Cellpose:

A generalist algorithm for cellular segmentation

Carsen Stringer , Michalis Michaelos,
Marius Pachitariu

INDEX

HYUNHP

INTRODUCTION

MODEL ARCHITECTURE

VECTOR FLOW REPRESENTATION

U-NET NEURAL NETWORK

RESULT

INTRODUCTION

INTRODUCTION

- The Cellpose 1.0 paper proposes a **deep learning-based image analysis method for cell segmentation and tracking.**

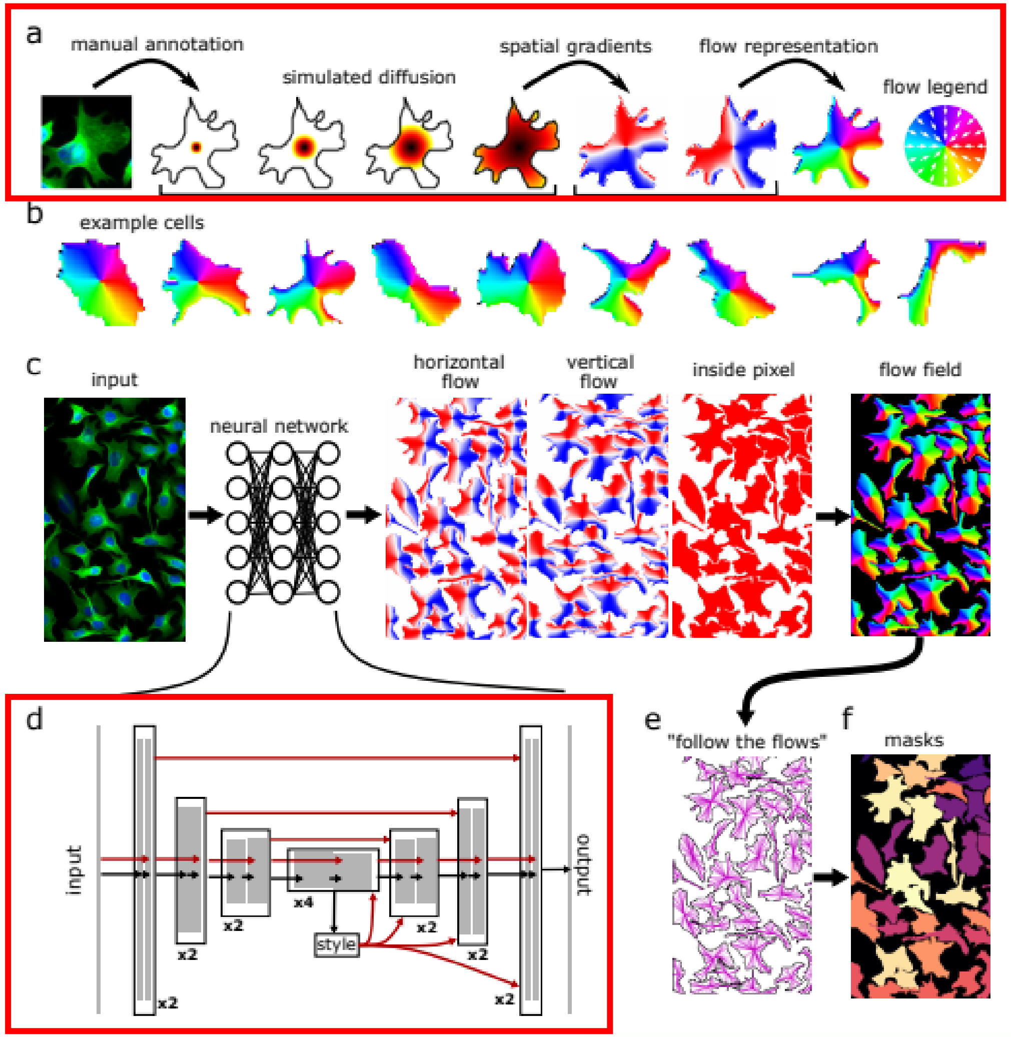
- **Traditional** image analysis methods require **manual parameter** tuning and often struggle with complex biological structures, but **Cellpose** offers an **automated solution** that can process diverse cell types and imaging modalities.

- The Cellpose model is based on a U-Net architecture with several enhancements for improved performance and versatility.

- Overall, the Cellpose 1.0 paper presents an important contribution to the field of cell biology and offers a promising new tool for automated image analysis.

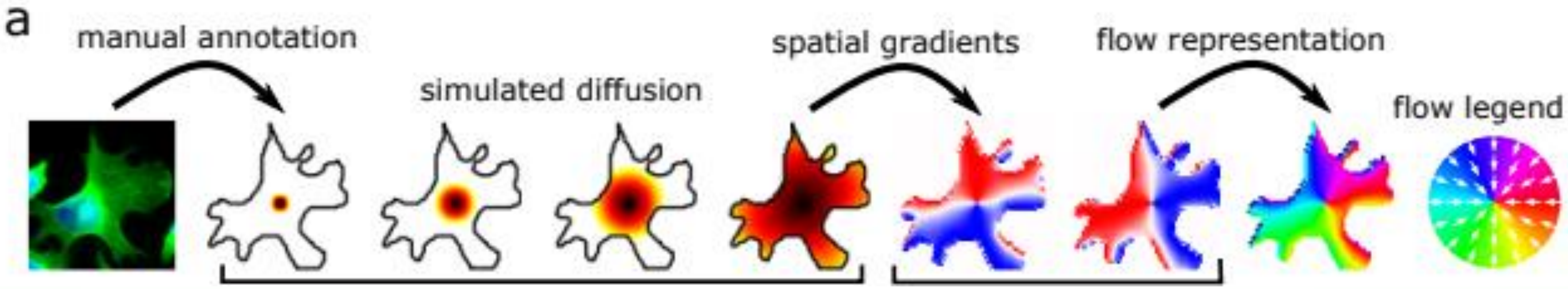
MODEL ARCHITECTURE

MODEL ARCHITECTURE



VECTOR FLOW REPRESENTATION

Procedure for Transforming Manually Annotated Masks into a Vector Flow Representation

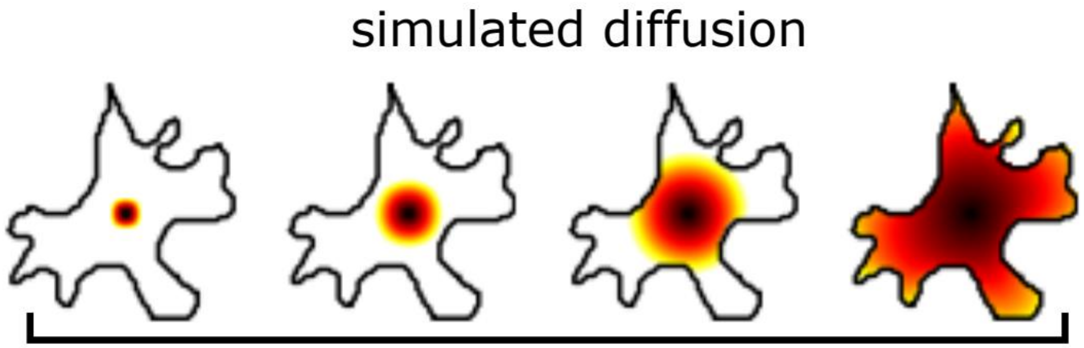


- The model architecture of Cellpose includes **a procedure for transforming manually annotated masks into a vector flow representation** that can be used as input to the neural network. This vector flow representation is derived **using a simulated diffusion process that starts at the center of the mask and spreads outwards.**

- During this diffusion process, **spatial gradients are computed at each pixel in the mask.** These gradients point towards the center of the cell, even around corners and edges. The X and Y components of these gradients are then combined into a **single normalized direction**, represented by an angle between 0 and 360 degrees.

- In other words, this procedure allows the model to capture the shape and orientation of the cell in a way that is robust to variations in the input image. By encoding the spatial gradients as a vector flow representation, the model can learn to segment cells with high accuracy, even in cases where traditional segmentation methods might fail.

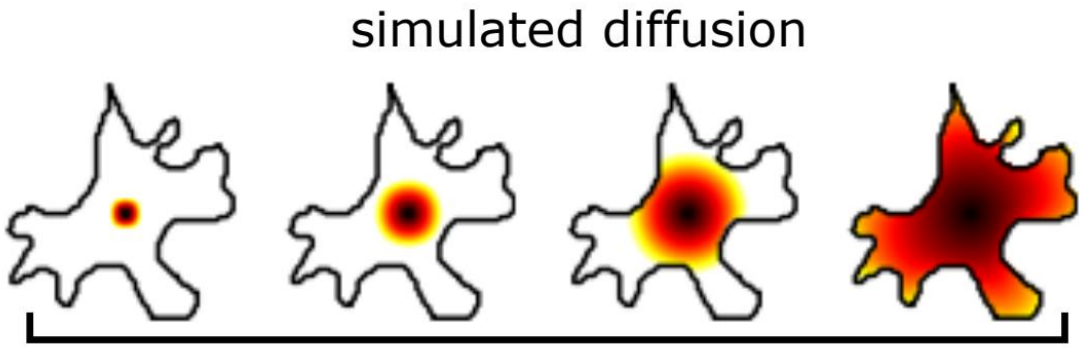
SIMULATED DIFFUSION (1)



- The simulated diffusion process used in Cellpose is based on the idea of computing a **distance transform of the manually annotated cell masks.** A distance transform is a technique that assigns a distance value to each pixel in the image based on **its proximity to the nearest boundary**. In this case, the boundary is the edge of the manually annotated mask.

- The distance transform is computed using a variant of the **Fast Marching Method**, which simulates the propagation of a wavefront through the image. **Starting at the center of the cell mask, the wavefront is propagated outward in all directions until it reaches the edge of the mask**. The distance transform is computed by measuring the distance of each pixel to the point where the wavefront passed through it.

SIMULATED DIFFUSION (2)



- Once the distance transform is computed, the spatial gradients are derived by computing the gradient of the distance transform at each pixel. These gradients point towards the center of the cell and are used to compute the normalized direction of the vector flow representation.

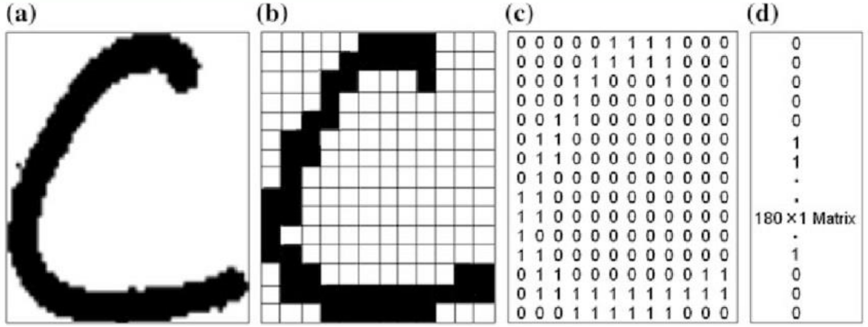
- The simulated diffusion process is a key part of the Cellpose model architecture, as it allows the model to **capture the shape and orientation of the cell in a way that is robust to variations in the input image.** By encoding the spatial gradients as a vector flow representation, the model can learn to segment cells with high accuracy, even in cases where traditional segmentation methods might fail.

Ref. Fast Marching Method (FMM)

- The Fast Marching Method is an algorithm for computing distance transforms of **binary images**. The basic idea of the method is to simulate the propagation of a wavefront through the image, starting from the set of pixels that are known to be on the boundary of a given region of interest.

- **The wavefront is initialized at the boundary pixels** and is propagated through the image by **updating the distance values of neighboring pixels based on the distance of the wavefront at the current location**. The update rule is designed in such a way that the distance values are always non-negative and decrease monotonically as the wavefront propagates through the image.

- The Fast Marching Method is widely used in image processing applications, such as medical image segmentation, shape analysis, and computer vision. **In the context of Cellpose, it is used to compute the distance transform of the manually annotated cell masks, which is then used to derive the spatial gradients for the vector flow representation.**

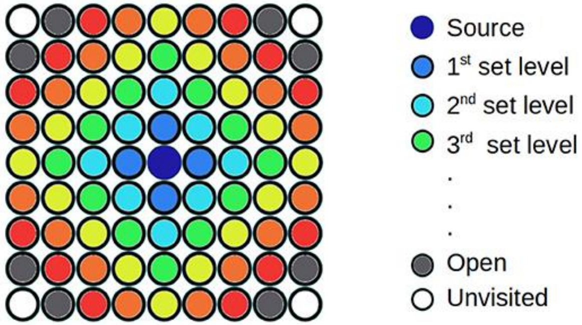


(c)

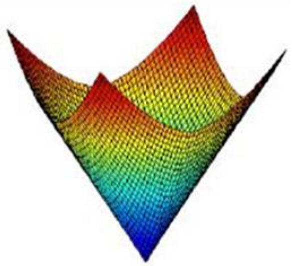
0	0	0	0	0	1	1	1	1	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0
0	0	0	1	1	0	0	0	1	0	0	0
0	0	0	1	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	1
0	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	0	0	0

(d)

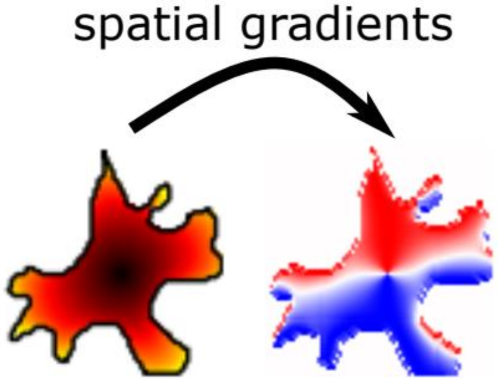
0
0
0
0
0
1
1
.
.
180 × 1 Matrix
.
1
0
0
0



● Source
● 1st set level
● 2nd set level
● 3rd set level
.
.
.
● Open
○ Unvisited



SPATIAL GRADIENTS (1)

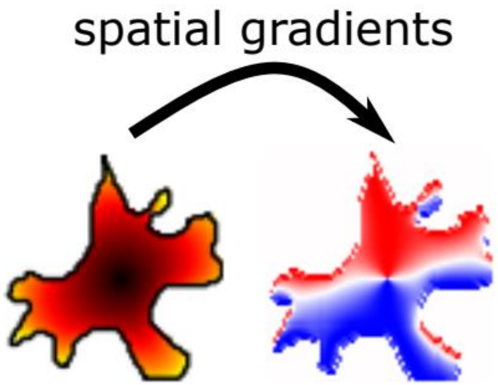


- FMM is used in the simulated diffusion part of the Cellpose algorithm to compute the distance transform of the manually annotated cell masks. **The spatial gradients are then derived from the distance transform and used to compute the vector flow representation.**

- Starting at the center of the cell mask, the wavefront is propagated outward in all directions until it reaches the edge of the mask. The distance transform is computed by measuring the distance of each pixel to the point where the wavefront passed through it.

- Once the distance transform is computed, the spatial gradients are derived by computing the gradient of the distance transform at each pixel. **These gradients point towards the center of the cell and are used to compute the normalized direction of the vector flow representation.**

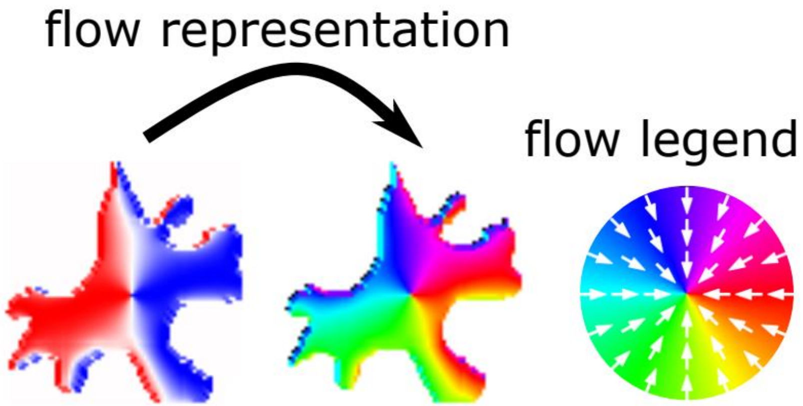
SPATIAL GRADIENTS (2)



- The spatial gradients are a key part of the Cellpose model architecture, as they **provide information about the orientation and shape of the cell that is used to improve the accuracy of the segmentation.** By encoding the spatial gradients as a vector flow representation, the model can learn to segment cells with high accuracy, even in cases where traditional segmentation methods might fail.

- In summary, the spatial gradients in Cellpose refer to **the direction and magnitude of change in the distance transform of the manually annotated cell masks.** They are used to compute the vector flow representation, which is a key component of the Cellpose model architecture.

Flow Representation

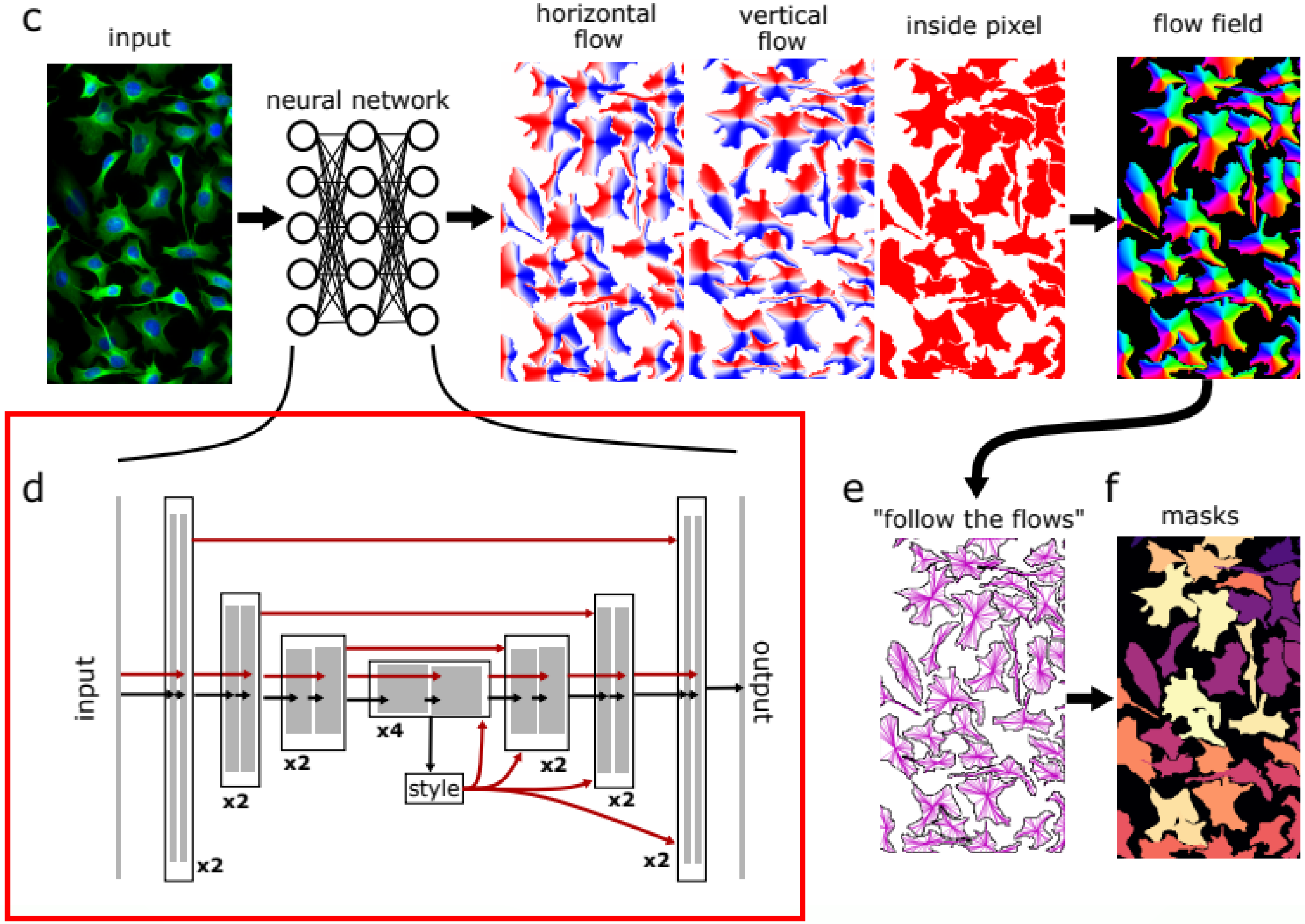


- Flow representation **is a vector field that encodes the normalized direction of flow towards the center of each cell.** The flow representation is derived from the spatial gradients, which are computed from the distance transform of the manually annotated cell masks using a variant of the Fast Marching Method.

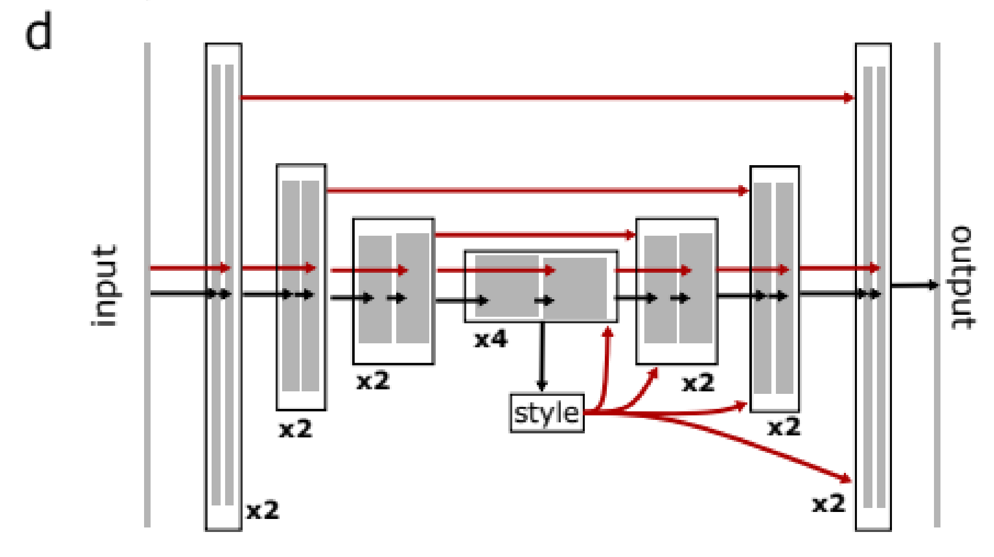
- To compute the flow representation, the spatial gradients are combined into a single normalized direction from 0◦ to 360◦, representing the direction of flow towards the center of the cell. This normalized direction is then used to compute a flow vector at each pixel in the image, which represents **the direction and magnitude of flow towards the center of the nearest cell.**

U-NET NEURAL NETWORK

U-NET Neural Network



U-NET Neural Network



1. **Input layer:** The input layer of the Cellpose model takes as input a grayscale microscopy image.

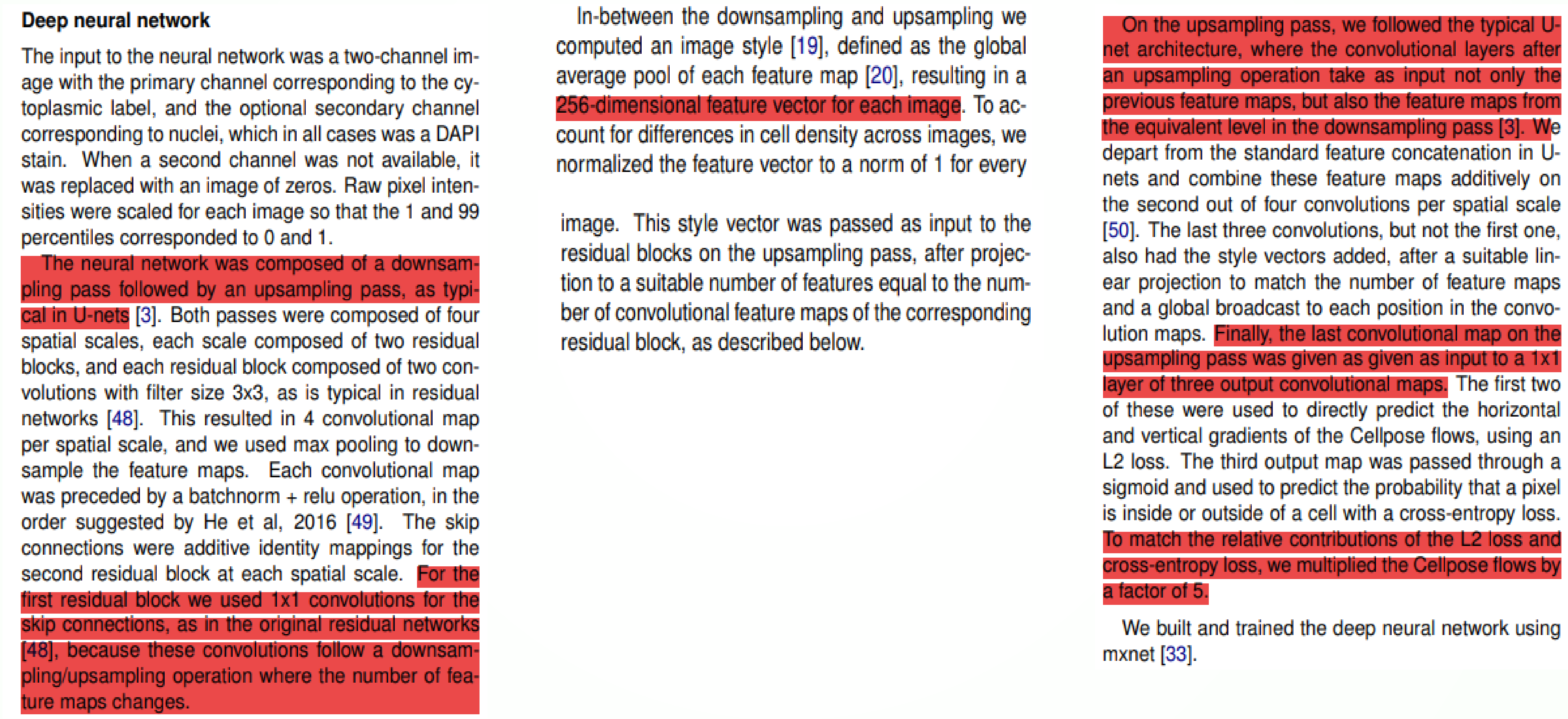
2. **Encoding layers:** The encoding layers of the model are organized in a U-shaped architecture, with multiple convolutional layers and max-pooling layers used to gradually reduce the spatial dimensions of the image while increasing the number of feature maps.

3. **Bottleneck layer:** The bottleneck layer of the model is a narrow layer with high feature map density that captures the most important features of the image.

4. **Decoding layers:** The decoding layers of the model are organized in a U-shaped architecture that is similar to the encoding layers, but in reverse order. These layers use transposed convolutional layers and upsampling layers to gradually increase the spatial dimensions of the feature maps while decreasing the number of feature maps.

5. **Output layer:** The output layer of the model is a sigmoid activation layer that generates a probability map for each pixel in the input image, indicating the likelihood of that pixel belonging to a cell.

U-NET ARCHITECTUE



Deep neural network

The input to the neural network was a two-channel image with the primary channel corresponding to the cytoplasmic label, and the optional secondary channel corresponding to nuclei, which in all cases was a DAPI stain. When a second channel was not available, it was replaced with an image of zeros. Raw pixel intensities were scaled for each image so that the 1 and 99 percentiles corresponded to 0 and 1.

The neural network was composed of a downsampling pass followed by an upsampling pass, as typical in U-nets [3]. Both passes were composed of four spatial scales, each scale composed of two residual blocks, and each residual block composed of two convolutions with filter size 3x3, as is typical in residual networks [48]. This resulted in 4 convolutional map per spatial scale, and we used max pooling to downsample the feature maps. Each convolutional map was preceded by a batchnorm + relu operation, in the order suggested by He et al, 2016 [49]. The skip connections were additive identity mappings for the second residual block at each spatial scale. For the first residual block we used 1x1 convolutions for the skip connections, as in the original residual networks [48], because these convolutions follow a downsampling/upsampling operation where the number of feature maps changes.

In-between the downsampling and upsampling we computed an image style [19], defined as the global average pool of each feature map [20], resulting in a 256-dimensional feature vector for each image. To account for differences in cell density across images, we normalized the feature vector to a norm of 1 for every image. This style vector was passed as input to the residual blocks on the upsampling pass, after projection to a suitable number of features equal to the number of convolutional feature maps of the corresponding residual block, as described below.

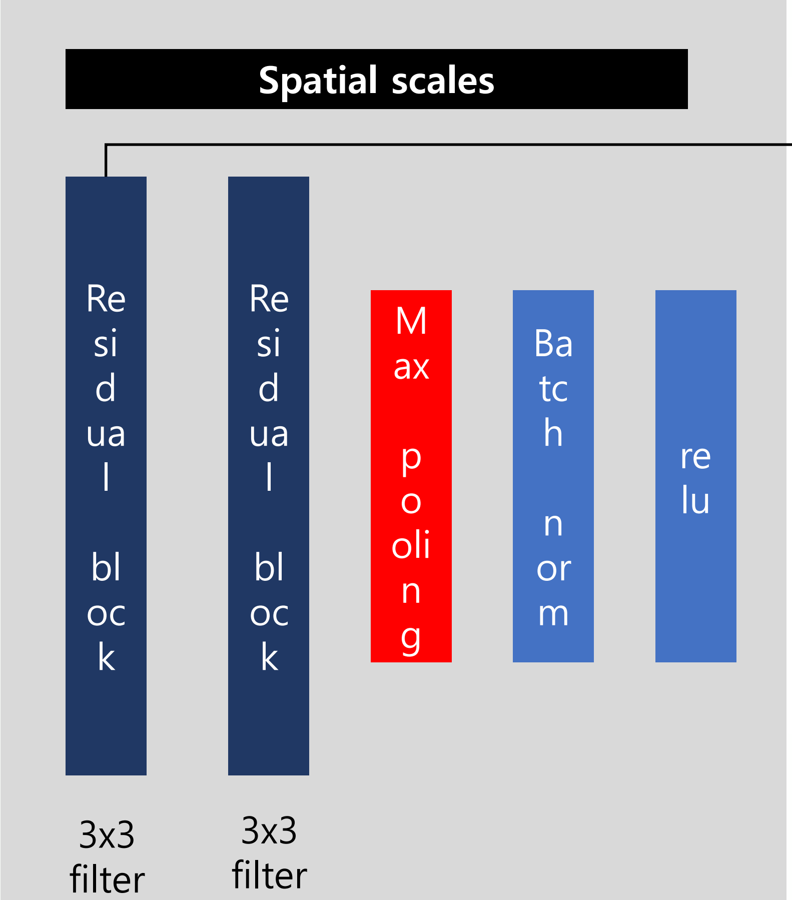
On the upsampling pass, we followed the typical U-net architecture, where the convolutional layers after an upsampling operation take as input not only the previous feature maps, but also the feature maps from the equivalent level in the downsampling pass [3]. We depart from the standard feature concatenation in U-nets and combine these feature maps additively on the second out of four convolutions per spatial scale [50]. The last three convolutions, but not the first one, also had the style vectors added, after a suitable linear projection to match the number of feature maps and a global broadcast to each position in the convolution maps. Finally, the last convolutional map on the upsampling pass was given as given as input to a 1x1 layer of three output convolutional maps. The first two of these were used to directly predict the horizontal and vertical gradients of the Cellpose flows, using an L2 loss. The third output map was passed through a sigmoid and used to predict the probability that a pixel is inside or outside of a cell with a cross-entropy loss. To match the relative contributions of the L2 loss and cross-entropy loss, we multiplied the Cellpose flows by a factor of 5.

We built and trained the deep neural network using mxnet [33].

Downsampling (Paper described)

INPUT

224 * 224 images



Spatial scales

Residual block

3x3 filter

Residual block

3x3 filter

Max pooling

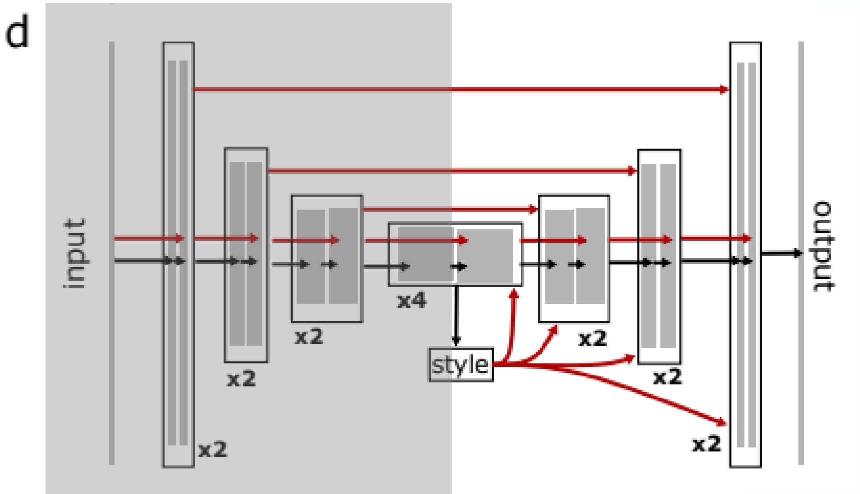
Batch norm

relu

For skip connections

1 x 1 convolution feature map

X 4

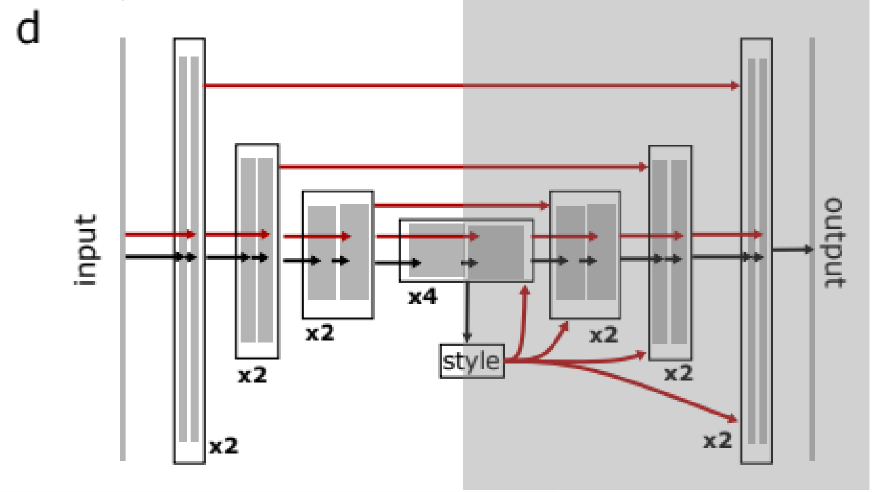


If no padding and no stride, then size only will be squeezed in the maxpooling layer

Upsampling (Paper described)

For skip connections

1 x 1 convolution feature map



Spatial scales

Residual block

Residual block

Batch norm

relu

3x3 filter

3x3 filter

X 4

Ref. U-NET Architecture Estimation

CASE 1. ZERO PADDING, STIRDE = 1

LAYER	DOWN SAMPLING				UP SAMPLING			
	1	2	3	4	4	3	2	1
INPUT	224	112	56	28	14	28	56	112
FILTER	3	3	3	3	3	3	3	3
PADDING	1	1	1	1	1	1	1	1
STRIDE	1	1	1	1	1	1	1	1
OUTPUT	224	112	56	28	14	28	56	112
FINAL RESULT	112	56	28	14	28	56	112	224

CASE 2. NO PADDING, STIRDE = 1

LAYER	DOWN SAMPLING				UP SAMPLING			
	1	2	3	4	4	3	2	1
INPUT	224	111	55	27	13	24	49	102
FILTER	3	3	3	3	3	3	3	3
PADDING	0	0	0	0	0	0	0	0
STRIDE	1	1	1	1	1	1	1	1
OUTPUT	222	109	53	25	11	22	47	100
FINAL RESULT	111	55	27	13	24	49	102	211

APPENDIX. STYLE TRANSFER

- In computer vision, style transfer refers to the process of transferring the style or visual appearance of one image to another image, while preserving the content of the latter. This technique is typically achieved using deep neural networks, specifically convolutional neural networks (CNNs).

- The general idea behind style transfer is to use a pre-trained CNN to extract the style information from a "style" image, and then apply that style to a "content" image while preserving the content of the latter. The CNN is usually trained on a large dataset of images, such as ImageNet, and can extract feature maps that capture the visual style of an image, such as its colors, textures, and patterns.

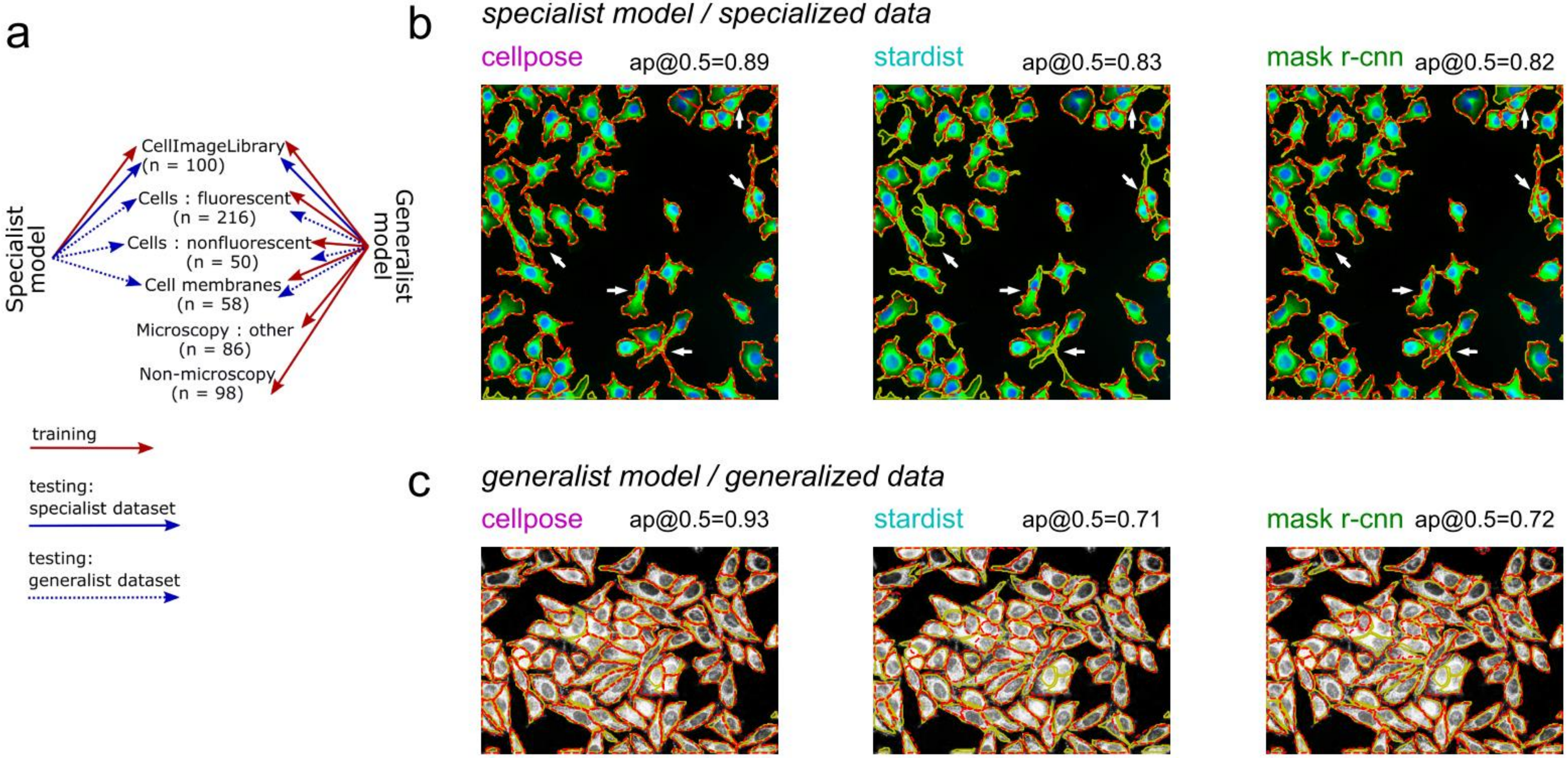
- To perform style transfer, the feature maps of the style image are used to define a Gram matrix, which captures the correlations between the different feature maps. This Gram matrix is then used to transform the feature maps of the content image, so that they match the correlations of the style image. This process is usually done by optimizing a loss function that balances the content and style information.

- The resulting image has the content of the original image, but with the visual style of the style image. This technique has a wide range of applications, including artistic stylization, image enhancement, and image synthesis.

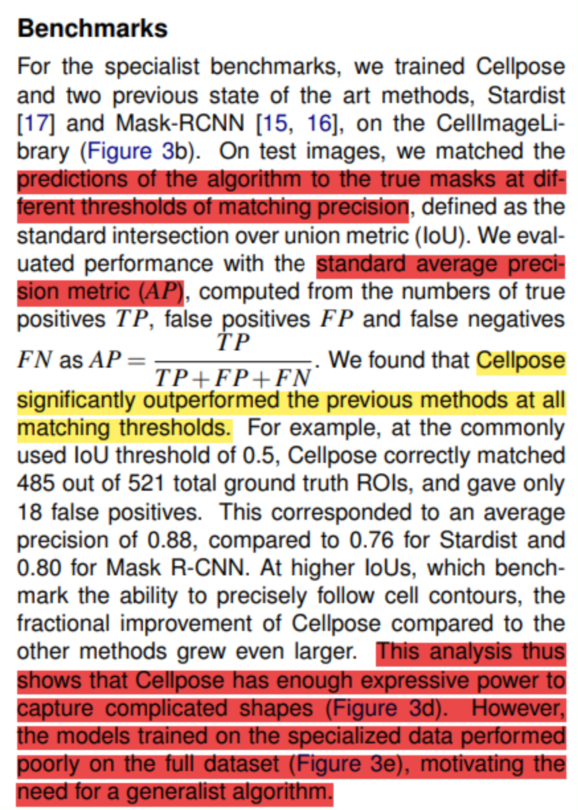
- **In summary, style transfer in computer vision involves using a pre-trained CNN to extract the style information from a style image, and then applying that style to a content image while preserving the content. The style information is typically represented as a set of feature maps and is used to transform the feature maps of the content image using a Gram matrix.**

RESULT

TRAINING METHODS

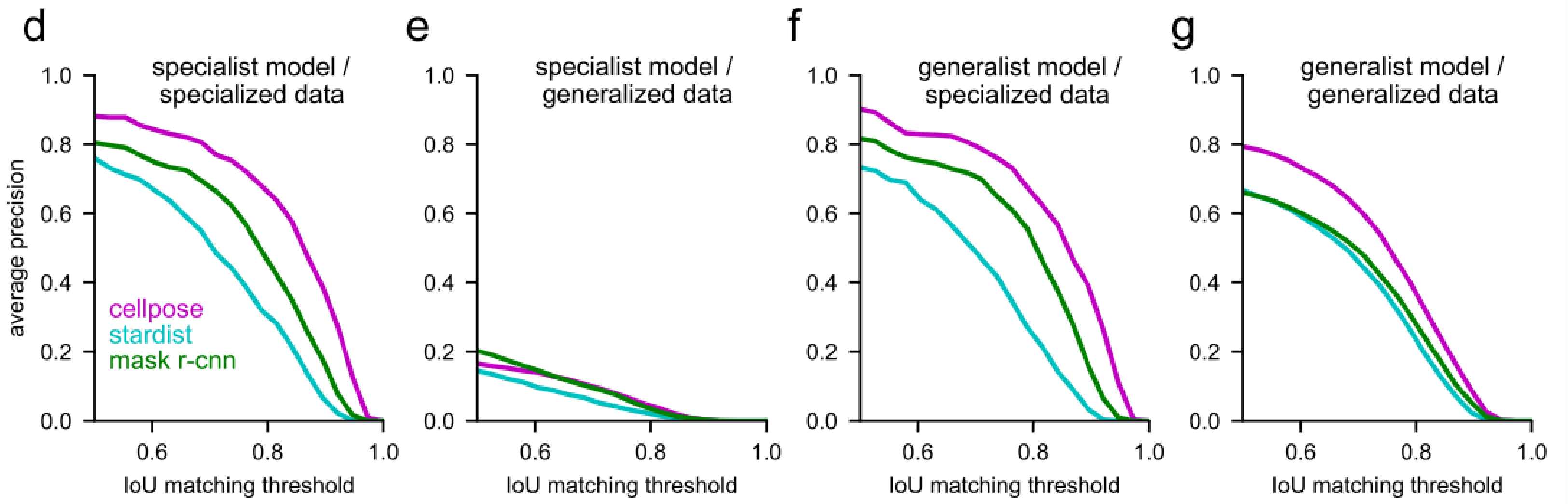

a
Specialist model
Generalist model
CellImageLibrary (n = 100)
Cells : fluorescent (n = 216)
Cells : nonfluorescent (n = 50)
Cell membranes (n = 58)
Microscopy : other (n = 86)
Non-microscopy (n = 98)
training
testing: specialist dataset
testing: generalist dataset
b
specialist model / specialized data
cellpose ap@0.5=0.89
stardist ap@0.5=0.83
mask r-cnn ap@0.5=0.82
c
generalist model / generalized data
cellpose ap@0.5=0.93
stardist ap@0.5=0.71
mask r-cnn ap@0.5=0.72

MODEL RESULT



Benchmarks

For the specialist benchmarks, we trained Cellpose and two previous state of the art methods, Stardist [17] and Mask-RCNN [15, 16], on the CellImageLibrary (Figure 3b). On test images, we matched the predictions of the algorithm to the true masks at different thresholds of matching precision, defined as the standard intersection over union metric (IoU). We evaluated performance with the standard average precision metric (AP), computed from the numbers of true positives TP, false positives FP and false negatives FN as $AP = \frac{TP}{TP+FP+FN}$. We found that Cellpose significantly outperformed the previous methods at all matching thresholds. For example, at the commonly used IoU threshold of 0.5, Cellpose correctly matched 485 out of 521 total ground truth ROIs, and gave only 18 false positives. This corresponded to an average precision of 0.88, compared to 0.76 for Stardist and 0.80 for Mask R-CNN. At higher IoUs, which benchmark the ability to precisely follow cell contours, the fractional improvement of Cellpose compared to the other methods grew even larger. This analysis thus shows that Cellpose has enough expressive power to capture complicated shapes (Figure 3d). However, the models trained on the specialized data performed poorly on the full dataset (Figure 3e), motivating the need for a generalist algorithm.



Specialized data
This dataset consisted of 100 fluorescent images of cultured neurons with cytoplasmic and nuclear stains obtained from the CellImageLibrary
Generalized data
The full dataset included the 100 images described above, as well as 516 additional images from various source. All these extra images were fully manually segmented by a single human operator (MM).

E.O.D.