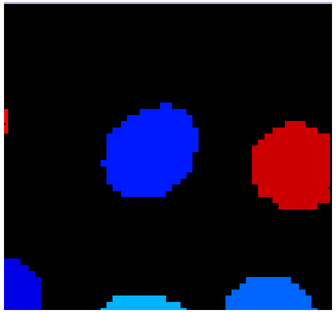
Separating touching objects – clump identification



Original image



Maxima
distance = 4

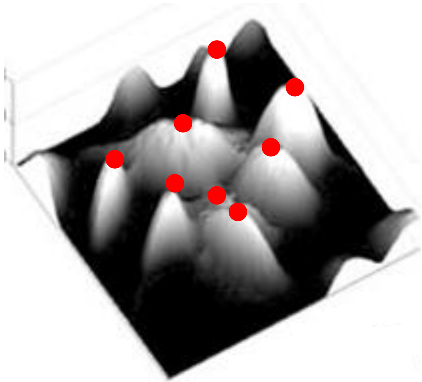


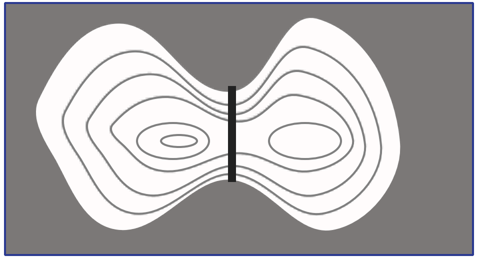
Maxima
distance = 8

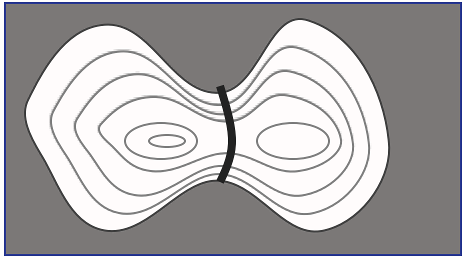
- Suppress Local Maxima
- Smallest distance allowed between object intensity peaks to be considered one object rather than a clump
- Decrease to reduce improper merging of objects in clumps

Separating touching objects

- Drawing boundaries: Two options
  - **Shape:** Draws boundary lines at the indentations
  - **Intensity:** Draws boundary lines at dimmest line between objects







So to sum up, here's how classical instance segmentation works

- Optionally, do SOMETHING to make your image "everything I care about is bright, and everything else is dark"
- Decide on a threshold for that image
- Then either
    - Say every pixel that touches its neighbors after thresholding is part of the same object (not shown above)
- OR
    - Figure out the centers (and thereby also, the count) of objects you have, based on some rational principle
    - Follow some rational principle to trace from the centers to the edges

Sorry, excuse me Beth, this is a deep learning course, why are you telling us irrelevant stuff?

Sorry, my bad, here’s how deep learning instance segmentation (typically) works

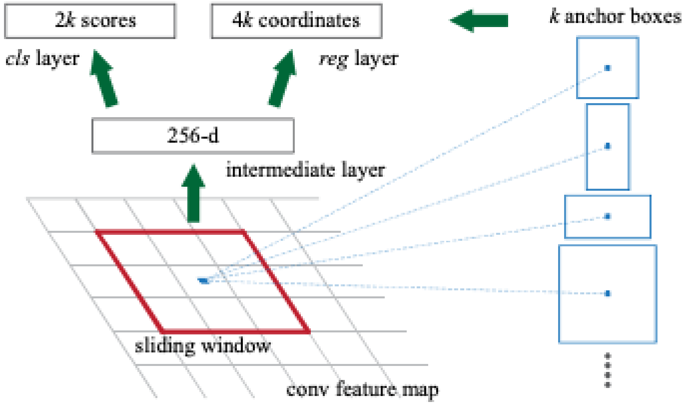
- Optionally, do SOMETHING to make your image “everything I care about is bright, and everything else is dark”
- Decide on a threshold for that image
- Also, either
  - Say every pixel that touches its neighbors after thresholding is part of the same object (not shown above)
- OR (one or both, in either order)
  - Figure out the approximate locations (centers and/or bounding boxes) of objects you have, based on some rational principle
  - Follow some rational principle to pull pixels together

Deep learning for object locations

- Optionally, do SOMETHING to make your image “everything I care about is bright, and everything else is dark”
- Decide on a threshold for that image
- Also, either
  - Say every pixel that touches its neighbors after thresholding is part of the same object (not shown above)
- OR (one or both, in either order)
  - Figure out the approximate locations (centers and/or bounding boxes) of objects you have, based on some rational principle
  - Follow some rational principle to pull pixels together

DL for approximate location finding – Region Proposal Networks (RPNs)

- Take in an image, put out bounding boxes of likely objects
- Used in things like Faster R-CNN (object detection) Mask R-CNN
- Can share convolutional layers with downstream tasks, reducing extra time required



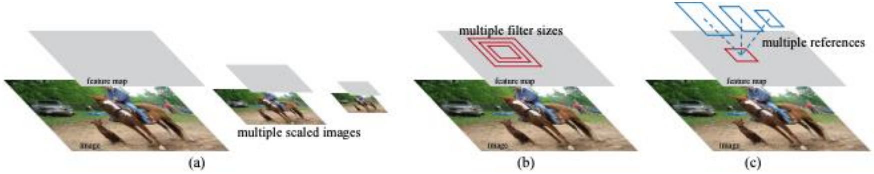


Figure 1: Different schemes for addressing multiple scales and sizes. (a) Pyramids of images and feature maps are built, and the classifier is run at all scales. (b) Pyramids of filters with multiple scales/sizes are run on the feature map. (c) We use pyramids of reference boxes in the regression functions.

Ren, He, Girshick, and Sun, NeurIPS 2015

Mask-R-CNN

- Supports overlapping objects
- RPN (gets class and box), PLUS n-classes binary semantic segmenters

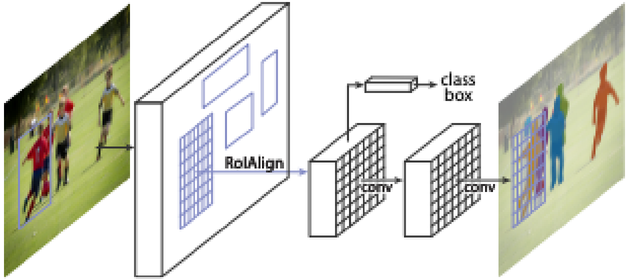
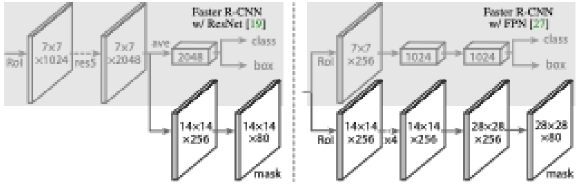
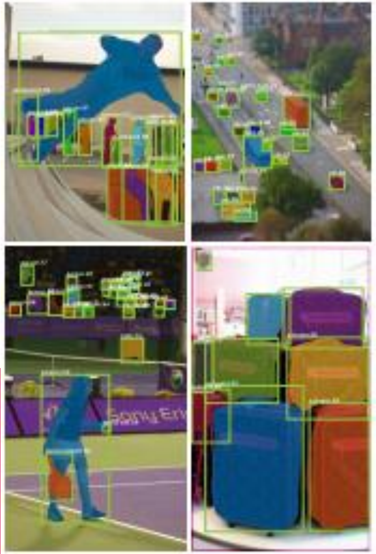


Figure 1. The **Mask R-CNN** framework for instance segmentation.





He, Gkioxari, Dollár, and Girshick, ICCV 2017

From each pixel, a distance polygon - StarDist

- Predict for each pixel how far it is from the edge of its object in each of 32 directions, making n-pixels polygons; also predict object probability
- Keep the highest-probability polygons
  - Later versions add some "majority-rules" shape refinement, to make the final thing less polygonal

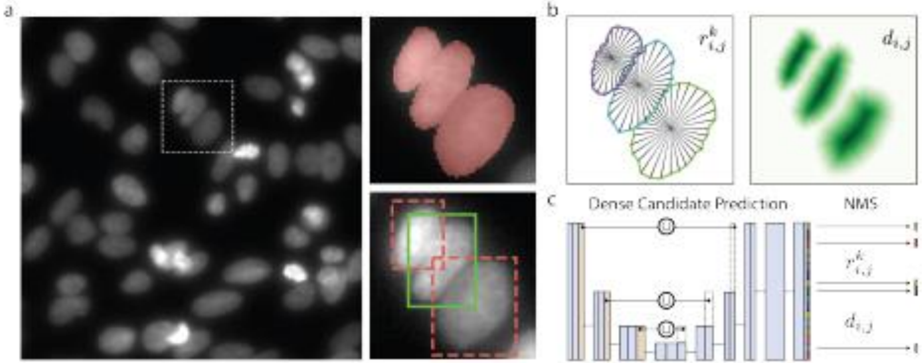


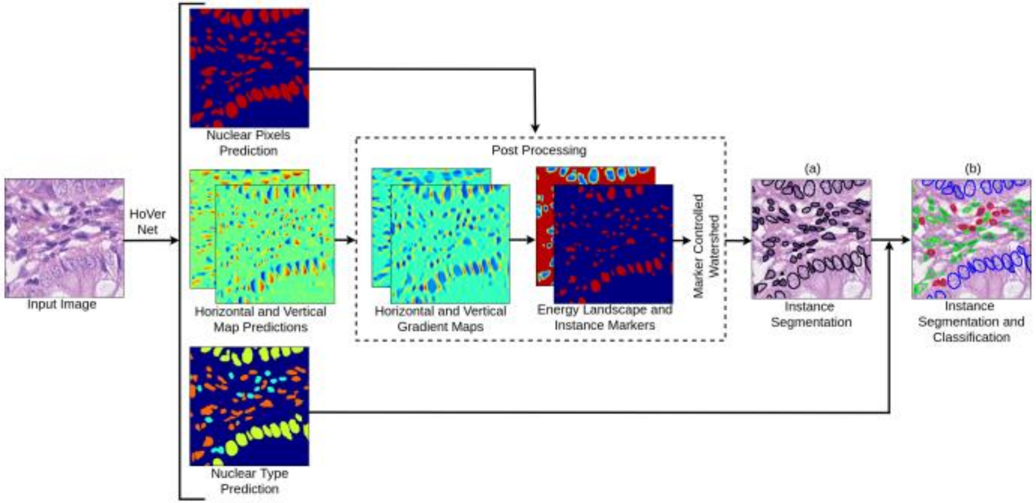
Fig. 1: *(a)* Potential segmentation errors for images with crowded nuclei: Merging of touching cells (upper right) or suppression of valid cell instances due to large overlap of bounding box localization (lower right). *(b)* The proposed StarDist method predicts object probabilities $d_{i,j}$ and star-convex 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).

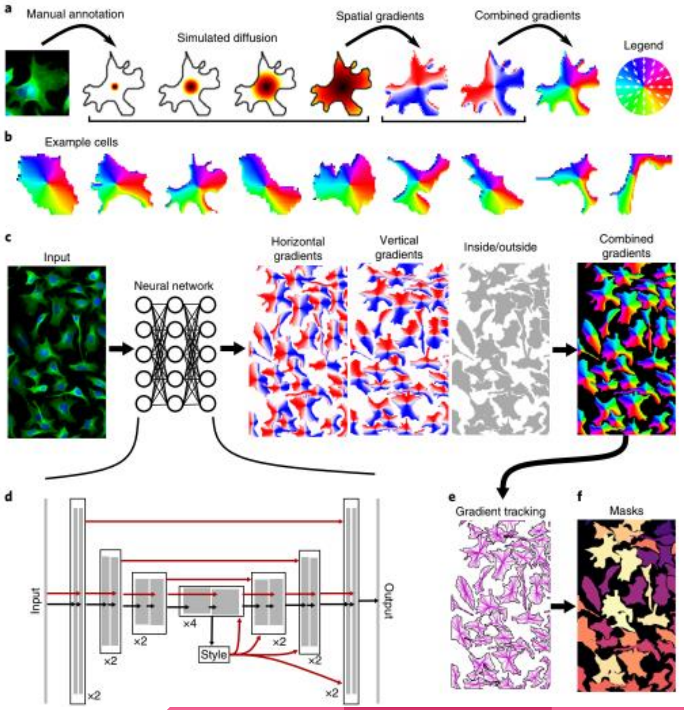
Schmidt, Weigert, Broaddus and Myers MICCAI 2018

Deep learning for together-ness

- Optionally, do SOMETHING to make your image “everything I care about is bright, and everything else is dark”
- Decide on a threshold for that image
- Also, either
  - Say every pixel that touches its neighbors after thresholding is part of the same object (not shown above)
- OR (one or both, in either order)
  - Figure out the approximate locations (centers and/or bounding boxes) of objects you have, based on some rational principle
  - Follow some rational principle to pull pixels together

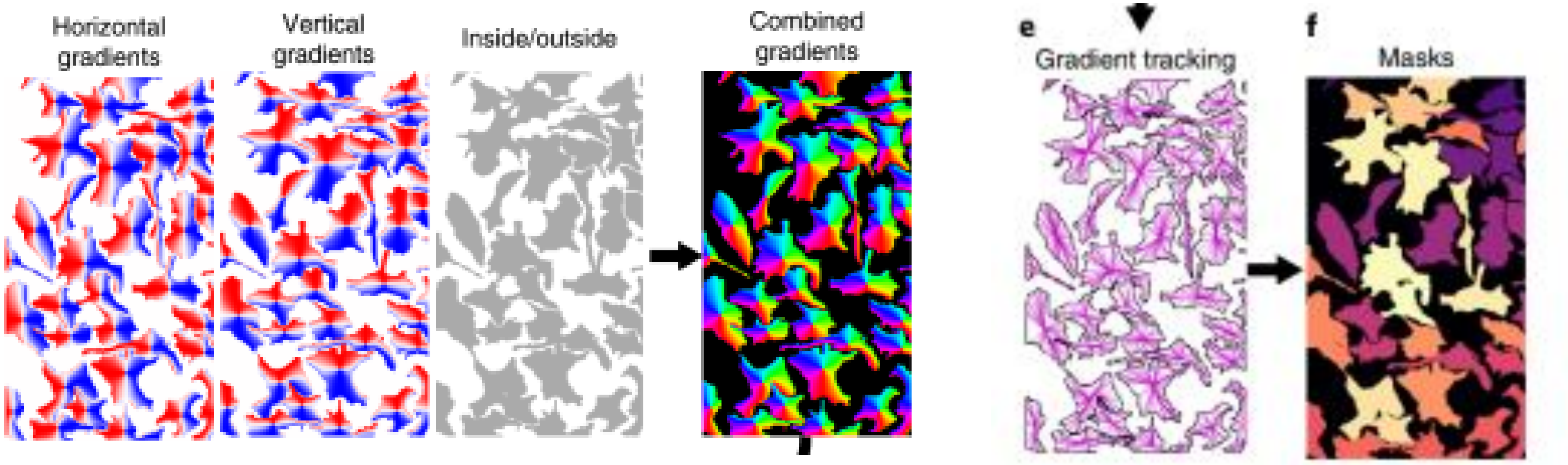
Gradient flows - Cellpose





Graham et al Medical Image Analysis 2019, Stringer et al Nat Meth 2021

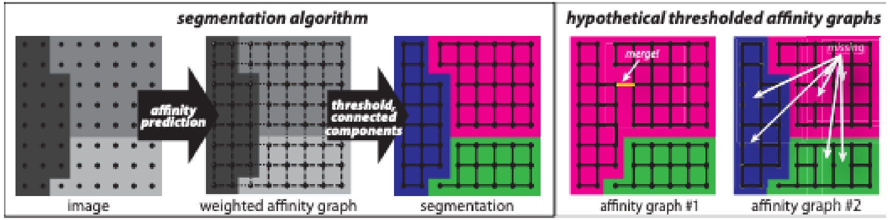
Gradient flows - Cellpose

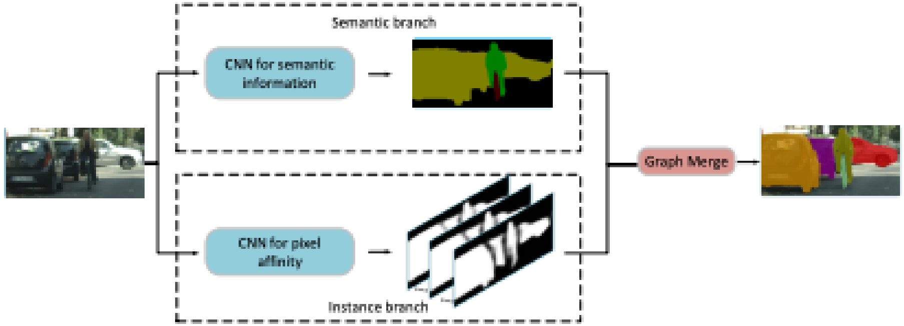


Stringer et al Nat Meth 2021

What belongs together – affinity learning

- Figure out what belongs with what, and group it together





Turaga et al 2009, Liu et al ECCV 2018

What belongs together – affinity learning

(a) (b)

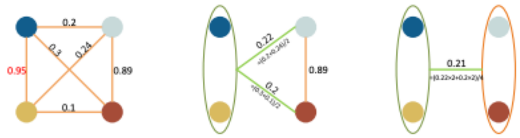


Fig. 4. A brief illustration for graph merge algorithm

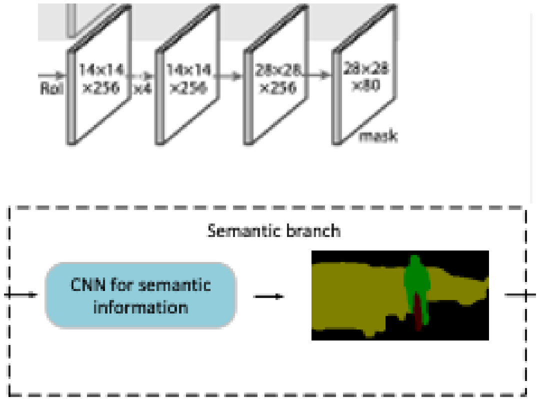
Fig. 6. Results on Cityscapes *val* dataset, original image, semantic result, instance result and ground truth from left to right. Results in last two rows are cropped from original ones for better visualization.

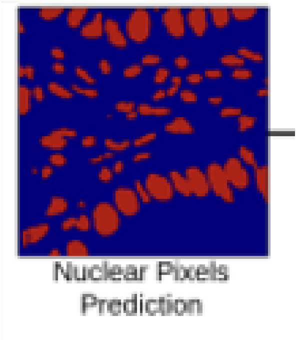
Liu et al ECCV 2018

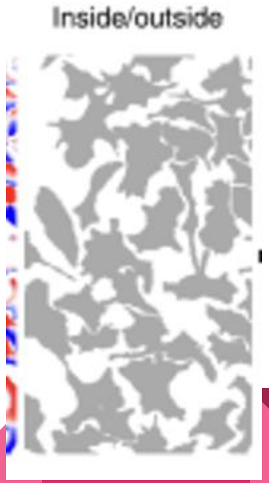
Deep learning for probability

- Optionally, do SOMETHING to make your image “everything I care about is bright, and everything else is dark”
- Decide on a threshold for that image
- Also, either
  - Say every pixel that touches its neighbors after thresholding is part of the same object (not shown above)
- OR (one or both, in either order)
  - Figure out the approximate locations (centers and/or bounding boxes) of objects you have, based on some rational principle
  - Follow some rational principle to pull pixels together

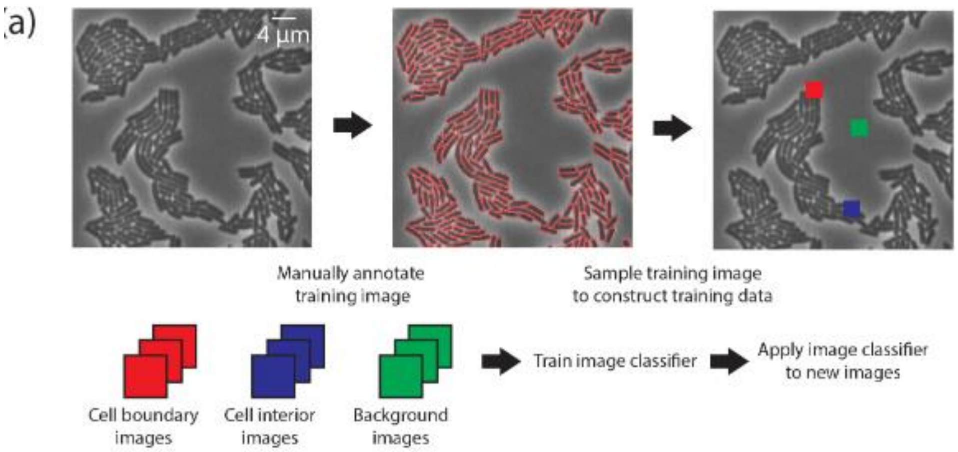
We’ve already seen this in the other things we’ve looked at too – can be combined with more or less fancy post-processing – typically (though not always) 2-class “inside the object” probabilities

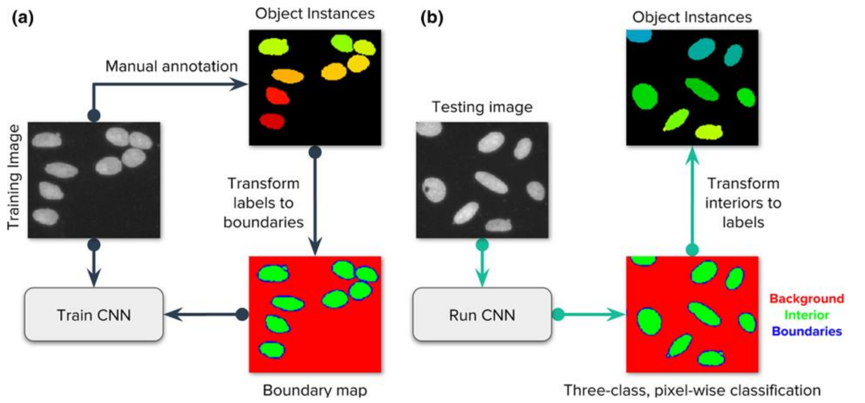






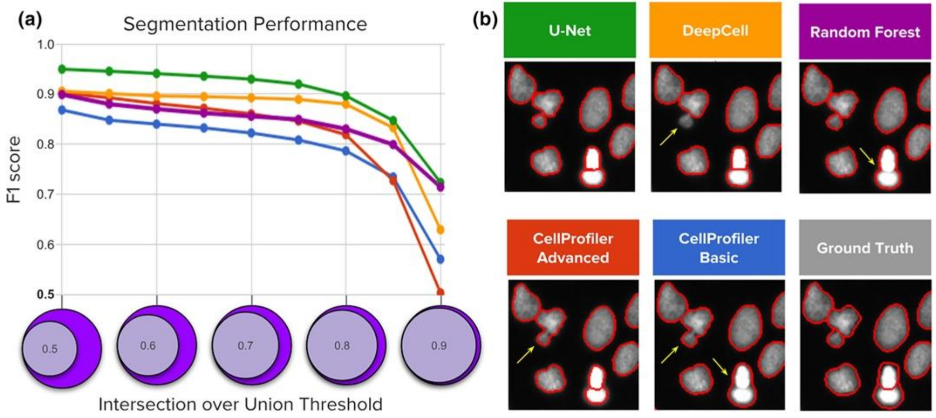
Semantic models based on 3 classes (cell boundary, interior, background) are also popular choices

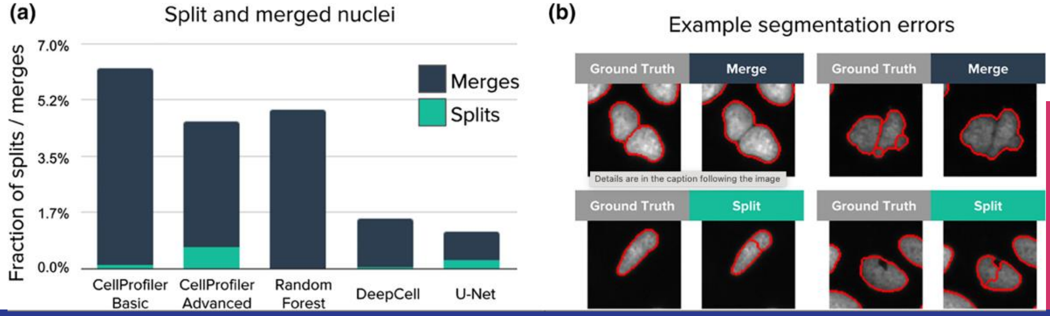




Van Valen et al 2016; Roth et al 2019

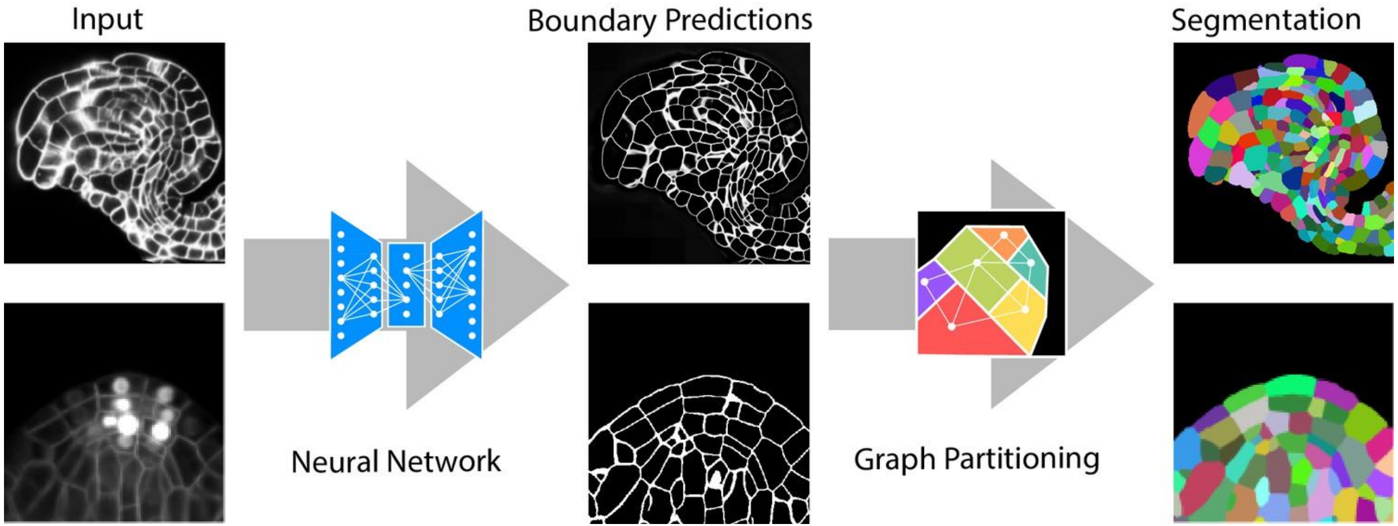
Models based on 3 classes (cell boundary, interior, background) are also popular choices





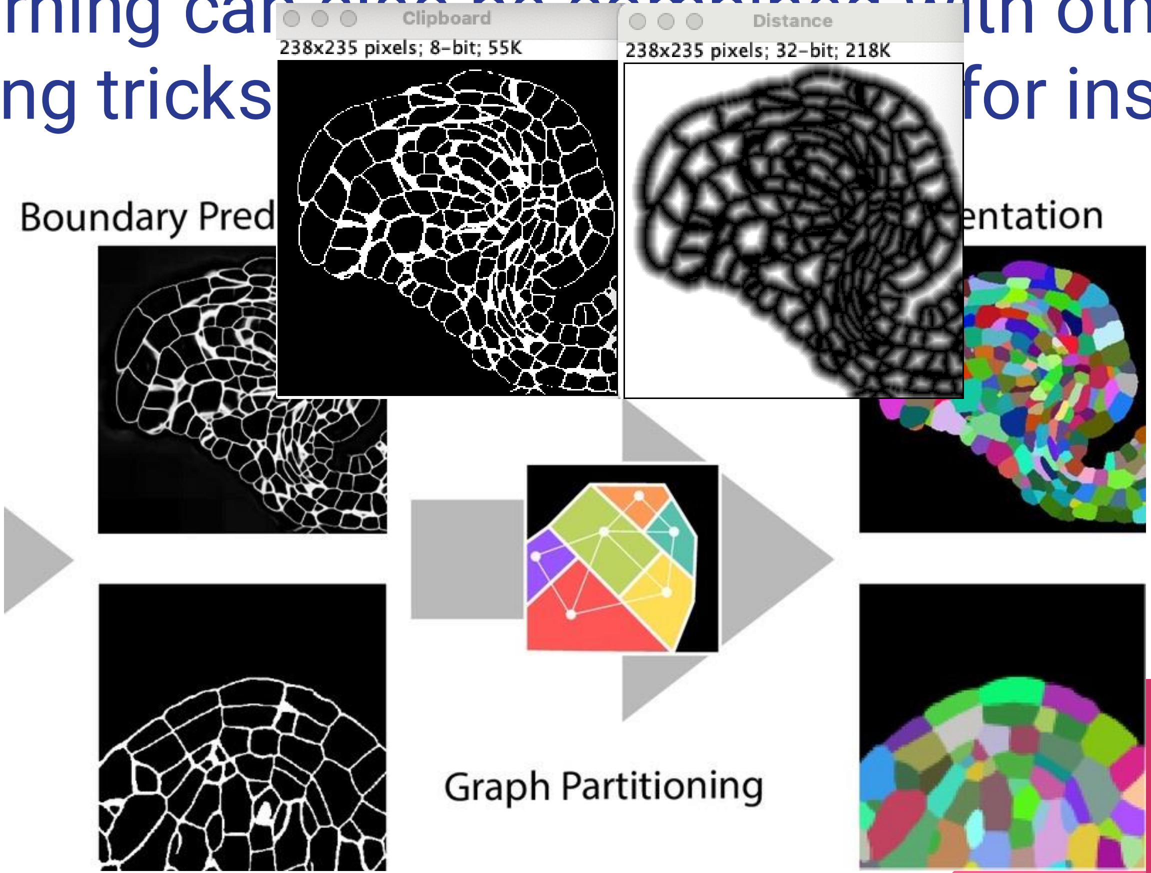
Roth et al 2019, Cytometry A

Deep learning can also be combined with other processing tricks to get the right input for instance-ing



Wolny et al, eLife 2020

Deep learning can also be combined with other processing tricks for instance-ing



Wolny et al, eLife 2020

So it sounds like I have a lot of options on how to put deep learning into my instance segmentation. Too many options maybe?

Tips for creating a good high content analysis workflow

- When finding the objects that you care about, ask yourself for your whole experiment:
  - Do I generally agree with most of the object segmentations from my analysis workflow?
  - Do I have an approximately equal number of regions/images where the threshold chosen by the algorithm for this image is a bit too low vs a bit too high?
  - Do I have an approximately equal number of oversegmentations/splits and undersegmentations/merges?
  - Very important: Do both the second and third bullet points hold true for both my negative control images and my positive control (or most extreme expected phenotype(s) sample) images?

https://carpenter-singh-lab.broadinstitute.org/blog/when-to-say-good-enough

Based on that and other things, here are some simple considerations

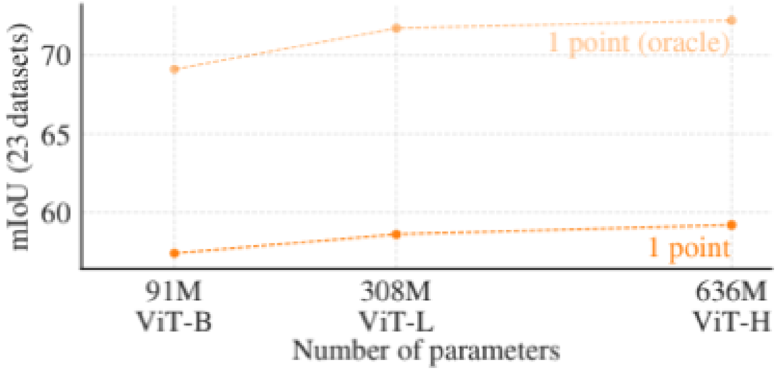
Think about… how much compute do you need (and how much do you have)?

- The more types of DL, and the presence of certain types of DL, the more parameters you'll have to tune

U-Net (2 class): We use the popular U-Net architecture [15] as a baseline to predict 2 output classes (cell, background). We use 3 down/up-sampling blocks, each consisting of 2 convolutional layers with $32 \cdot 2^k (k = 0, 1, 2)$ filters of size 3×3 (approx. 1.4 million parameters in total). We apply a threshold σ on the cell probability map and retain the connected components as final result (σ is optimized on the validation set for every dataset).

U-Net (3 class): Like U-Net (2 class), but we additionally predict the *boundary pixels* of cells as an extra class. The purpose of this is to differentiate crowded cells with touching borders (similar to [4,5]). We again use the connected components of the thresholded cell class as final result.

Mask R-CNN: A state-of-the-art instance segmentation method combining a bounding-box based region proposal network, non-maximum-suppression (NMS), and a final mask segmentation (approx. 45 million parameters in total). We use a popular open-source implementation[6]. For each dataset, we perform a grid-search over common hyper-parameters, such as detection NMS threshold, region proposal NMS threshold, and number of anchors.



Schmidt, et al MICCAI 2018; Kirillov et al 2023

Think about… how much compute do you need (and how much do you have)?

Segment Anything

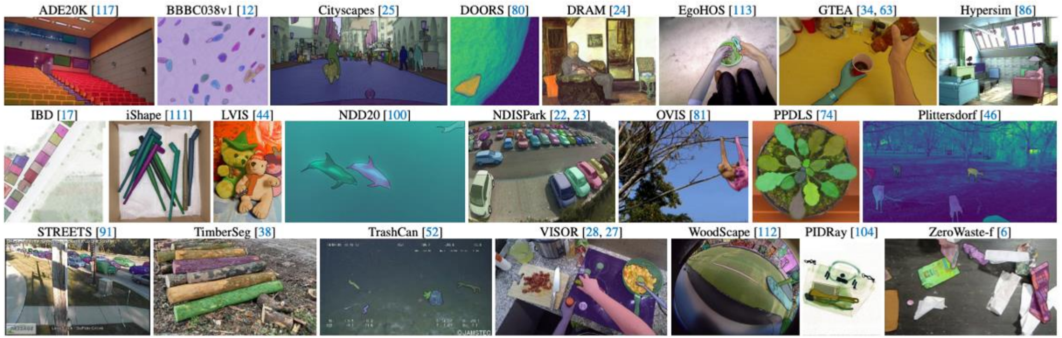
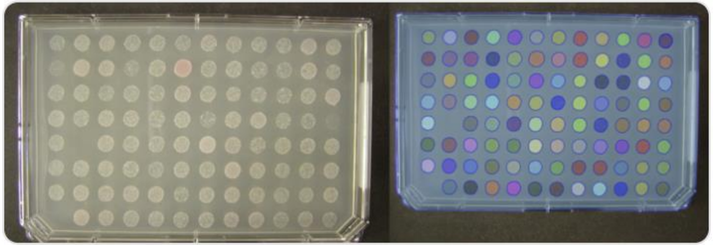
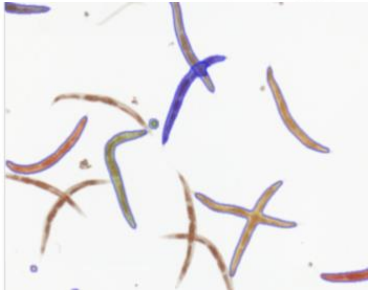
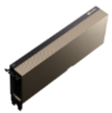


Figure 8: Samples from the 23 diverse segmentation datasets used to eval







NVIDIA Tesla A100 Ampere 40 GB Graphics Card - PCIe...
2 ratings

3 options
sorted by price + delivery: low to high
Filter

New
$7,988.00
FREE delivery **Wednesday, May 10**. Details
Add to Cart
Ships from: Global Tech Discount
Sold by: Global Tech Discount
(98 ratings)
88% positive over lifetime

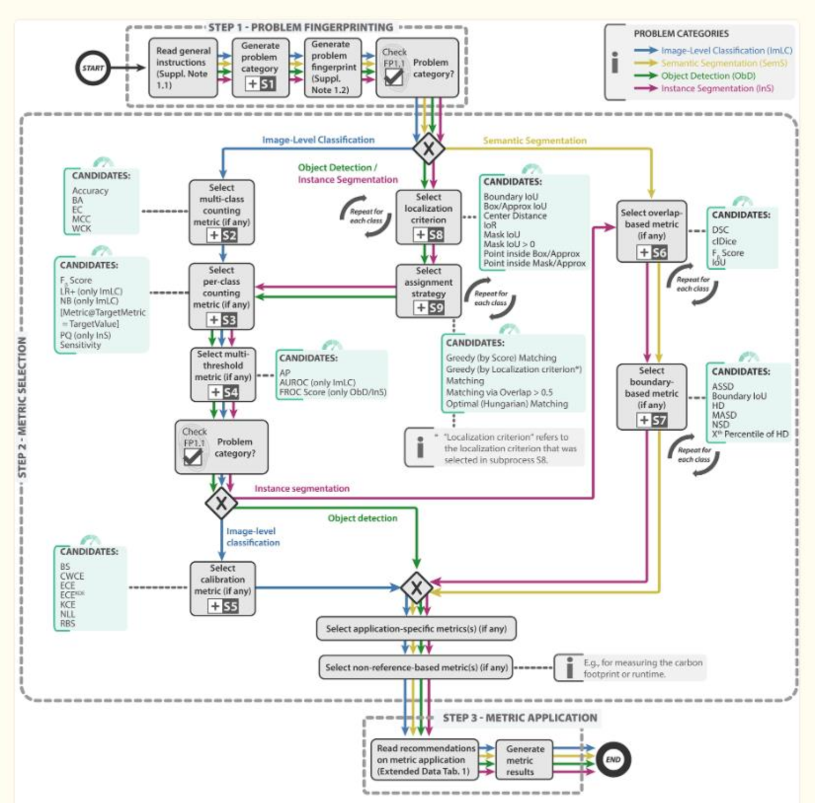
New
$8,009.00
FREE delivery **May 10 - 12**. Details
Add to Cart
Ships from: Professional Graphics Solutions
Sold by: Professional Graphics Solutions
(90 ratings)
94% positive over lifetime

New
$9,999.00
FREE delivery **May 11 - 15**. Details
... More
Add to Cart
Ships from: Aventis Systems
Sold by: Aventis Systems
(5594 ratings)
70% positive over last 12 months

Kirillov et al 2023

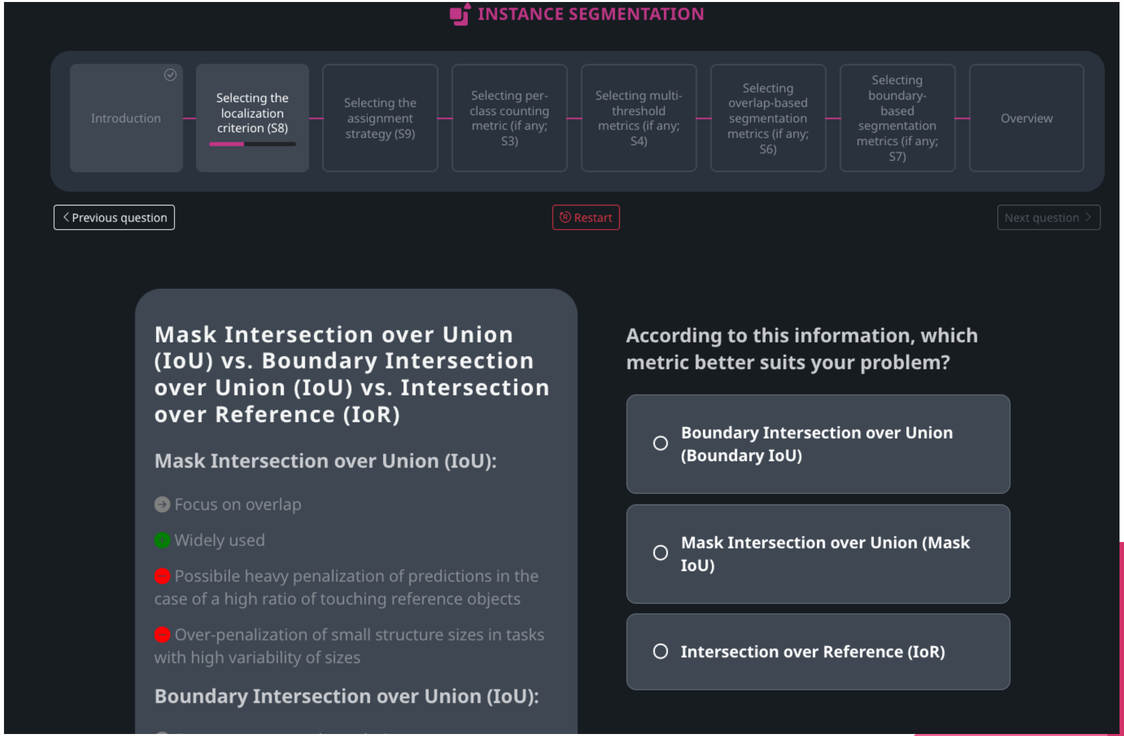
Think about… how will you actually determine if the model is working?

- What is the right way to assess accuracy? Which parts of assessing accuracy are fast enough to incorporate into your loss function? Can you train your model in parts?



Maier-Hein et al 2024 Nat Meth

Think about… how will you actually determine if the model is working?

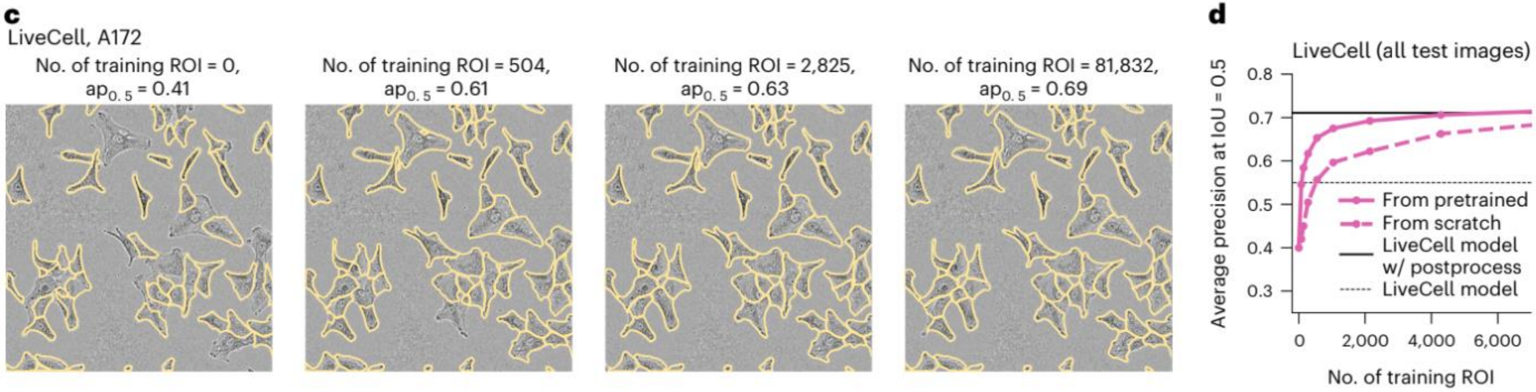


Maier-Hein et al 2024 Nat Meth; https://metrics-reloaded.dkfz.de/

Think about… where will I get my data?

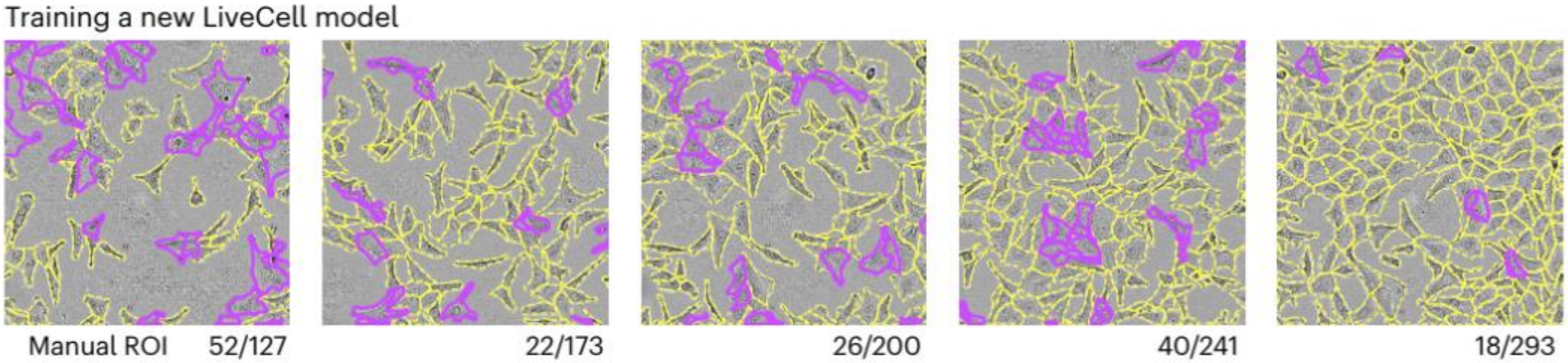
- Certain kinds of models (especially transformers) need a LOT of data
- Some sets:
  - CytoImageNet
  - Broad Bioimage Benchmark Collection
  - Cell Painting Gallery
  - Cell Tracking Challenge
  - Image Data Resource
  - Human Protein Atlas
  - Open Cell
  - Bioimage Archive
- But all still dwarfed by natural images (we never made cell Facebook, darn it)
  - Pretraining!

Think about… how can I make it easy to get more data if I need to?



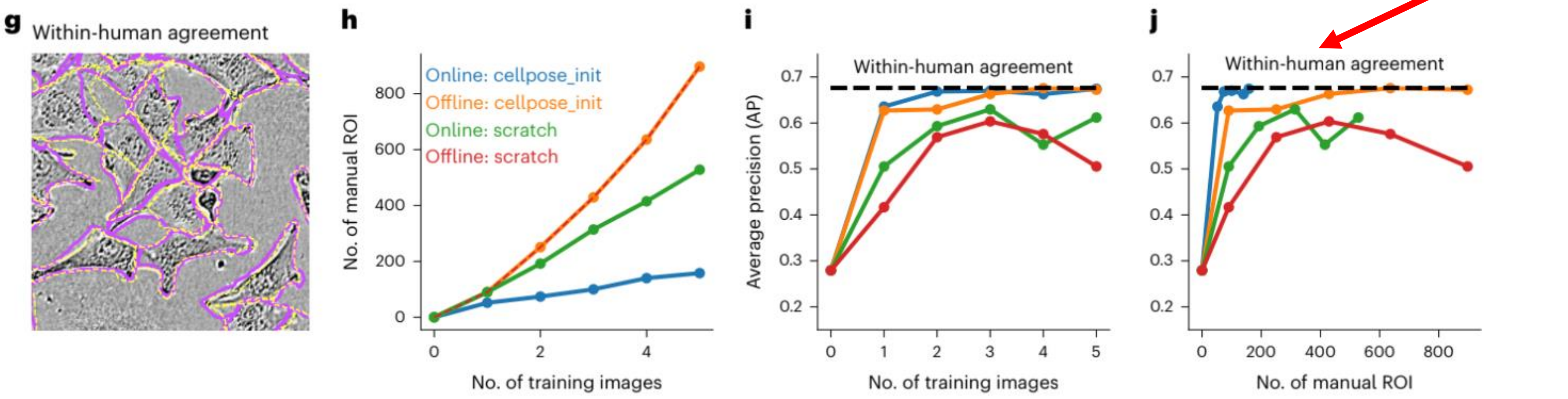
Pachitariu & Stringer, Nat Meth 2022

Think about… how can I make it easy to get more data if I need to?



Pachitariu & Stringer, Nat Meth 2022

Think about… how can I make it easy to get more data if I need to?



Pachitariu & Stringer, Nat Meth 2022

The world's best instance segmenter is still  - but maybe not for much longer!

BROAD INSTITUTE IMAGING PLATFORM

Cimini Lab members

Beth Cimini
Shatavisha Dasgupta
Nodar Gogoberidze
Kamal Hamid
Ksenija Krasina
Le Liu
Paula Llanos
Esteban Miglietta
Andréa Papaleo
Rajavi Shah
Suganya Sivagurunathan
Erin Weisbart

Carpenter-Singh Lab members

Anne Carpenter
Shantanu Singh
John Arevalo
Niranj Chandrasekaran
Jessica Ewald
Hugo Hakem
Alexander Kalinin
Sara Khosravi
Ksenija Krasina
Ank Kumar
Alán Muñoz González
Srijit Seal
Tim Treis

Use:

PIXIMI™
CellProfiler™ cell image analysis software
PyCytominer
CellProfiler Analyst™ data exploration software
Distributed Science

Read:

image.sc
Bioimagingguide.org

Join:

BioImaging North America
QUAREP-LiMi
SBI² Society of Biomolecular Imaging and Informatics
CytoData

Recent major funding for this work provided by:

- CZI Imaging Scientist Fellowship
- NIH NIGMS: MIRA R35 GM122547
- CZI Software Fellows program
- NIH NIGMS: P41 GM135019
- Massachusetts Life Sciences Center

broad.io/InstanceDL_2024

Many thanks to our funders and our many biology collaborators!!!!!!