Deep Learning for Image Segmentation

Karl Rohr Janis Meyer Kerem Celikay Moritz Kunzmann

Biomedical Computer Vision Group (BMCV)
BioQuant, IPMB, Heidelberg University

Deep Learning for Image Segmentation: Introduction and Overview

Karl Rohr

Biomedical Computer Vision Group (BMCV)
BioQuant, IPMB, Heidelberg University

Deep Learning for Image Segmentation: Content

- **Introduction and overview** (Karl Rohr)
  - What is segmentation, classical methods
  - Convolution, CNNs, DL for image segmentation

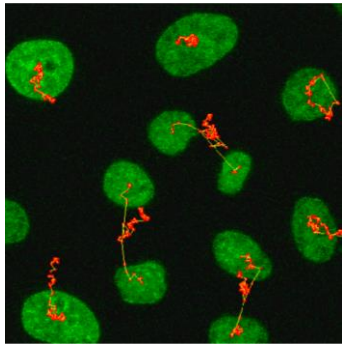
- **CNNs** (Janis Meyer)
  - Building blocks of CNNs, batch normalization
  - Loss functions, dropout, dilated convolution

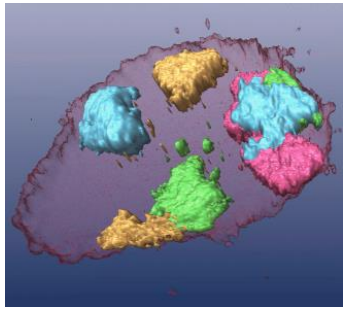
- **U-Net** (Kerem Celikay)
  - Network architecture, windowing
  - Data augmentation, performance metrics

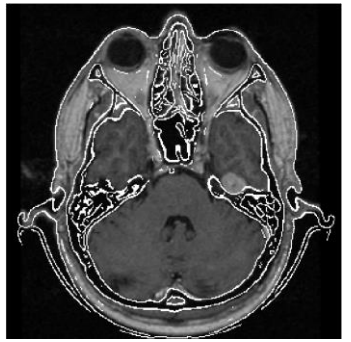
- **Cellpose** (Moritz Kunzmann)
  - Network architecture, image representation
  - Instance segmentation, training, datasets

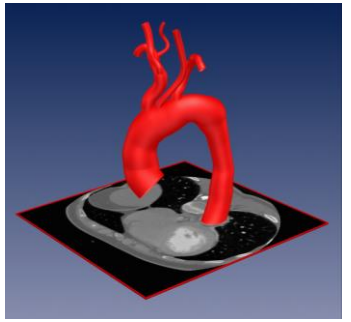
Biomedical Computer Vision Group (BMCV)

(Head: Karl Rohr)









Computer-based analysis of biological and medical images

Biomedical image analysis

Cell segmentation, cell tracking, particle tracking, classification, image registration, vessel segmentation, landmark localization

Methods

Deep learning, model-based methods, probabilistic methods

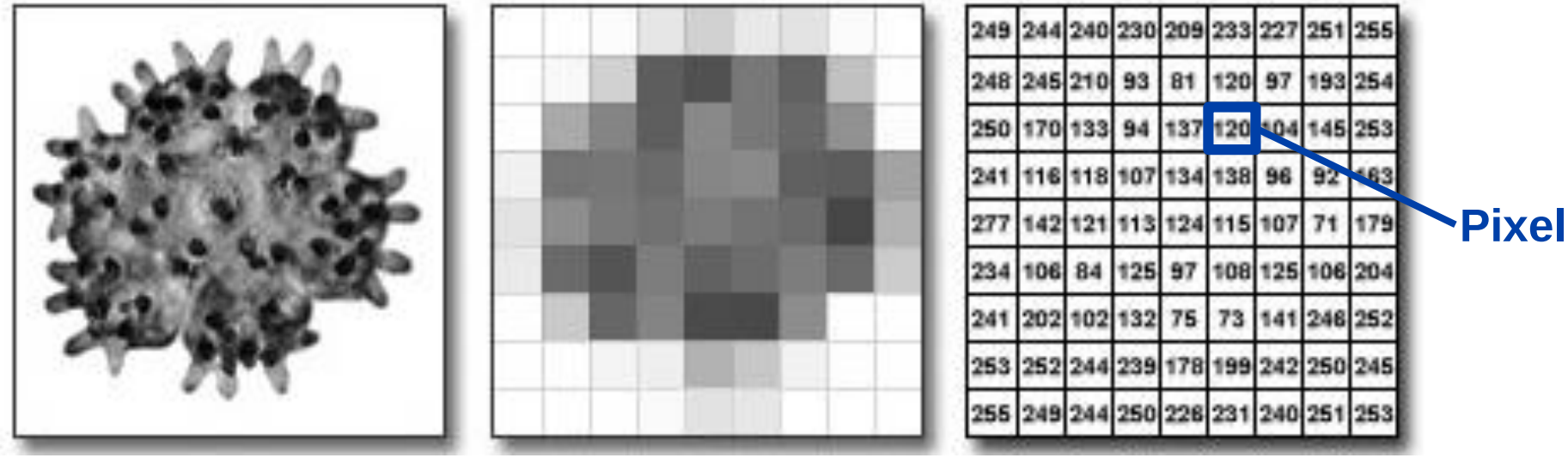
Applications

Cell migration, cell division, particle motion, virus infection, genome architecture, neurology, vascular diseases

www.bioquant.uni-heidelberg.de/bmcv

Image Segmentation

Digital Image



Analog image

Sampling
(space positions)

Quantization
(intensity values)

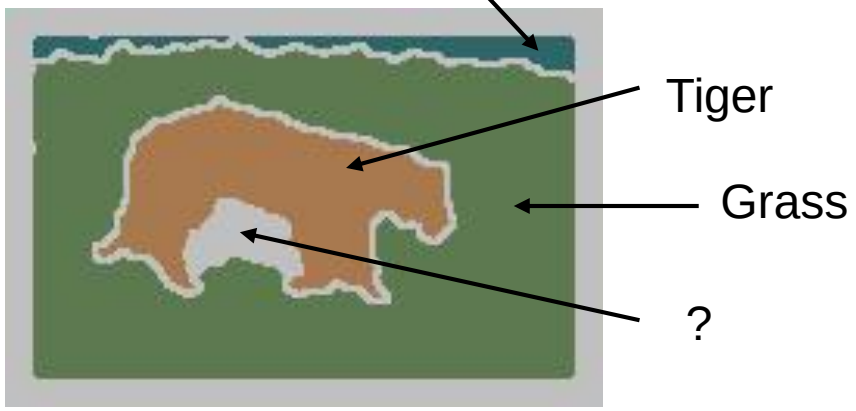
Digital image: **Sampled** (discrete positions) and **quantized** (discrete intensity values)

Pixel: Picture element, 2D, position (x,y) and intensity value $I(x,y)$

Molecular Expressions

Image Segmentation

Original image



Segmented image

Image segmentation

Partitioning *an image into a* ***set of regions*** *that cover it.*
The regions should represent meaningful areas of the image.

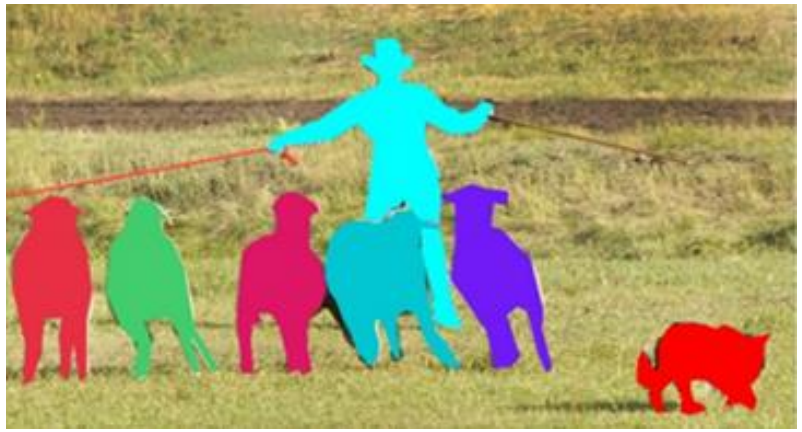
Assign a **label** to **each pixel** of an image (each image point)

Shapiro/Stockman 2001, J. Malik

Semantic Segmentation vs. Instance Segmentation

Original image

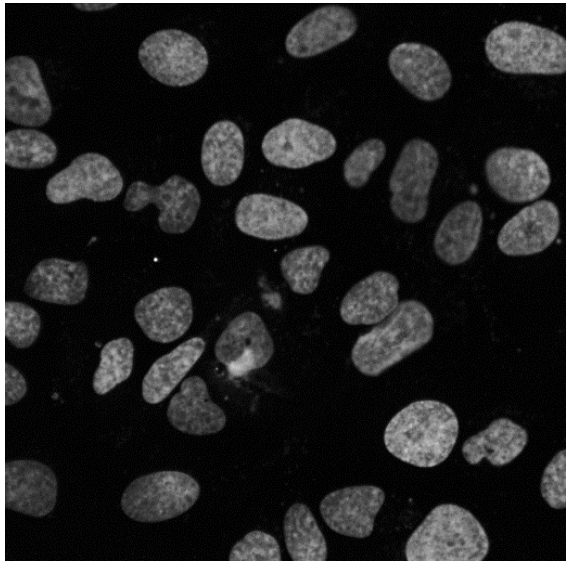
Semantic segmentation
(assign pixel labels for object classes)



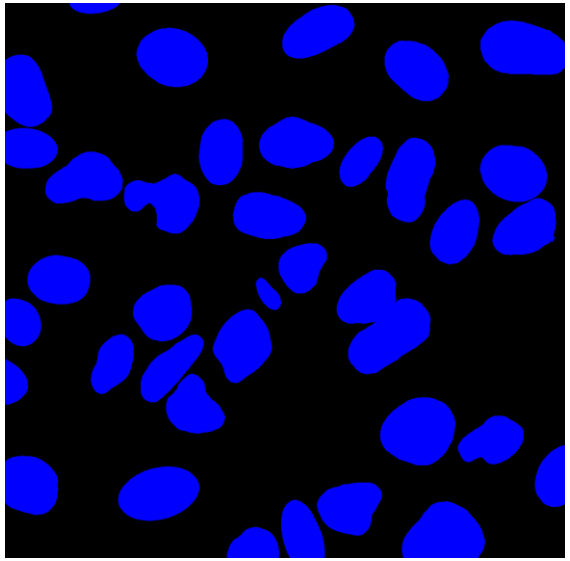
Instance segmentation
(assign pixel labels for instances of object classes)

Lin 2015

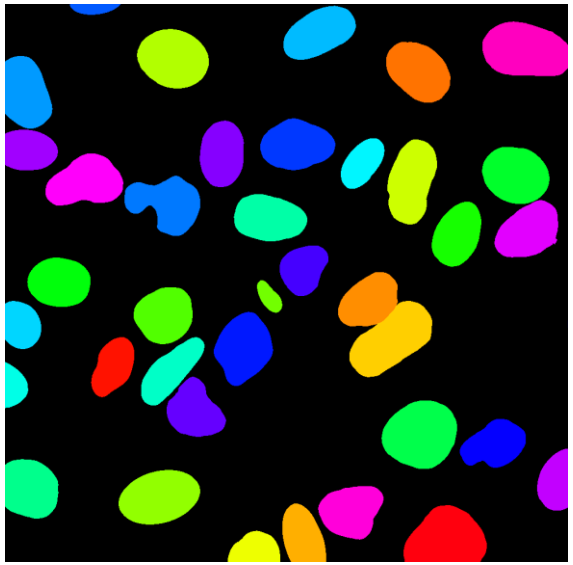
Segmentation of Cell Microscopy Images



Original image of cells



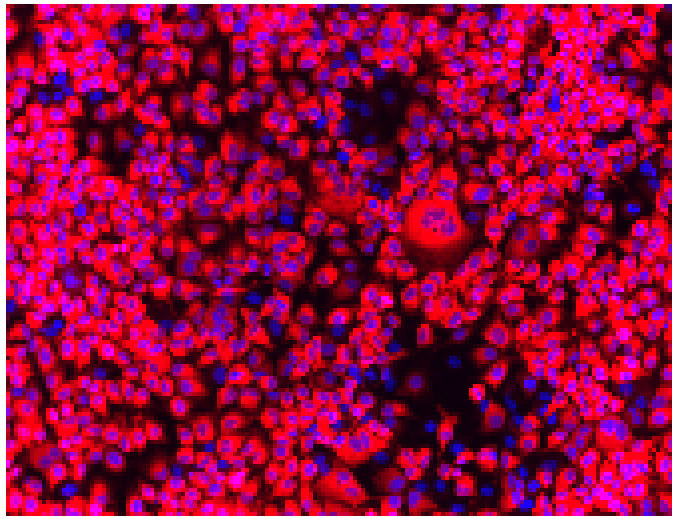
Semantic segmentation



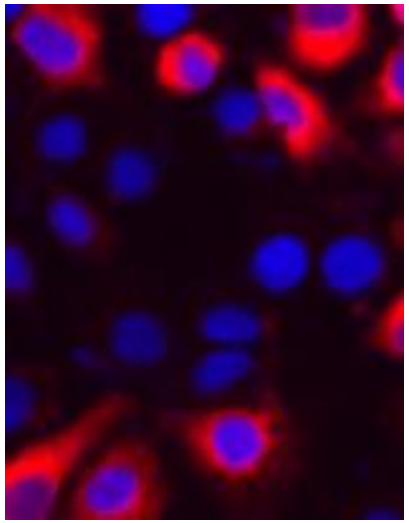
Instance segmentation

Kostrykin/Rohr 2023

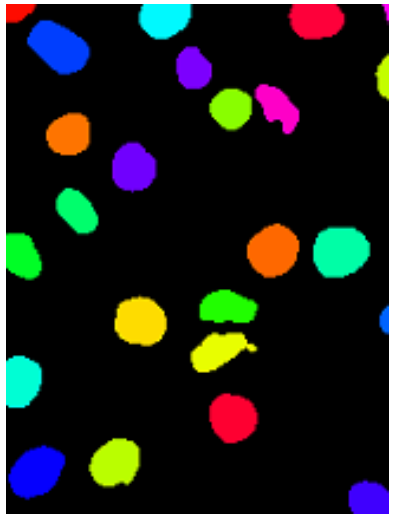
Application: Segmentation of Cell Microscopy Images



Original image
Virus infected cells



Original image



Cell nuclei (ch1)

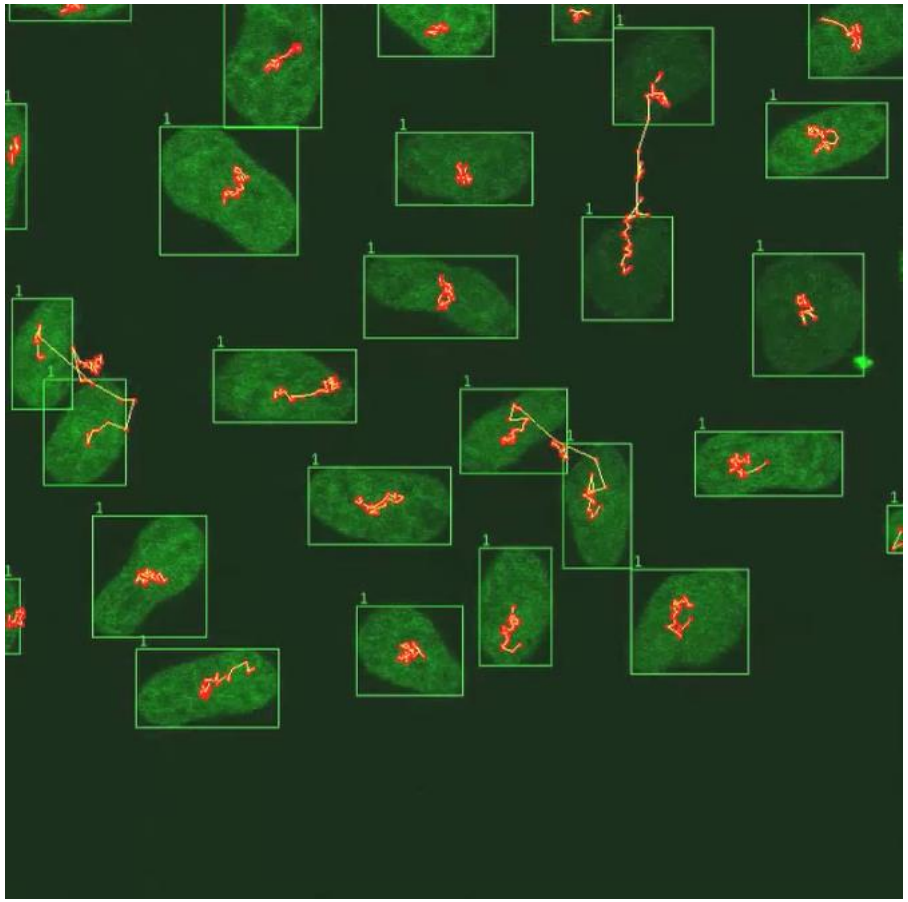
Cytoplasm (ch2)

Automatic analysis of high-throughput RNAi knock-down screens

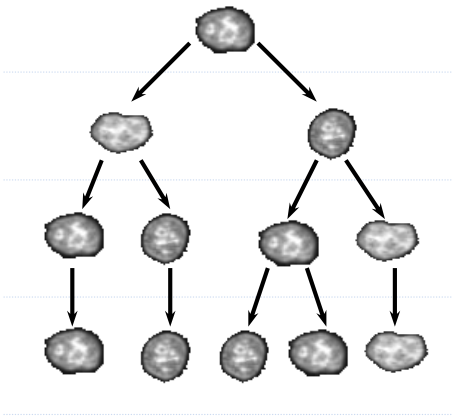
- Segmentation of cell nuclei and whole cells
- Quantify fluorescence intensity (infection level) of cells to identify relevant genes

Matula/Rohr et al. 2008

Application: Segmentation and Tracking of Cells



Segmentation and tracking of cells:
Object trajectories

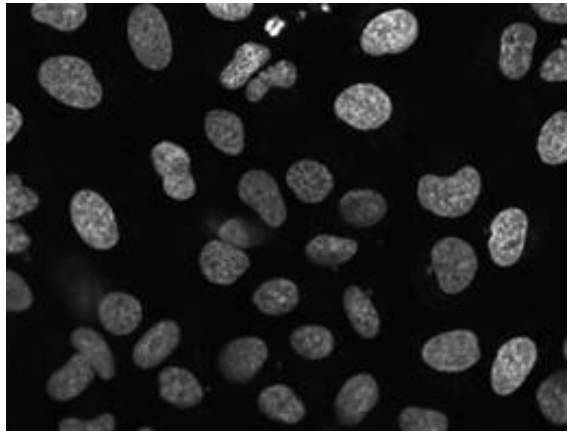
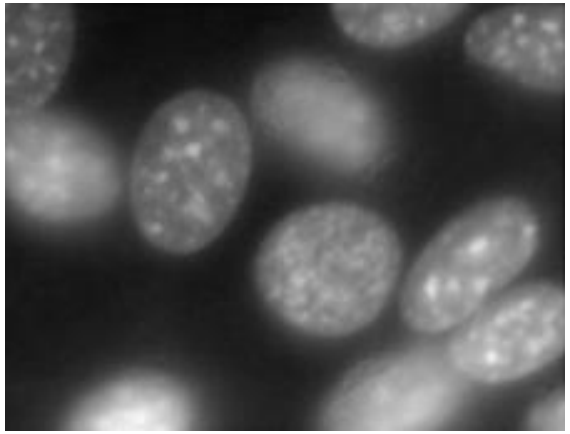
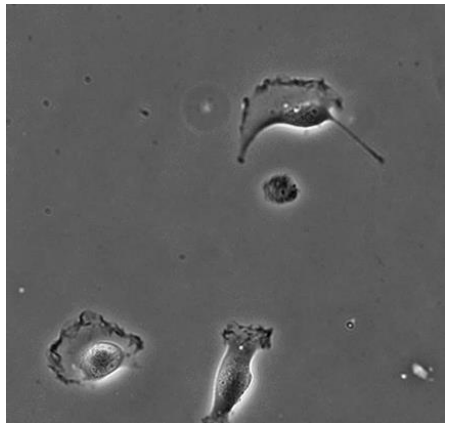
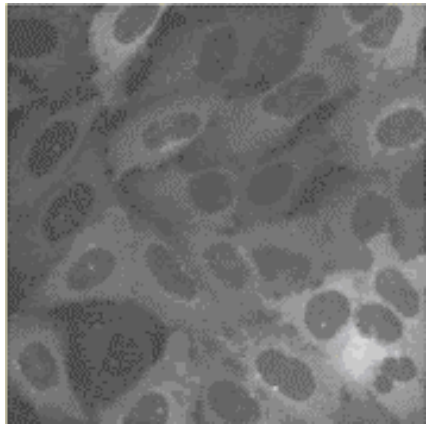


Cell lineage tree

Harder/Rohr et al. 2009, 2015

Challenges of Cell Segmentation

- High variation of shape and image intensity
- Inhomogeneous image intensities
- Strong image noise, image blur
- Low image contrast (e.g., object borders)
- High object density, touching/overlapping objects
- High number of objects

Threshold-Based Image Segmentation

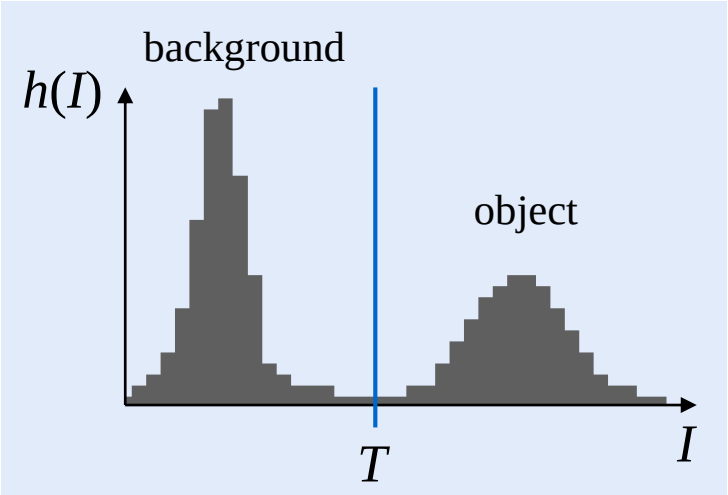
Assumptions

- Objects have roughly constant intensities $g(x,y) \approx const$
- Intensities of objects and background differ strongly

$$g_T(x,y) = \begin{cases} 0 & g(x,y) < T \quad \text{background} \\ 1 & g(x,y) \geq T \quad \text{object} \end{cases}$$

Selection of threshold T (e.g., use of histogram)

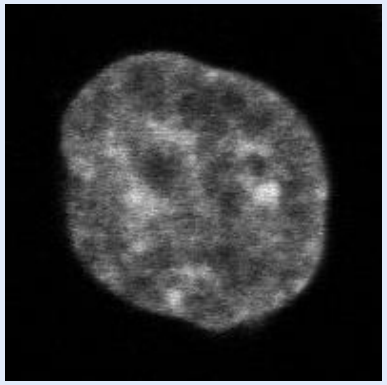
- Manual
- Automatic (e.g., Otsu's method)



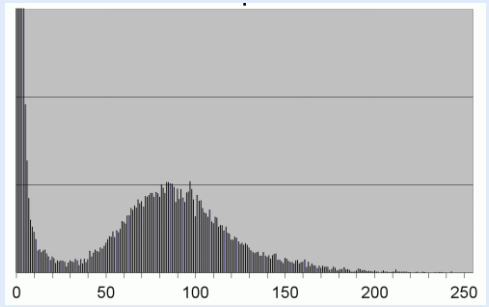
Histogram of intensity values I of an image $g(x,y)$

Histogram of an Image

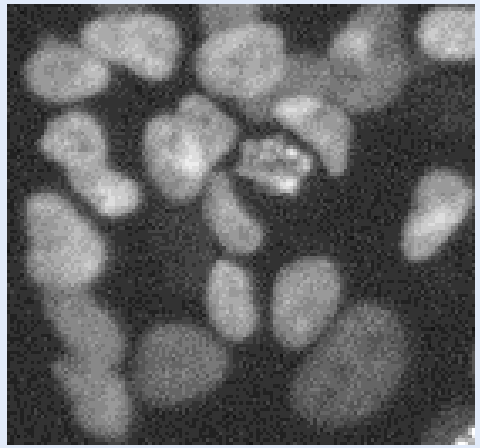
Represents how often the intensity values appear in an image



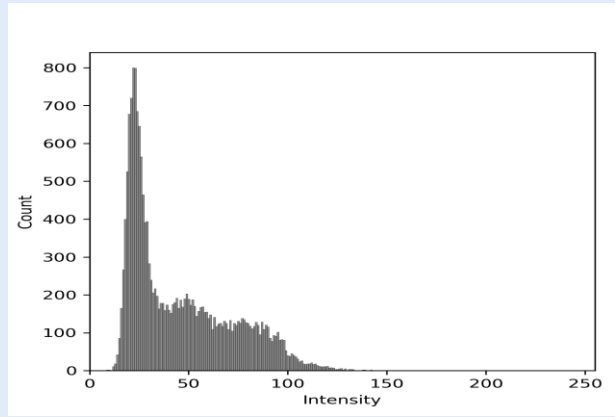
Original image
(one cell)



Histogram



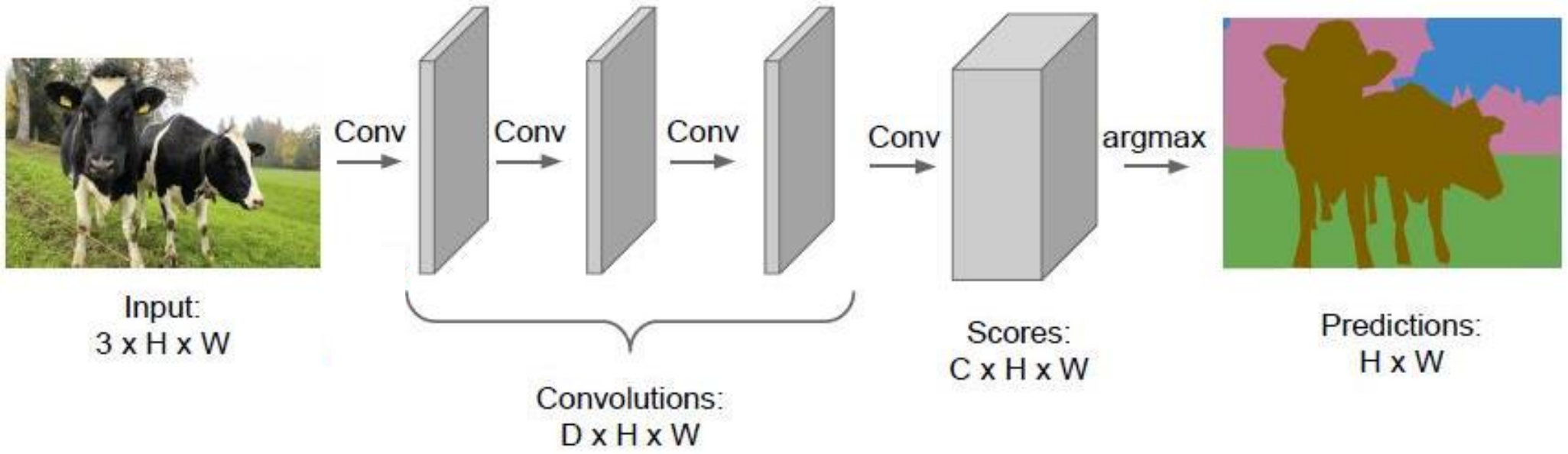
Original image
(several cells)



Histogram

Deep Learning for Image Segmentation: Convolutional Neural Networks (CNNs)

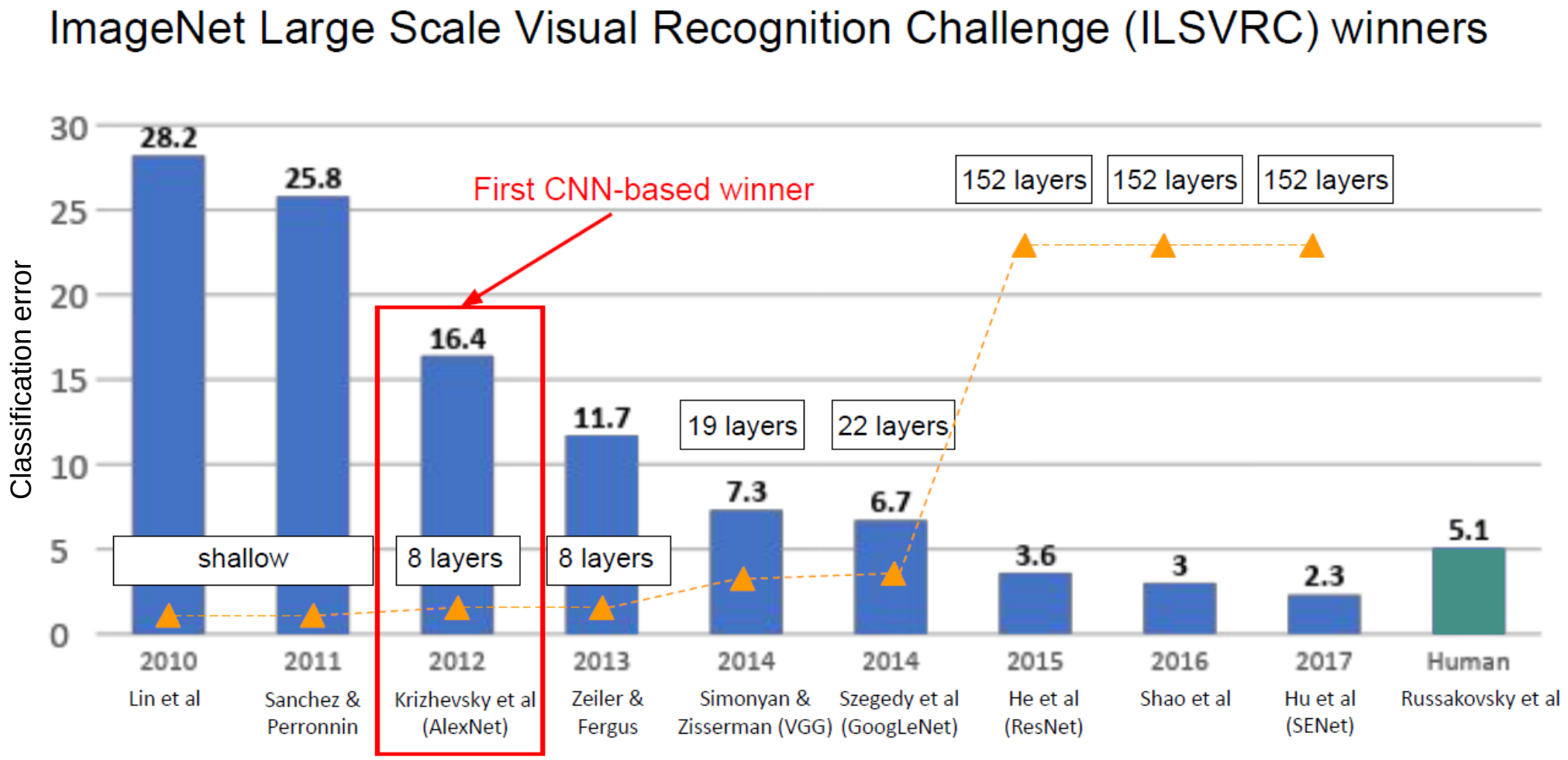
Convolutional Neural Network (CNN) for Image Segmentation



- CNN consists of several convolution layers
- Predicts labels for all pixels of an image

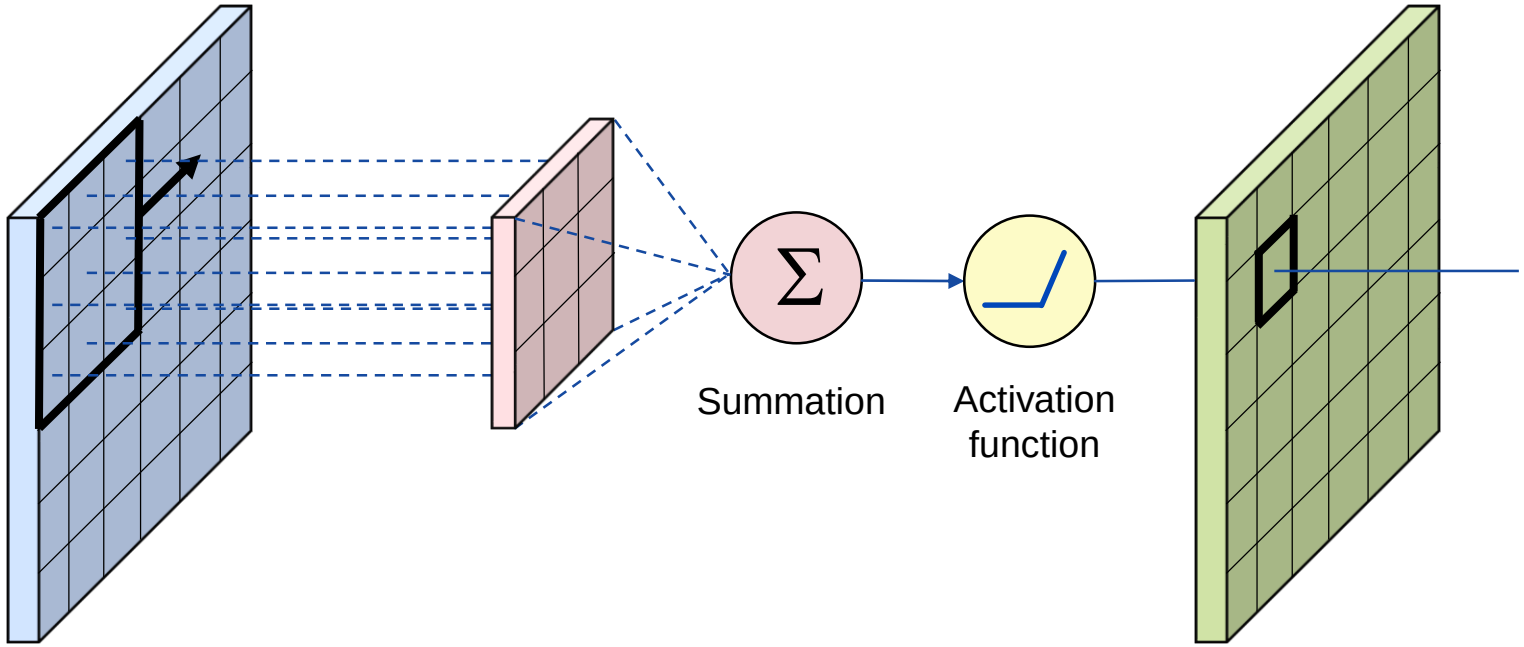
Li et al. 2023, cs231n

CNNs in Object Recognition Challenge



Li et al. 2023, cs231n

Convolution Layer



Summation

Activation function

$$y = \sigma\left(\sum_i w_i\, x_i + b\right)$$

x_i Input
y Output
w_i Filter weights
b Bias
σ Activation function

Input layer
(e.g., image)

Filter

Output layer
(feature map, activation map)

Convolution Layer

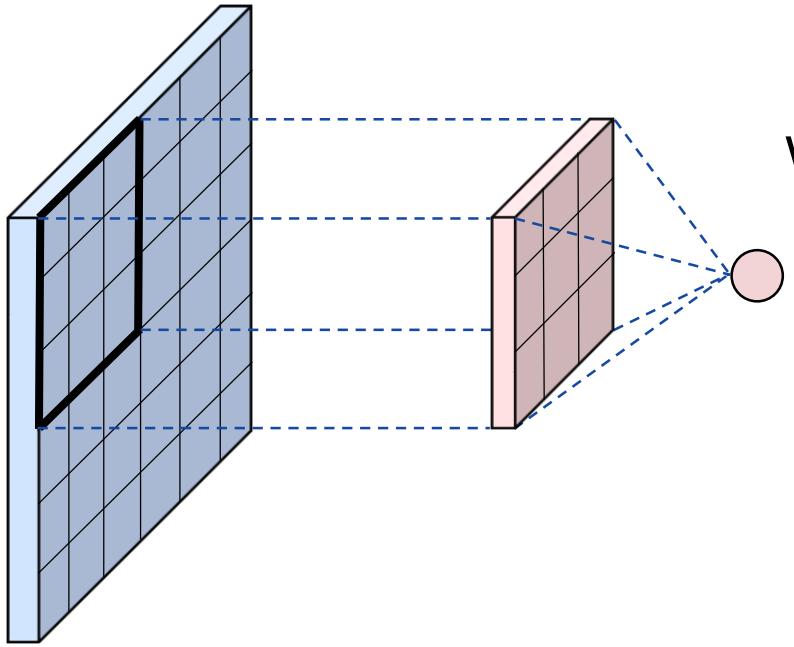


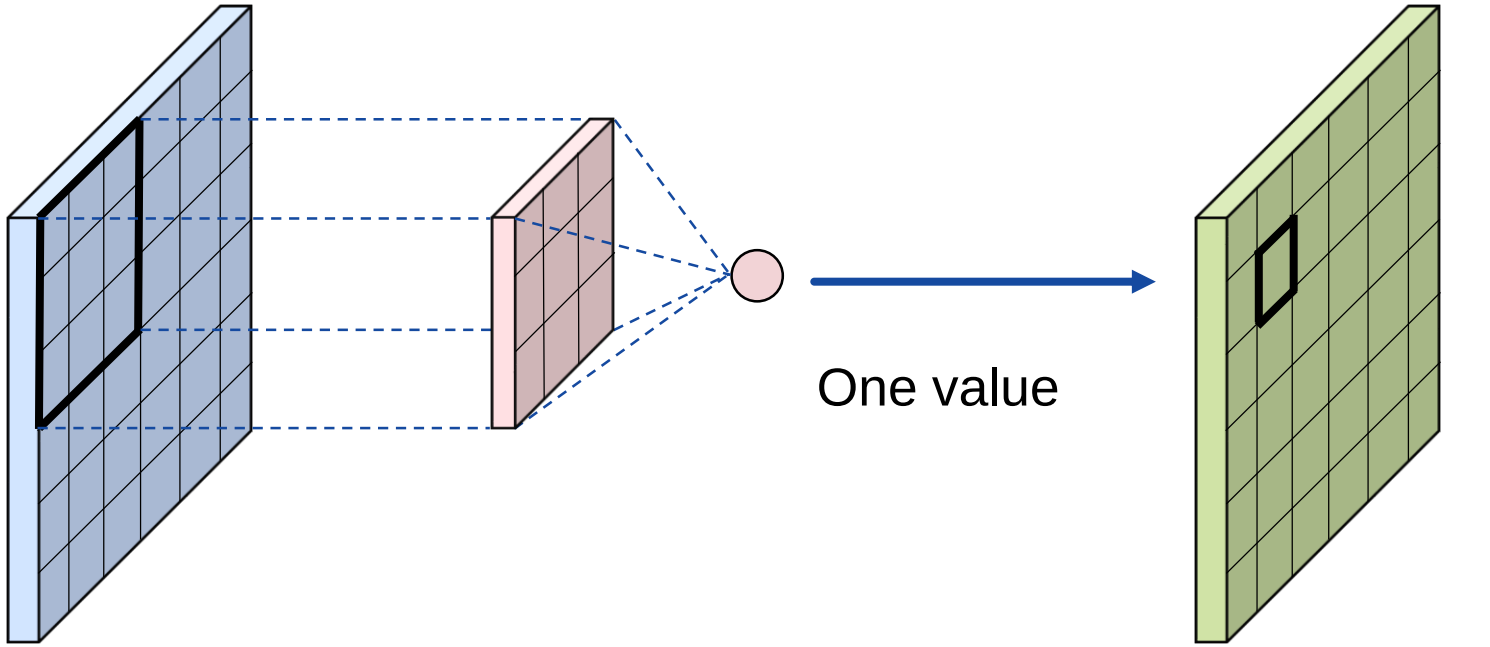
Image Filter

Weighted sum of input values, Scalar product (dot product)

$$y = \sum_i w_i \, x_i + b = \boldsymbol{w}^T \boldsymbol{x} + b$$

x_i Input
y Output
w_i Filter weights
b Bias

Convolution Layer



Convolution Layer

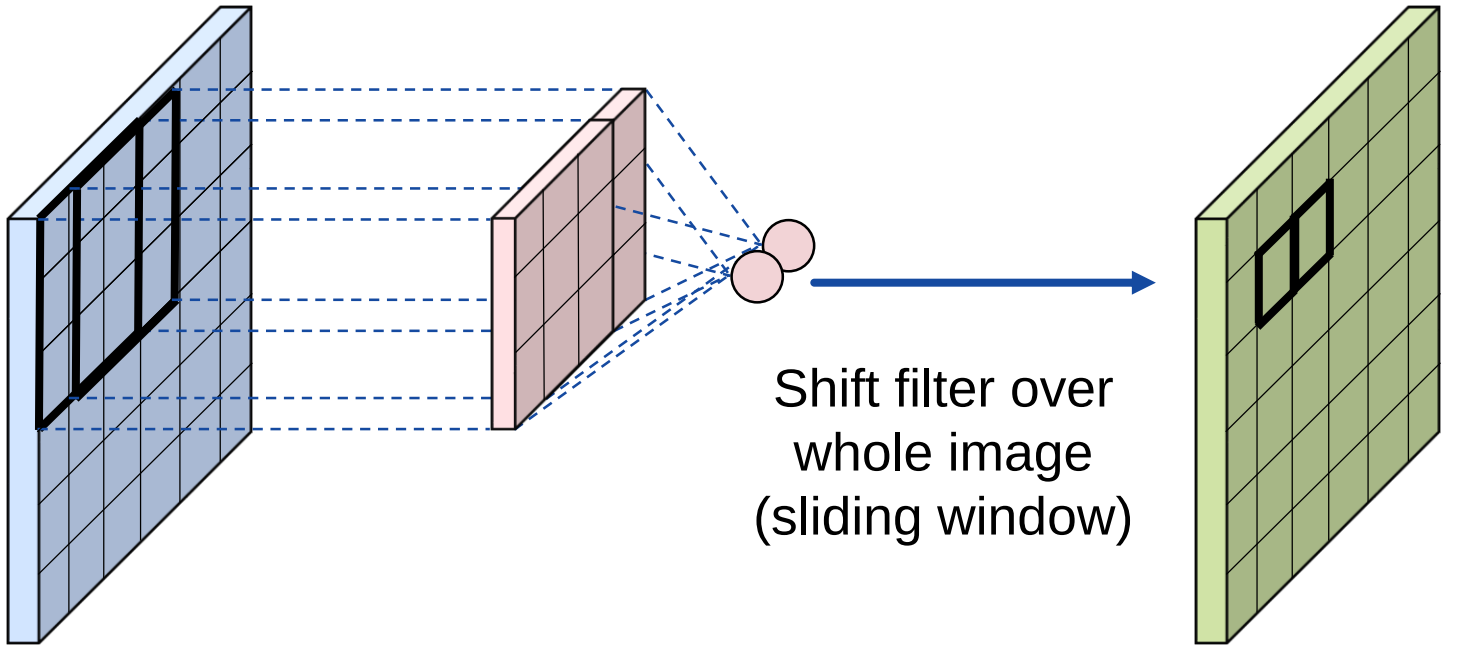
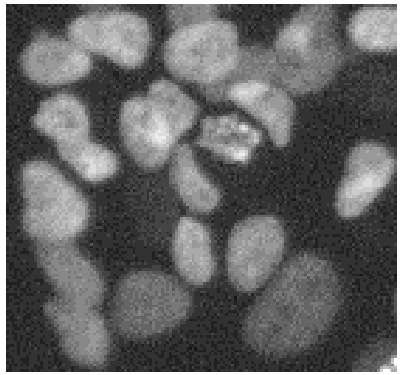


Image Filter Feature map

Convolution Layer

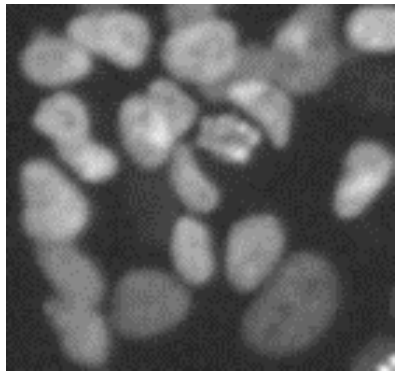
Different filters yield different feature maps



Image

1	1	1
1	1	1
1	1	1

Filter



Smoothed intensities

1	0	−1
2	0	−2
1	0	−1

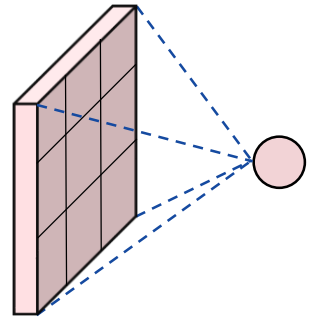
Filter

Enhanced edges

Convolution Layer

CNN: Filter weights are learned

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9



- Filter integrates information in **local neighborhood** (receptive field of neuron)
- **Shared weights**, same filter for whole image
- Reduced number of parameters compared to Fully Connected Networks
- Reduced computation time, higher robustness

Convolution

Convolution of 2D image $g(x,y)$ with 2D filter $h(x,y)$ (continuous):

$$g_{conv}(x,y) = g(x,y) * h(x,y)$$

$*$ Convolution operation

$$= \int\int g(\xi,\eta)h(x-\xi,y-\eta)d\xi d\eta$$

Convolution

Convolution of 2D image $g(x,y)$ with 2D filter $h(x,y)$ (discrete):

$$g_{conv}(x,y) = g(x,y) * h(x,y)$$

$*$ Convolution operation

$$= \sum_i \sum_j g(i,j)h(x-i,y-j)$$

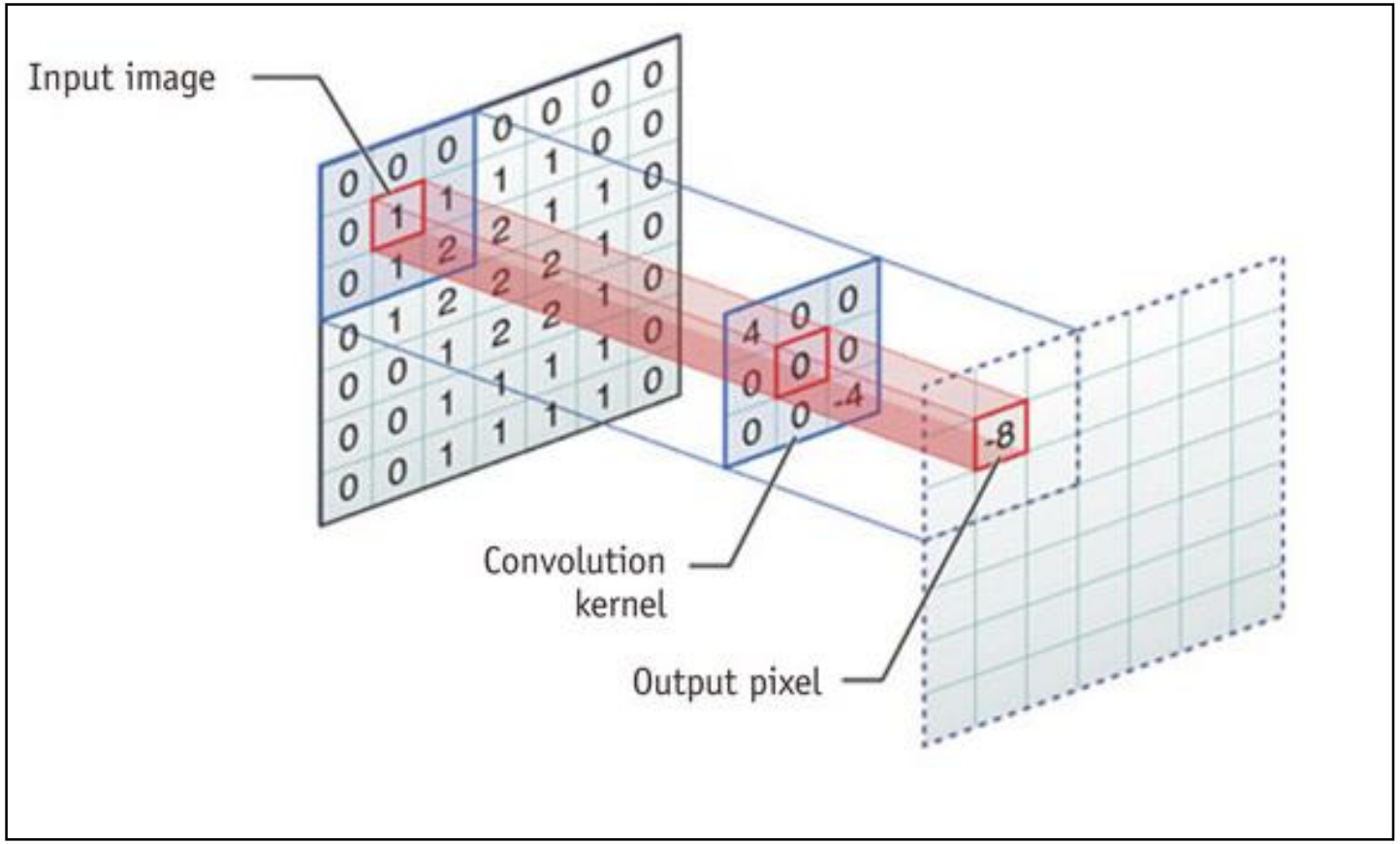
Negative sign: Flipping of filter

Main steps of convolution

1. Flipping of filter (mirroring w.r.t. x- and y-axes)
2. Multiply filter values with image intensity values
3. Summation of multiplication results

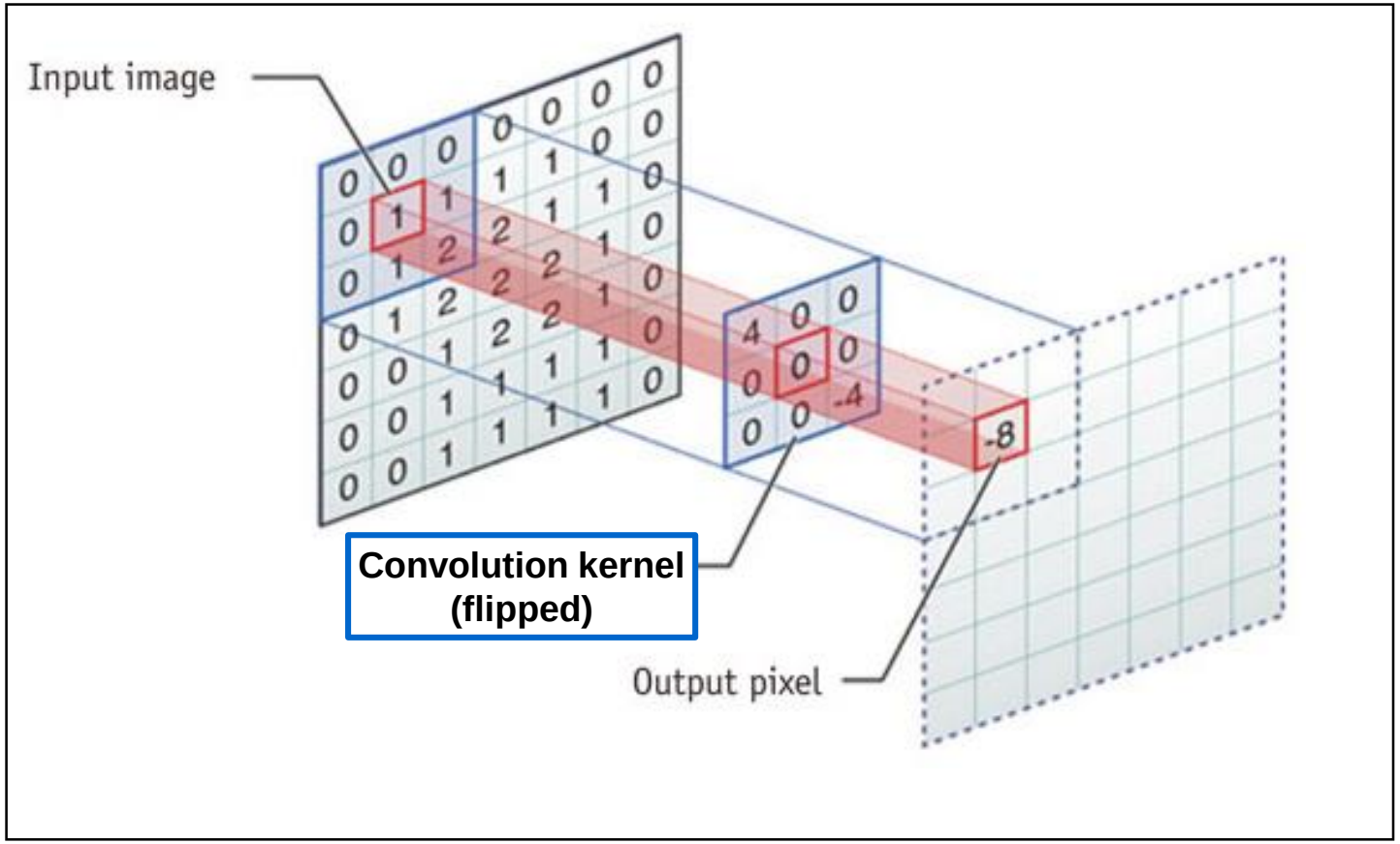
CNNs do **not** perform convolution, but cross-correlation (no flipping)

Convolution



~~Convolution~~ with filter kernel

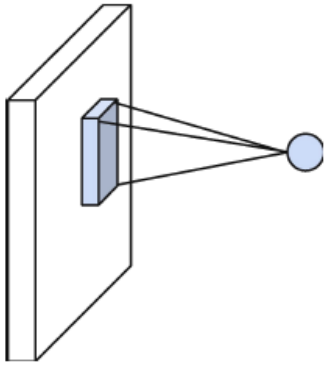
Convolution



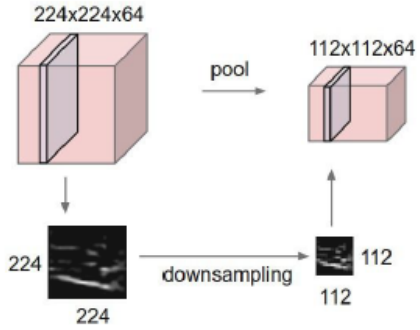
Convolution with filter kernel

CNN Components

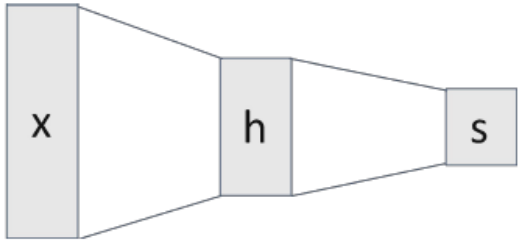
Convolution Layers



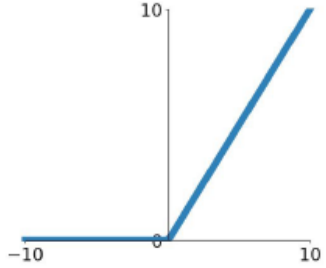
Pooling Layers



Fully-Connected Layers



Activation Function

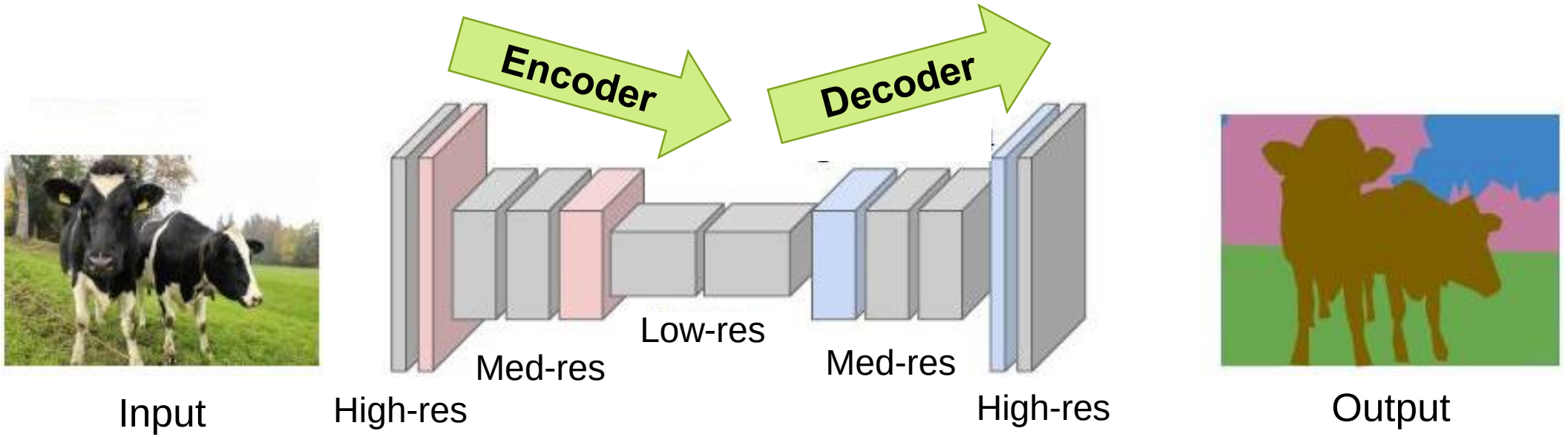


Normalization (Batch normalization)

$$\hat{x}_{i,j} = \frac{x_{i,j} - \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}}$$

Li et al. 2023, cs231n

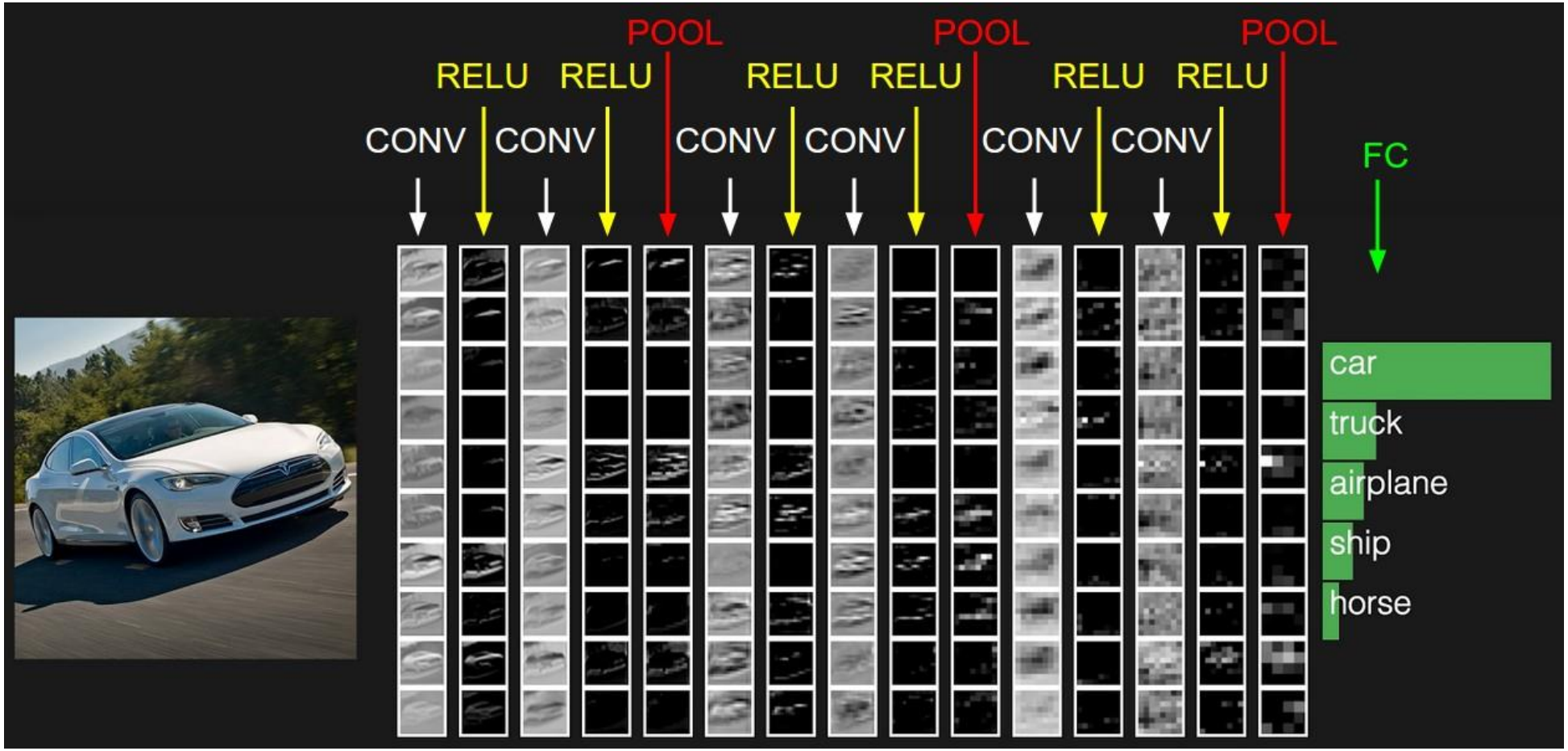
Encoder-Decoder CNN



- CNN with encoder and decoder (compression and decompression)
- Much less parameters and computations

Li et al. 2023, cs231n

Convolutional Neural Network

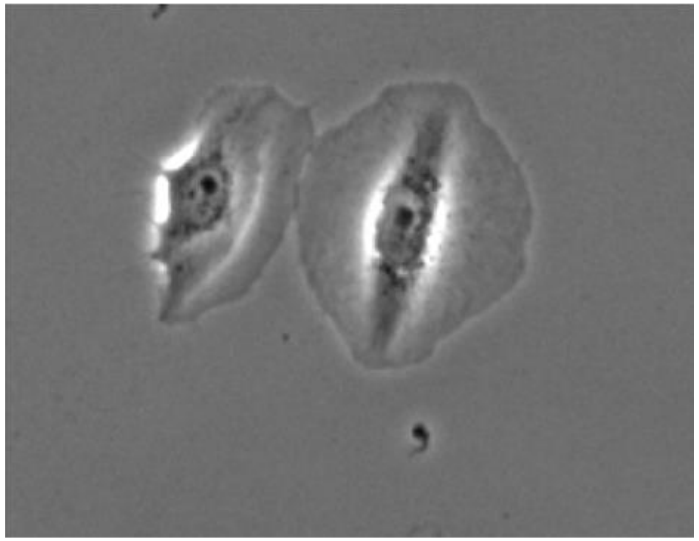


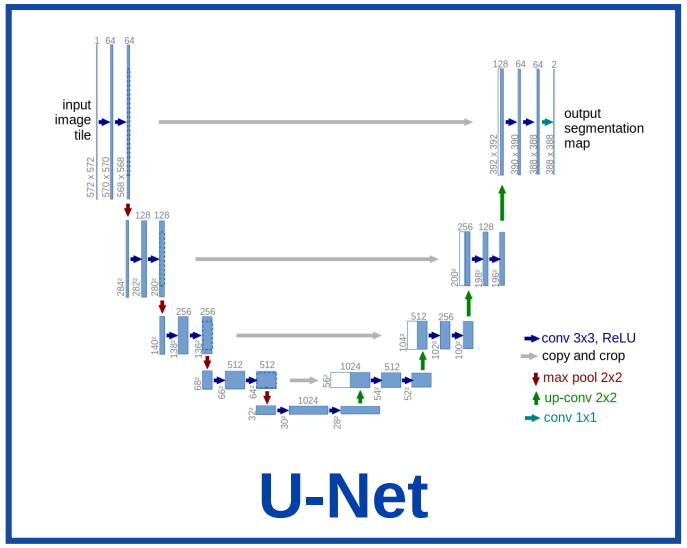
Feature maps (activation maps) of CNN for object classification
(tiny VGG ConvNet architecture)

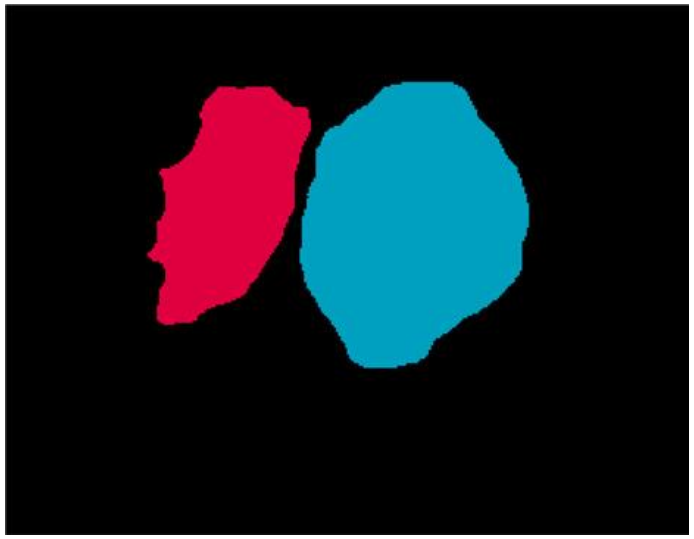
Li et al. 2023, cs231n

U-Net for Image Segmentation

U-Net for Image Segmentation







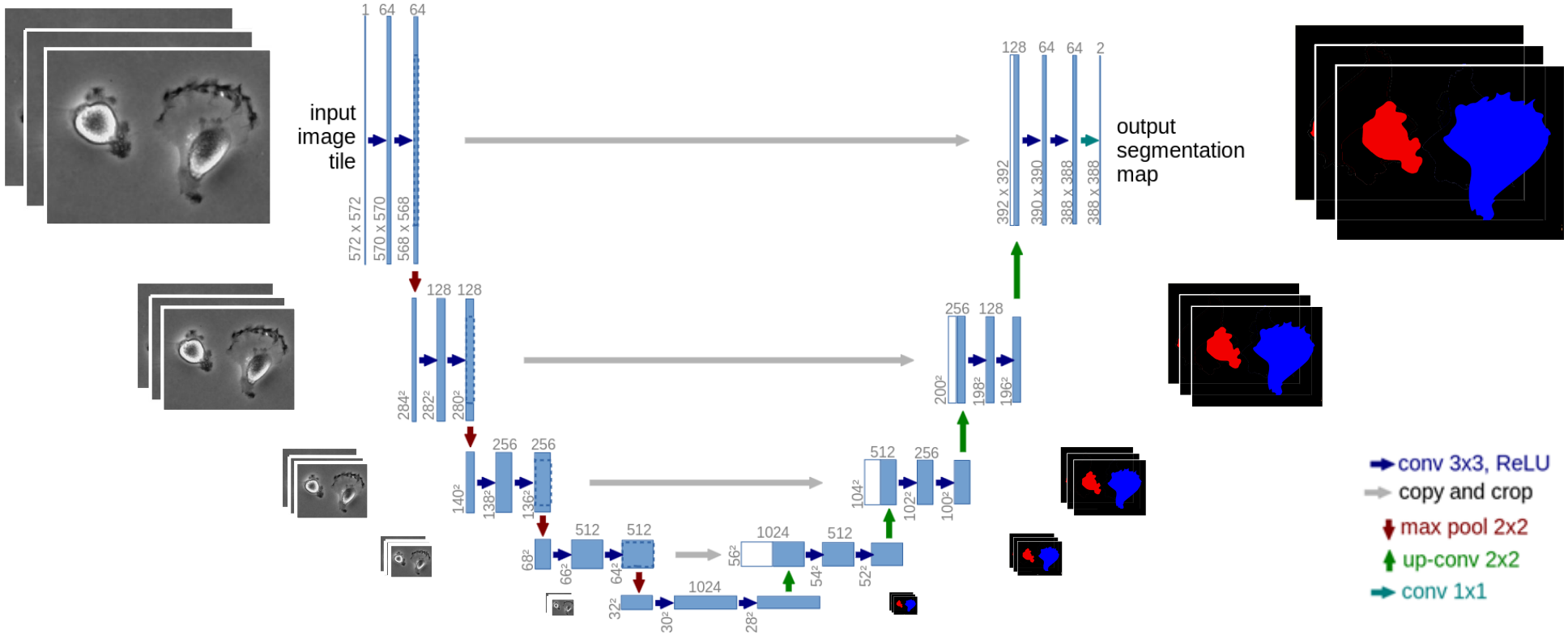
Original image

Segmented image

- **End-to-end trainable CNN** for **semantic segmentation** (pixel labelling)
- **Encoder-decoder network architecture**
- Instance segmentation by connected component labelling
- Widely used architecture for different image analysis tasks

Ronneberger et al. 2015

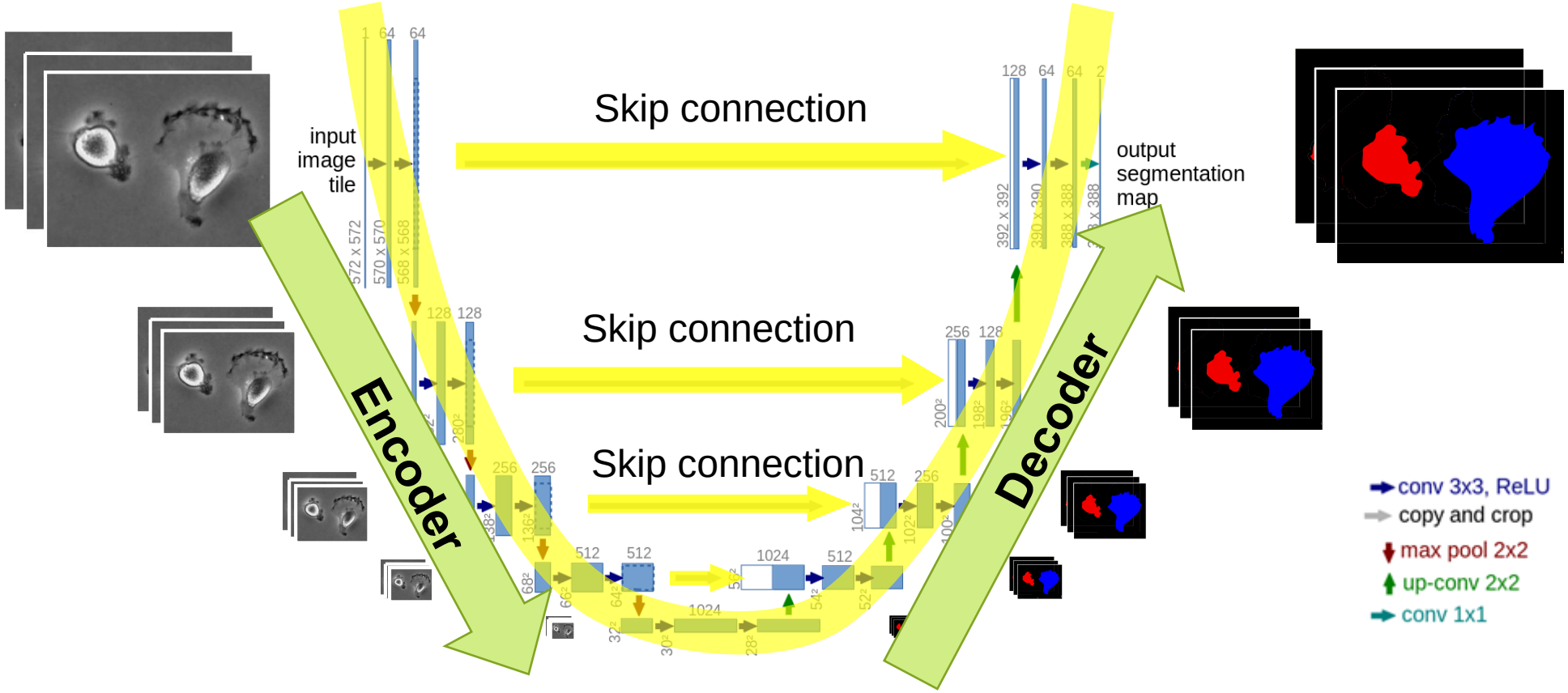
U-Net for Image Segmentation



- Network architecture: U shape
- **Encoder and decoder paths** (downsampling, upsampling)
- **Skip connections**: Connect image data with feature maps to preserve high-res info
- Pixel-wise soft-max in final layer, cross-entropy loss for training

Adapted from Ronneberger et al. 2015

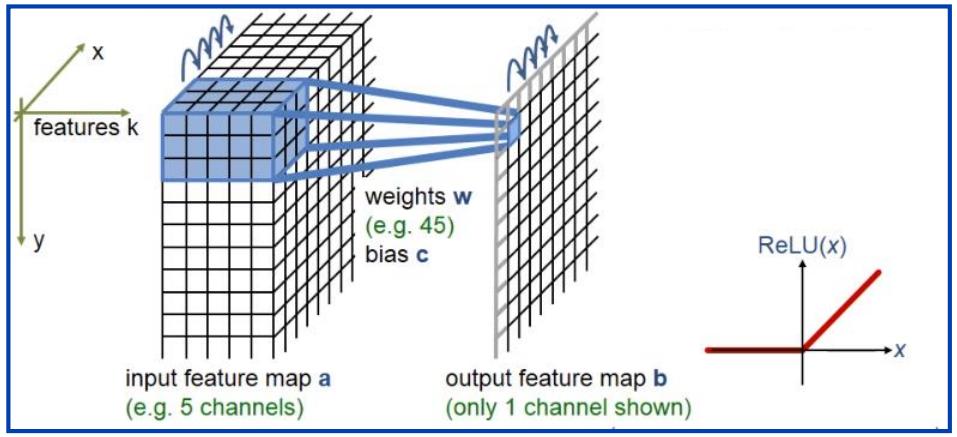
U-Net for Image Segmentation



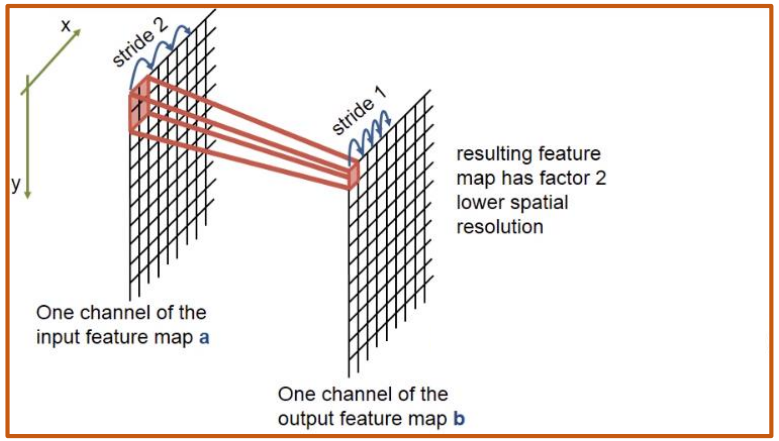
- Network architecture: U shape
- **Encoder and decoder paths** (downsampling, upsampling)
- **Skip connections**: Connect image data with feature maps to preserve high-res info
- Pixel-wise soft-max in final layer, cross-entropy loss for training

Adapted from Ronneberger et al. 2015

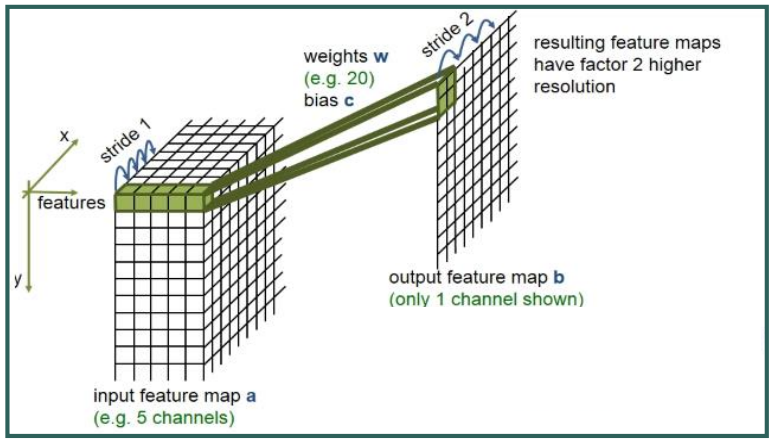
U-Net Components



Convolution, ReLU activation function



Max pooling



Up-convolution

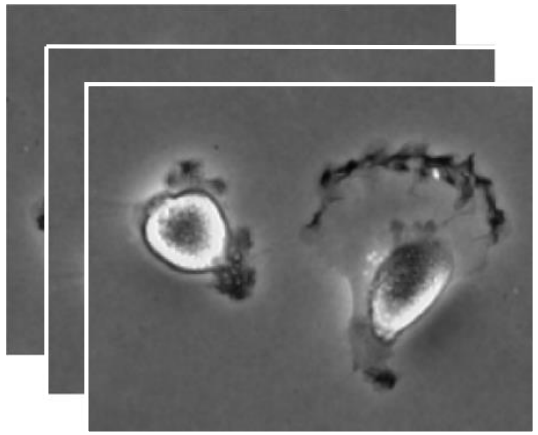
conv 3x3, ReLU

max pool 2x2 stride 2

up-conv 2x2 stride 2

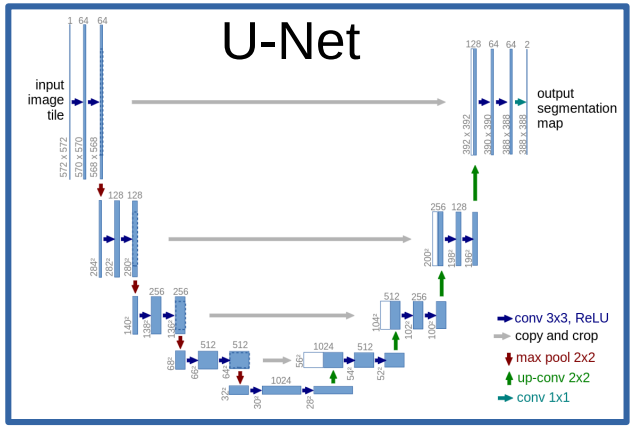
Ronneberger et al. 2015

U-Net Training



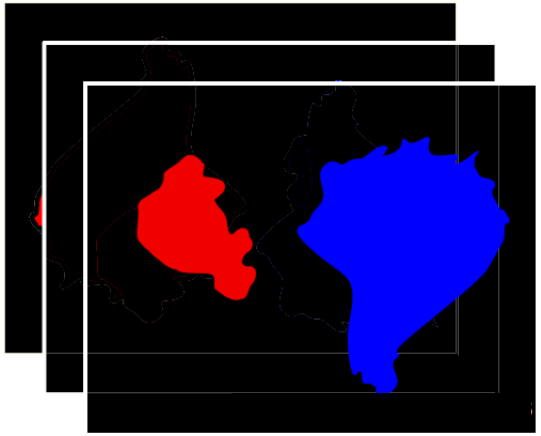
Original images

Input



Prediction

Annotation



Annotated images

Iterative adaption of network weights

Loss function

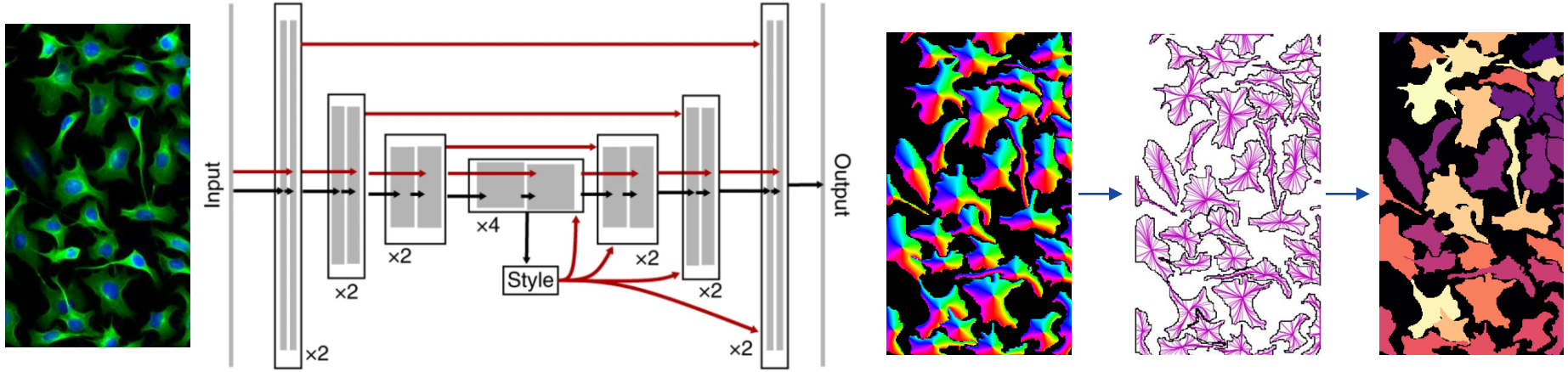
Predicted images

Adapted from Ronneberger et al. 2015, rupert ai

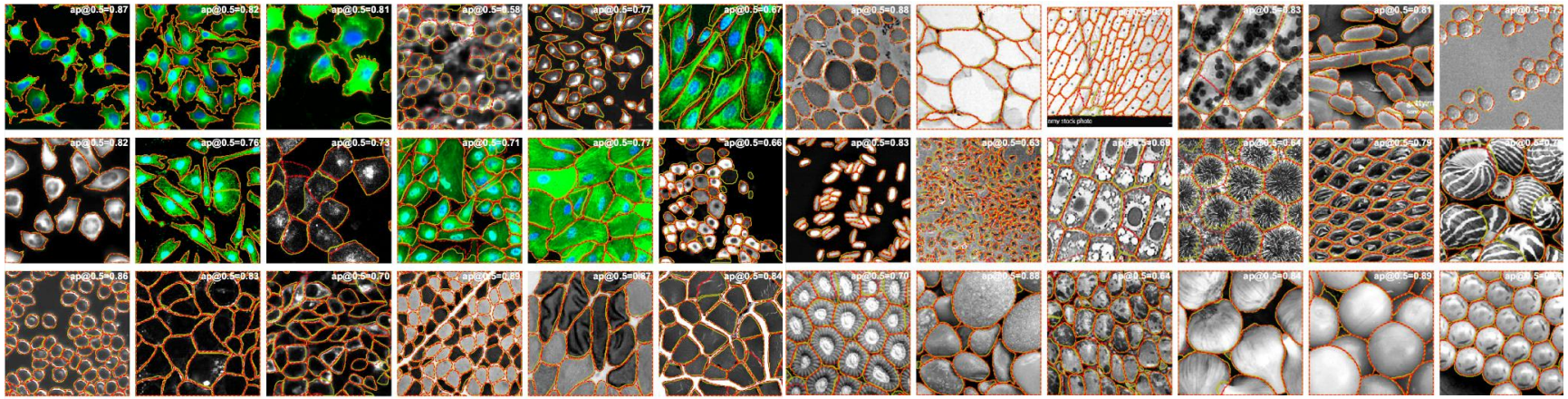
Cellpose for Image Segmentation

Cellpose

Cell instance segmentation, U-Net architecture, vector flow representation of images

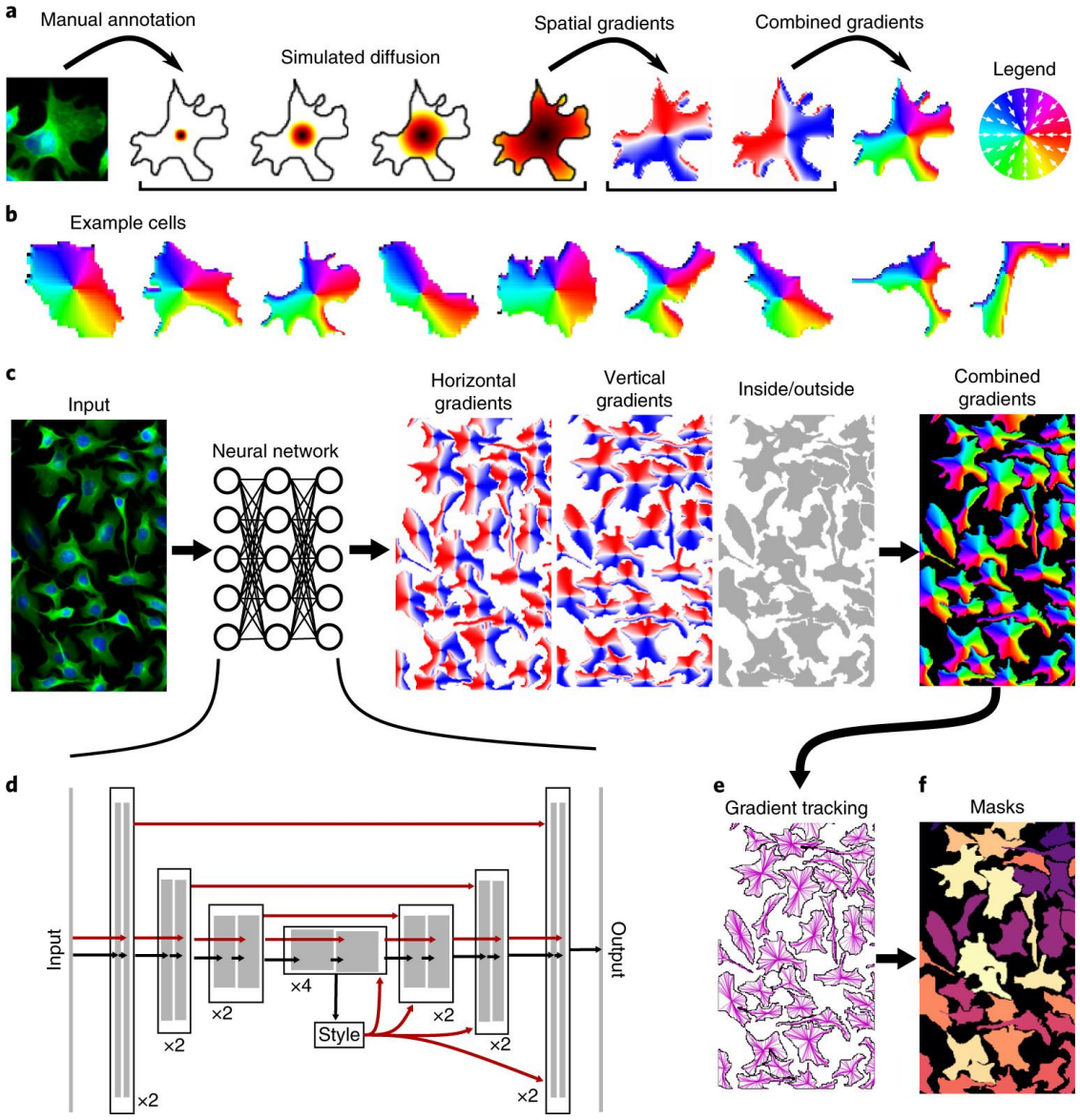


Large dataset for training (70,000 cells) ⇨ **generalist method** for cell segmentation



Stringer et al. 2021

Cellpose



Vector flow representation

- Simulated diffusion from center of annotated cell masks
- Spatial gradients pointing towards cell center

Neural network predicts

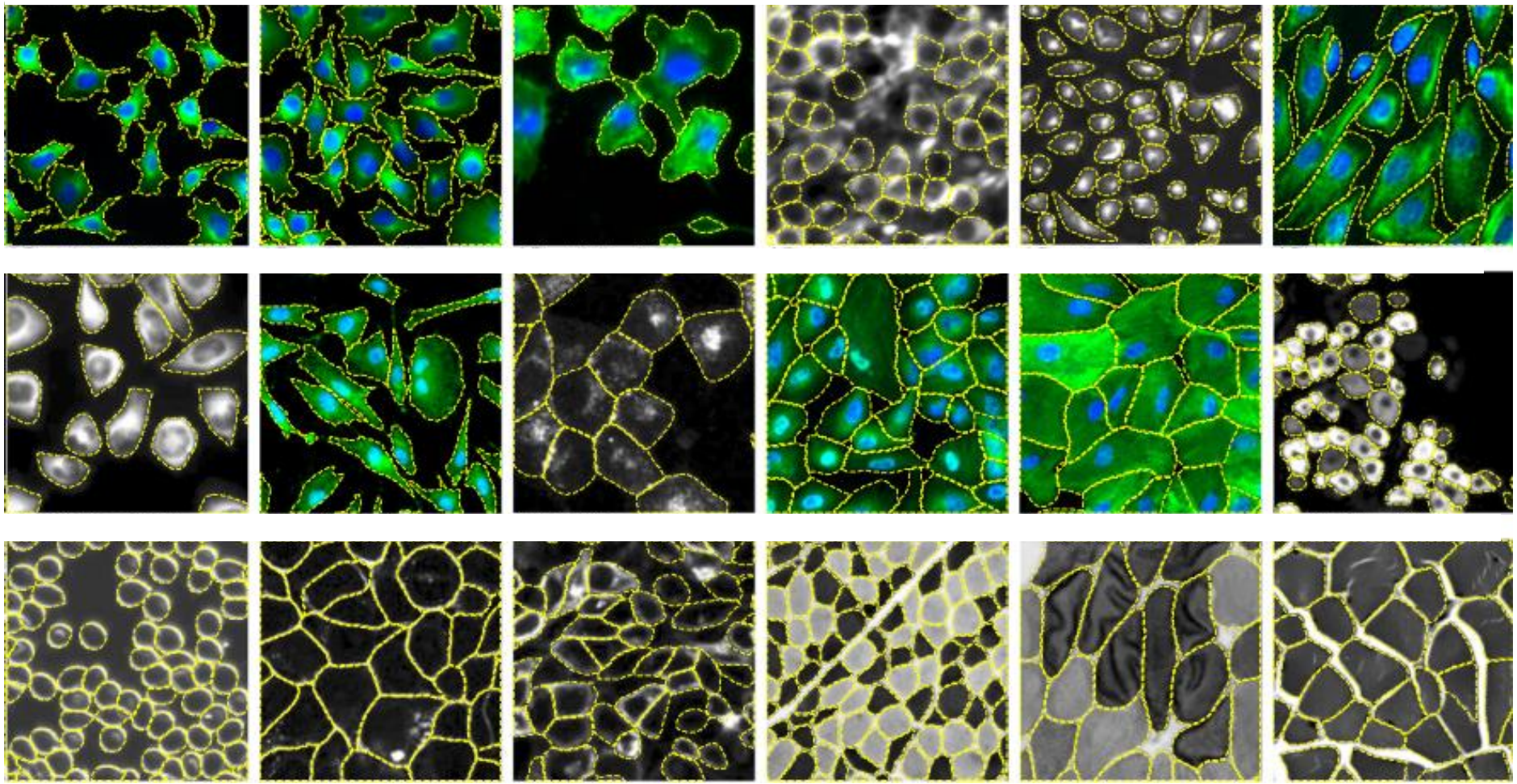
- Horizontal and vertical spatial gradients
- Foreground/background (whether a pixel belongs to a cell or not)

Cell instance segmentation

- Gradient tracking towards fixed points
- Pixels that converge to the same fixed point are assigned to the same cell mask

Stringer et al. 2021

Cellpose



Segmentation results for different cell images (yellow outlines)

Stringer et al. 2021

References

- Goodfellow I, Bengio Y, Courville A. (2016) Deep Learning, MIT Press, www.deeplearningbook.org
- Ronneberger O, Fischer P, Brox T. (2015) U-Net: Convolutional Networks for Biomedical Image Segmentation, MICCAI 2015, 234–241
- Falk T, Brox T, Ronneberger O et al. (2019) U-Net: deep learning for cell counting, detection, and morphometry, Nat Methods 16, 67-70
- Stringer C, Wang T, Michaelos M, Pachitariu M. (2021) Cellpose: a generalist algorithm for cellular segmentation, Nat Methods 18, 100–106
- Pachitariu M, Stringer C. (2022) Cellpose 2.0: how to train your own model, Nat Methods 19, 1634–1641
- Schätz M. (2023) Simple Visual Cellpose Cheat Sheet https://figshare.com/articles/figure/Simple_Visual_Cellpose_Cheat_Sheet/24441214/1

Questions ?

Deep Learning for Image Segmentation: Convolutional Neural Networks (CNNs)

Janis Meyer

Biomedical Computer Vision Group (BMCV)
BioQuant, IPMB, Heidelberg University

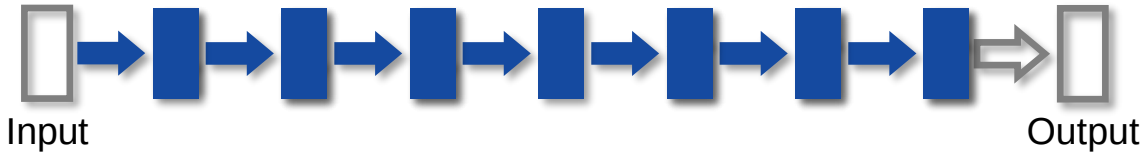
Convolutional Neural Networks

Fully Convolutional Neural Networks: networks without fully-connected layers

Specific building blocks:

- **Convolution Layers**
  (extract information from spatial data)
- **Batch Normalization**
  (improve network training)
- **Activation Functions**
  (introduce non-linearity)

Convolution Block



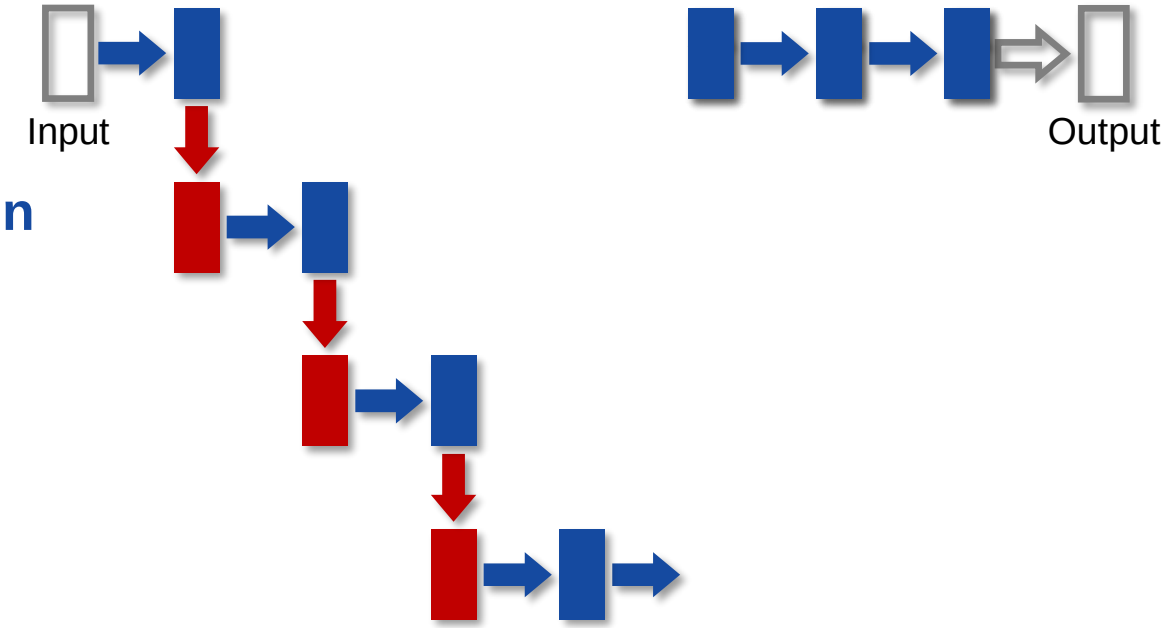
Convolutional Neural Networks

Fully Convolutional Neural Networks: networks without fully-connected layers

Specific building blocks:

- **Convolution Layers**
  (extract information from spatial data)
- **Batch Normalization**
  (improve network training)
- **Activation Functions**
  (introduce non-linearity)
- **Downsampling**
  (reduce spatial resolution)

Convolution Block



Convolutional Neural Networks

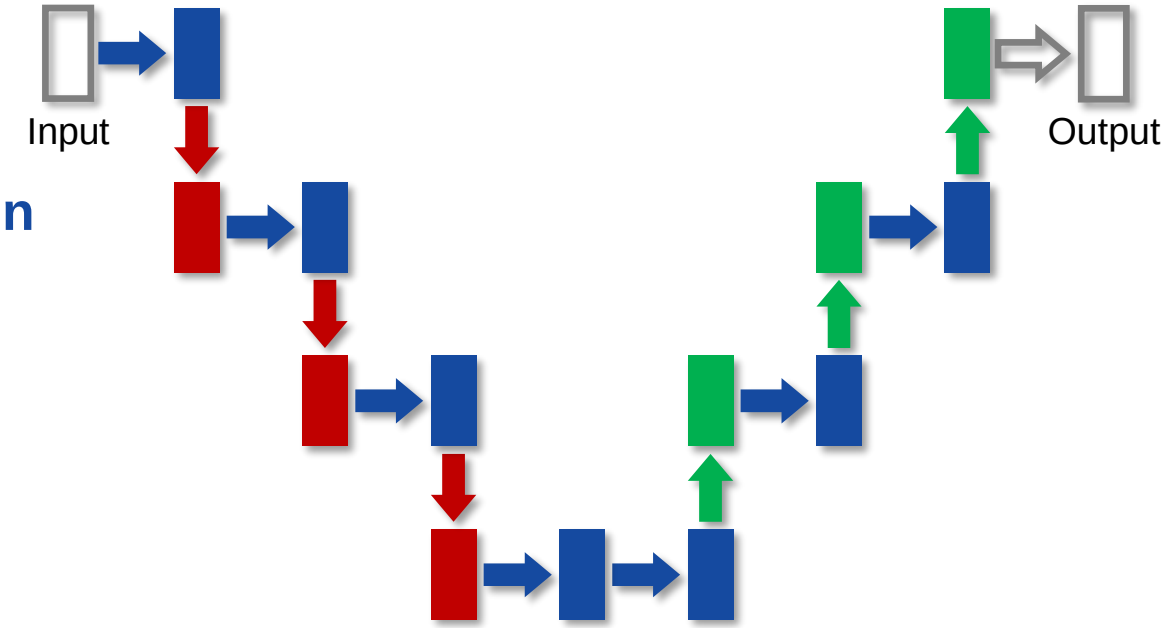
Fully Convolutional Neural Networks: networks without fully-connected layers

Specific building blocks:

- **Convolution Layers**
  (extract information from spatial data)
- **Batch Normalization**
  (improve network training)
- **Activation Functions**
  (introduce non-linearity)

Convolution Block

- **Downsampling**
  (reduce spatial resolution)
- **Upsampling**
  (increase spatial resolution, recover input resolution)



Convolutional Neural Networks

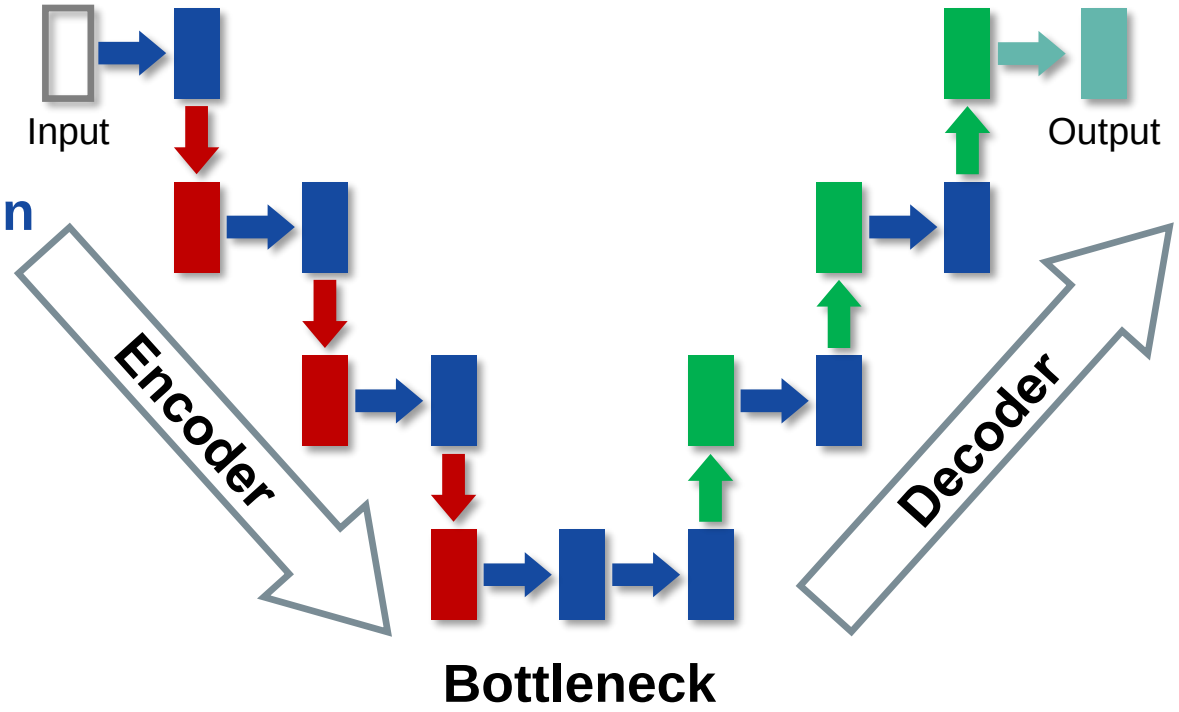
Fully Convolutional Neural Networks: networks without fully-connected layers

Specific building blocks:

- **Convolution Layers**
  (extract information from spatial data)
- **Batch Normalization**
  (improve network training)
- **Activation Functions**
  (introduce non-linearity)

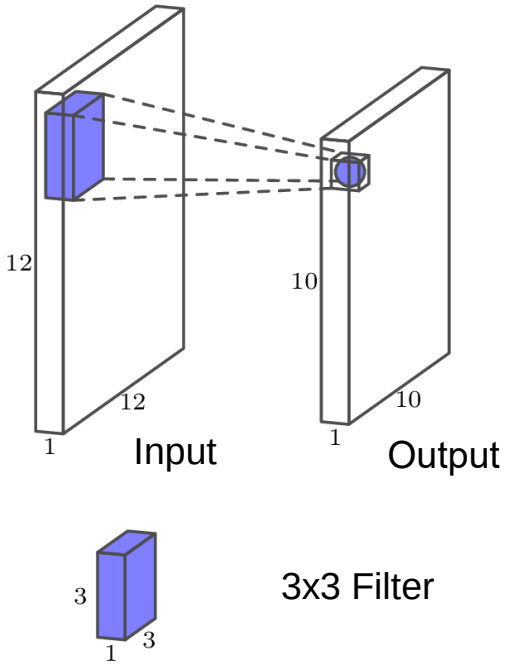
Convolution Block

- **Downsampling**
  (reduce spatial resolution)
- **Upsampling**
  (increase spatial resolution, recover input resolution)
- **Output Activation Function**
  (scale output)



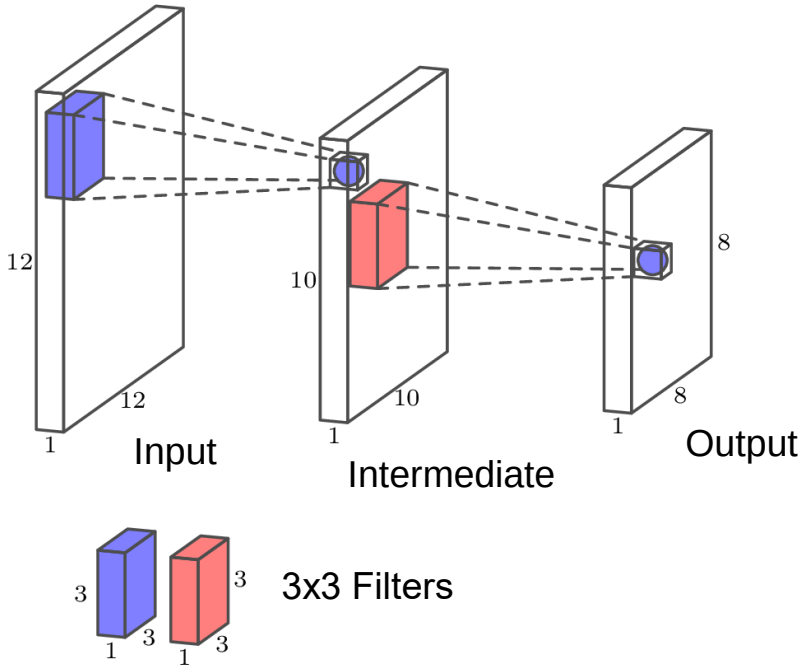
Convolution Layers

- Based on the mathematical operation of **convolution**
- Used to recover **local spatial patterns** in images (2D)
  - 1D (time series data) or 3D (volumetric data) also possible

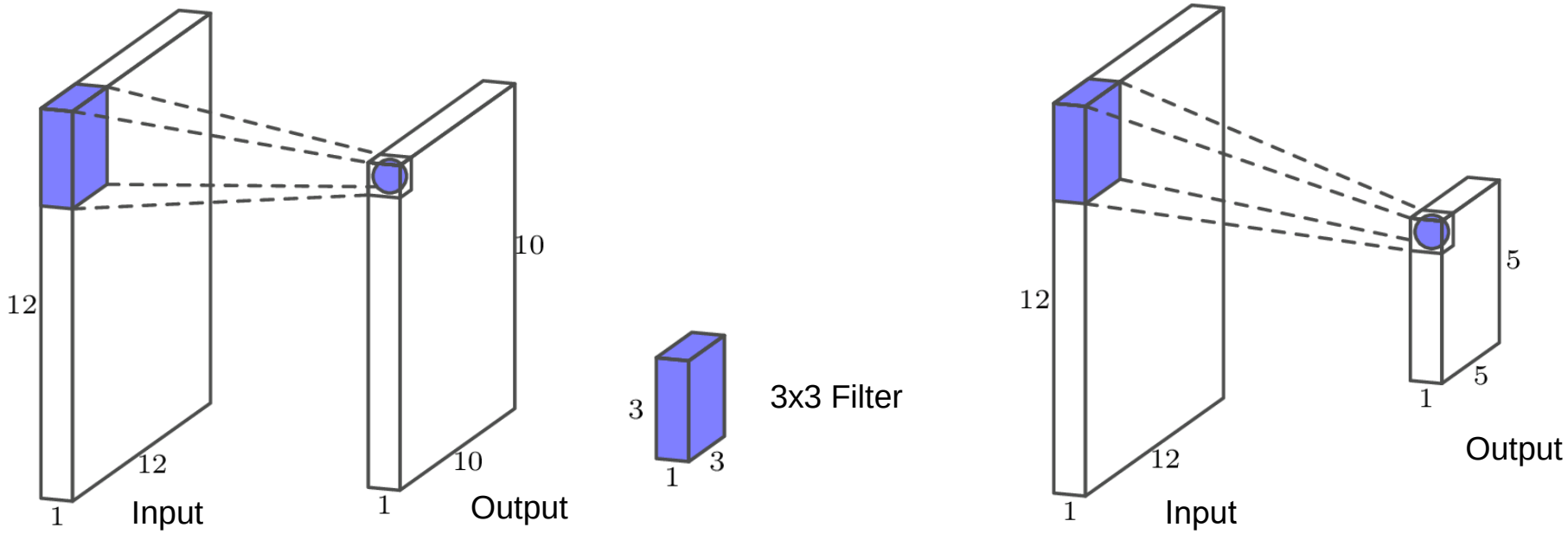


Convolution Layers

- Based on the mathematical operation of **convolution**
- Used to recover **local spatial patterns** in images (2D)
  - 1D (time series data) or 3D (volumetric data) also possible
- **Composite spatial patterns:**
  - Successive convolution layers
- Filters usually have a spatial size of 3x3 or 1x1



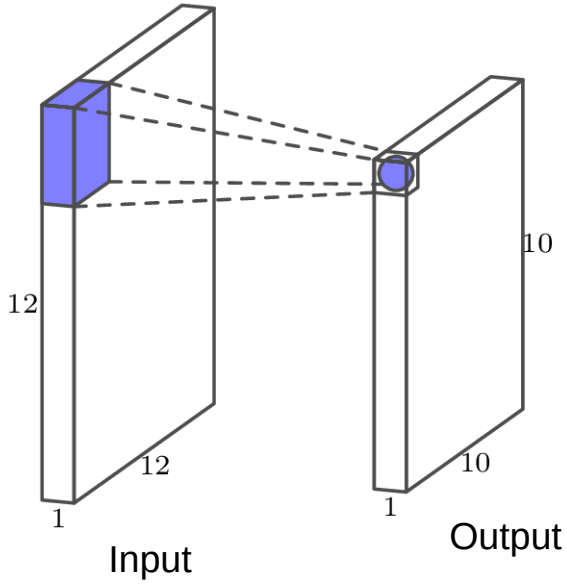
Convolution Layers

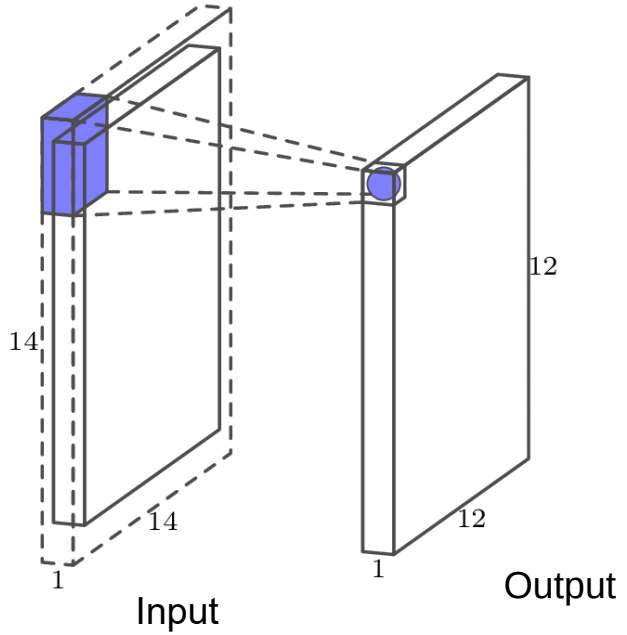


- A **learnable filter** is applied at every location in the input
  - Dot product is computed between values in local neighbourhood and the filter values
- Produces **feature maps** with an entry for each filter application
- **Strided convolutions** are applied only at every n-th location based on their stride

Convolution Layers: Border Problem

How to deal with the border problem?





No Padding (VALID)

- Consider only locations with completely included windows

➢ Output size is smaller than input

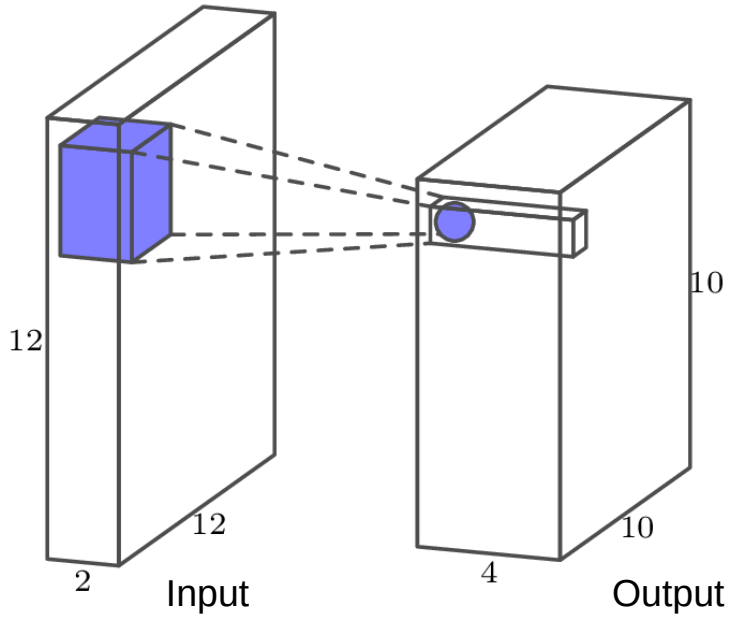
Zero-Padding (SAME)

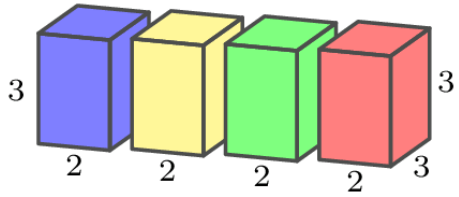
- Pad the input with zeros

➢ Input size is preserved

Convolution Layers

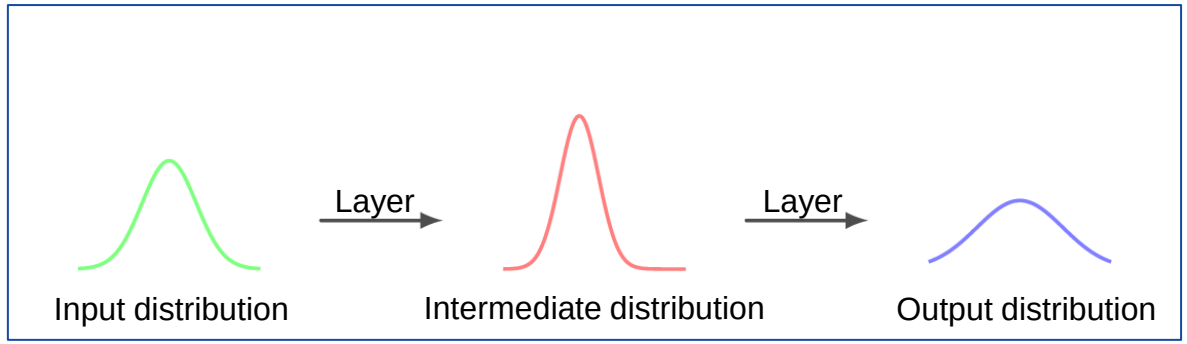
- Multiple filters are applied at the same time
  - Recover many different patterns
- Yields **feature maps** with multiple **channels** (e.g., 256, 512, 1024)
  - Filters cover the full depth of the input feature maps

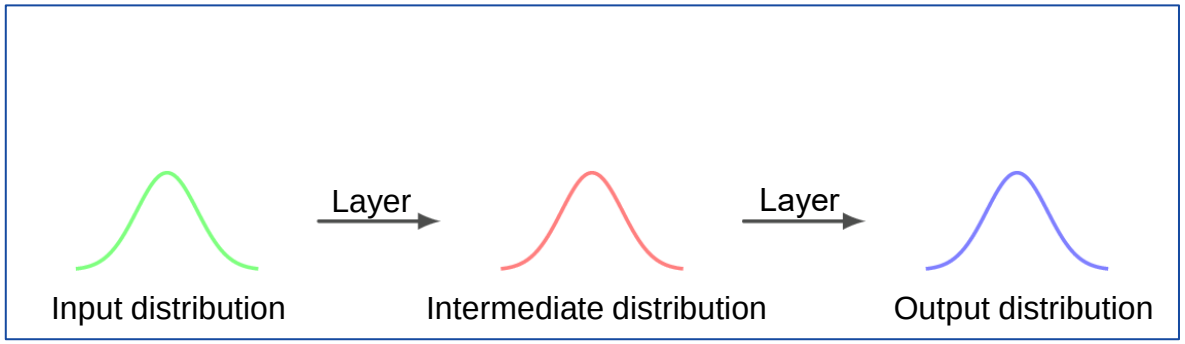




Batch Normalization

Network layer to deal with **internal covariate shift**





No Batch Normalization

- During training: parameters are updated
- Distributions of the input features of a layer change
- Every layer continuously needs to adapt to new input distributions

➢ Complicates training

With Batch Normalization

\+ Reduces importance of parameter initialization
\+ Speeds up convergence
\+ Reduces overfitting
\+ Stabilizes training

Ioffe/Szegedy, ICML 2015

Batch Normalization

Layer outputs y are normalized to have **zero mean** and **unit variance**:

$$z = \frac{y - E[y]}{\sqrt{Var[y]}}$$

$E[y]$: Expectation of y
$Var[y]$: Variance of y

However, this limits the expressiveness of the network

- Normalized outputs z are scaled and shifted using learnable parameters γ and β:

$$y_s = \gamma z + \beta$$

- **Training**: Statistics are calculated along the **batch** dimension and for each feature value separately
- **Inference**: Statistics from training are used

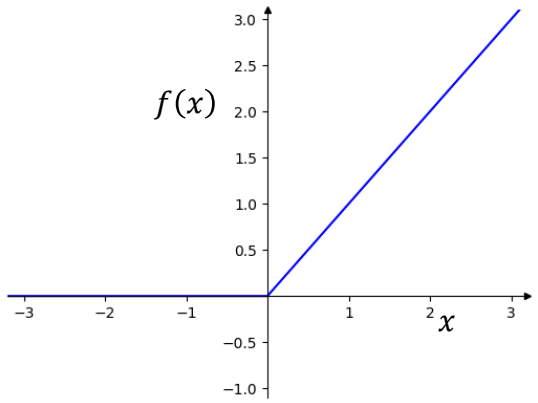
Alternative Versions: LayerNorm, GroupNorm

Ioffe/Szegedy, ICML 2015

Activation Functions

Introduce non-linearity into the network

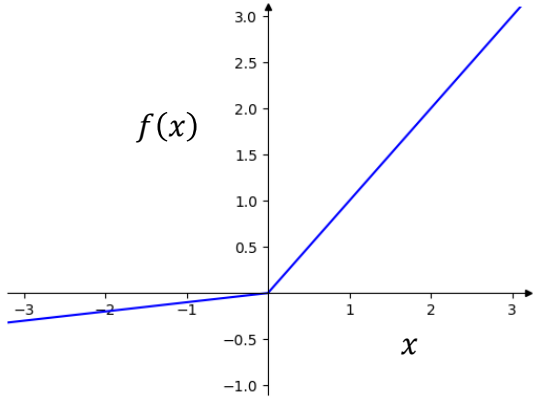
➢ learn non-linear functions



ReLU (Rectified Linear Unit)

$$f(x) = \max(0, x)$$

+ Greatly accelerates convergence
+ Simple and cheap to compute
+ No saturation problem
- Dead neuron problem
- Most common choice



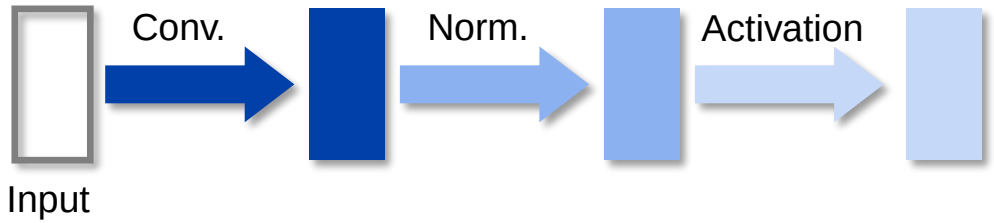
Leaky ReLU

$$f(x) = \begin{cases} \alpha x, & x < 0 \\ x, & x \geq 0 \end{cases}$$

+ Fixes the dead neuron problem
- Adds additional hyperparameter

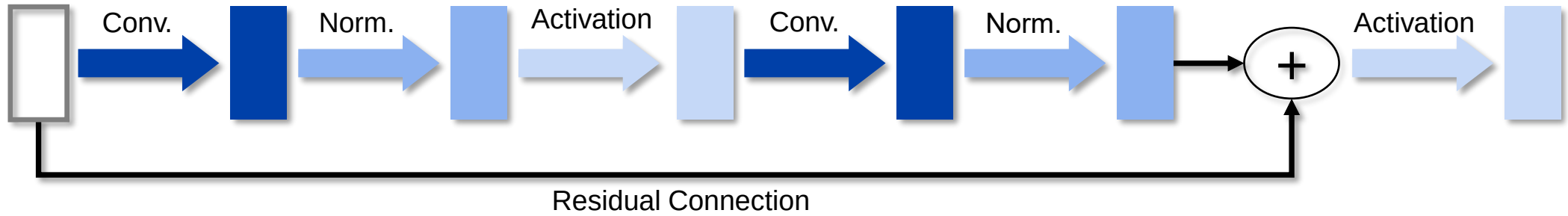
Convolution Block

Regular Convolution Block



- Convolution:
  - Kernel Size: 3x3
  - Padding: SAME
  - Stride: 1
- Normalization:
  - usually Batch Norm
- Activation: ReLU

Residual Convolution Block

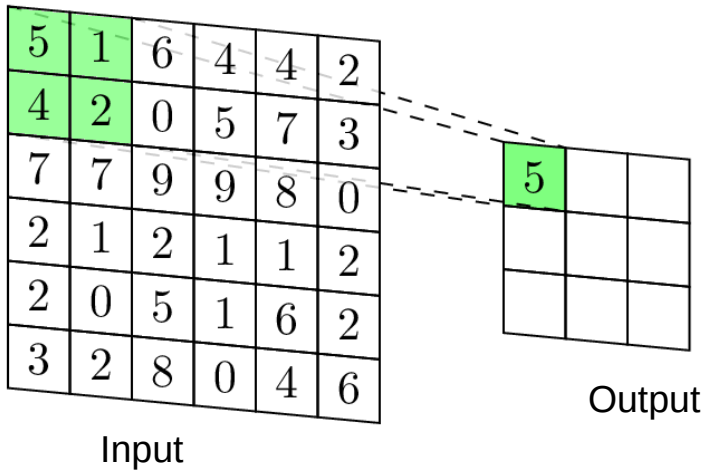


He et al., CVPR 2016

Downsampling

Reduce the spatial size by half to reduce computational load

- usually followed by doubling the number of channels in the next convolution layer



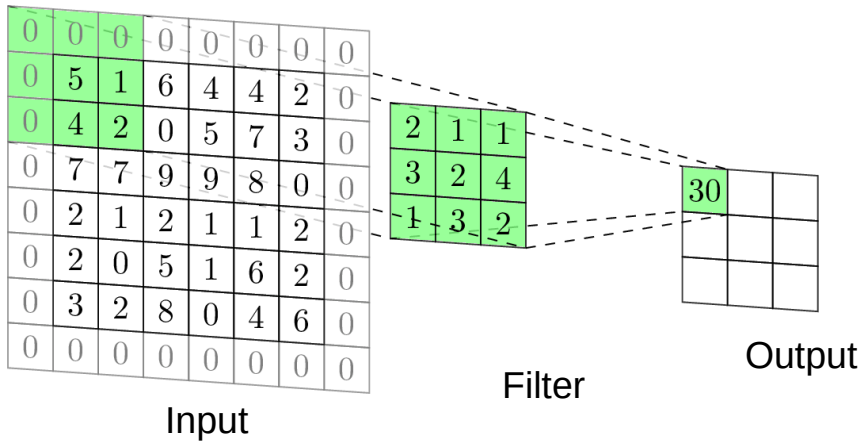
Max Pooling

- Most common choice for downsampling
- Applies a 2x2 maximum filter with stride 2
  - Alternatively: mean filter
- Can be reversed for simple upsampling (Unpooling)

Downsampling

Reduce the spatial size by half to reduce computational load

- usually followed by doubling the number of channels with the next convolution layer



Strided Convolution

- Convolution with 3x3 filter, stride 2 and zero-padding

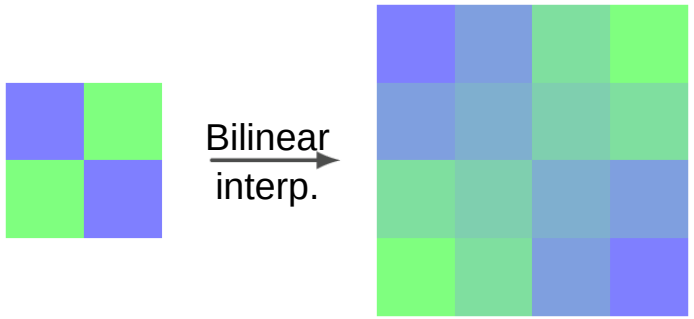
\+ Learn the downsampling operation

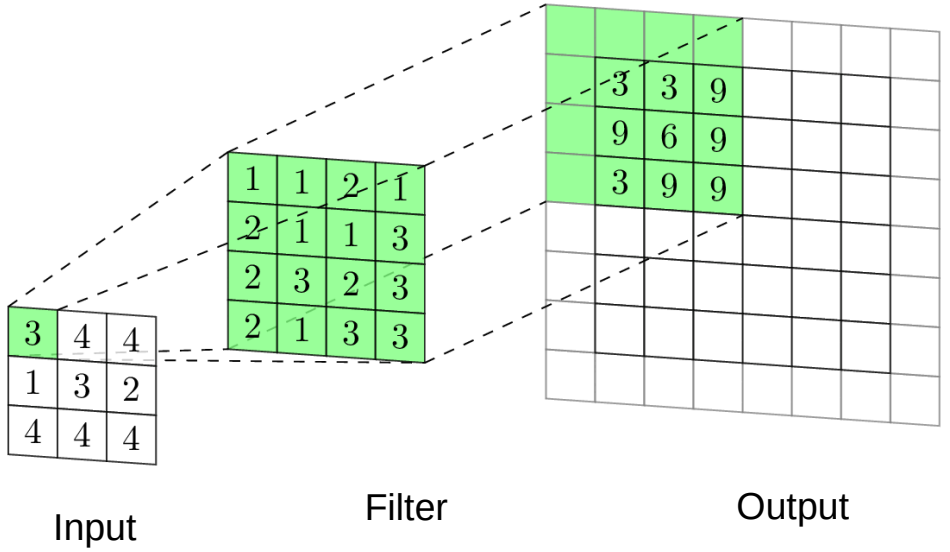
\- Slower than pooling

\- Increases number of parameters in the network

Upsampling

Double the spatial size and eventually restore the original input size





Interpolation

(nearest neighbour, bilinear)

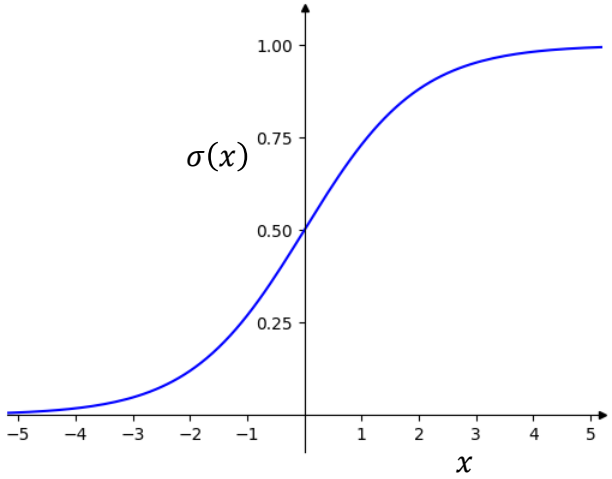
+ No parameters needed
+ Fast to execute
• Most common choice

Transposed Convolution

• Reverse convolution operation
+ Learn the upsampling operation
- Additional parameters required
- Relatively expensive to compute

Output-Activation Functions

Scale the network output to a certain range



Sigmoid

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- Squash values to [0,1]
- Used for binary segmentation

$$f\left(\begin{bmatrix} 1.0 \\ 2.3 \\ 0.6 \\ 4.2 \end{bmatrix}\right) = \begin{bmatrix} 0.0335 \\ 0.1228 \\ 0.0224 \\ 0.8213 \end{bmatrix}$$

Softmax

$$f_i(x) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

- Input is vector of values for each class
- Scales values to [0,1] so they sum up to 1
- Used for multi-class segmentation

Loss Functions

1. Measure the error of the network output y compared to the ground truth $\hat{y}$
2. Update the model parameters to reduce the error

$\hat{y}_{i,c}$: ground truth for pixel i and class c
$y_{i,c}$: prediction for pixel i and class c

Mean-Squared Error

$$L_{MSE} = \frac{1}{N}\sum_{i}^{N}\left(\frac{1}{2}(\hat{y}_i - y_i)^2\right)$$

Default choice for regression

Cross-Entropy Loss

$$L_{CE} = \frac{1}{N}\sum_{i}^{N}\left(\sum_{c}^{C} -\hat{y}_{i,c}\log y_{i,c}\right)$$

Default choice for classification

Dice Loss

$$L_{Dice} = 1 - \frac{1}{C}\sum_{c}^{C}\left(\frac{2\sum_{i}^{N}(\hat{y}_{i,c}\cdot y_{i,c})}{\sum_{i}^{N}\hat{y}_{i,c} + \sum_{i}^{N} y_{i,c}}\right)$$

Measures the overlap between predicted class and ground truth class

Loss Functions

1. Measure the error of the network output y compared to the ground truth $\hat{y}$
2. Update the model parameters to reduce the error

$\hat{y}_{i,c}$: ground truth for pixel i and class c
$y_{i,c}$: prediction for pixel i and class c

Pixel-wise evaluation

Calculated for each pixel i and then averaged

Mean-Squared Error

$$L_{MSE} = \frac{1}{N}\sum_{i}^{N}\left(\frac{1}{2}(\hat{y}_i - y_i)^2\right)$$

Default choice for regression

Cross-Entropy Loss

$$L_{CE} = \frac{1}{N}\sum_{i}^{N}\left(\sum_{c}^{C} -\hat{y}_{i,c}\log y_{i,c}\right)$$

Default choice for classification

Object-wise evaluation

Calculated for each class c and then averaged

Dice Loss

$$L_{Dice} = 1 - \frac{1}{C}\sum_{c}^{C}\left(\frac{2\sum_{i}^{N}(\hat{y}_{i,c}\cdot y_{i,c})}{\sum_{i}^{N}\hat{y}_{i,c} + \sum_{i}^{N} y_{i,c}}\right)$$

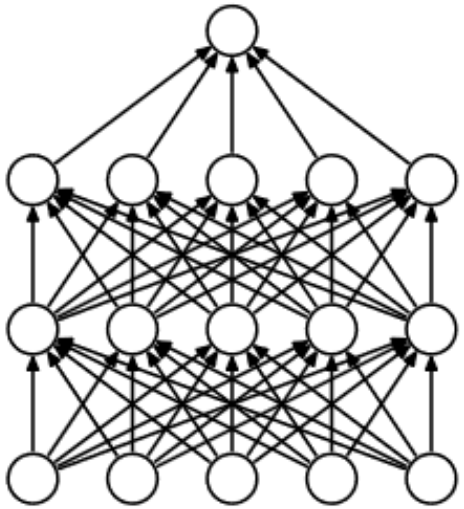
Measures the overlap between predicted class and ground truth class

\+ Better object-level performance

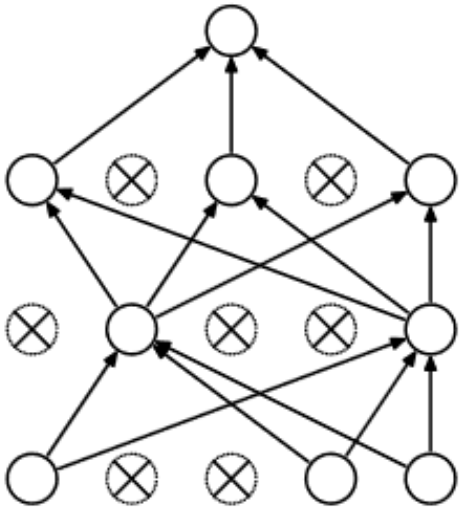
\- Less accurate at object boundaries

Dropout

Network layer designed to reduce overfitting and improve overall performance



Standard Neural Network



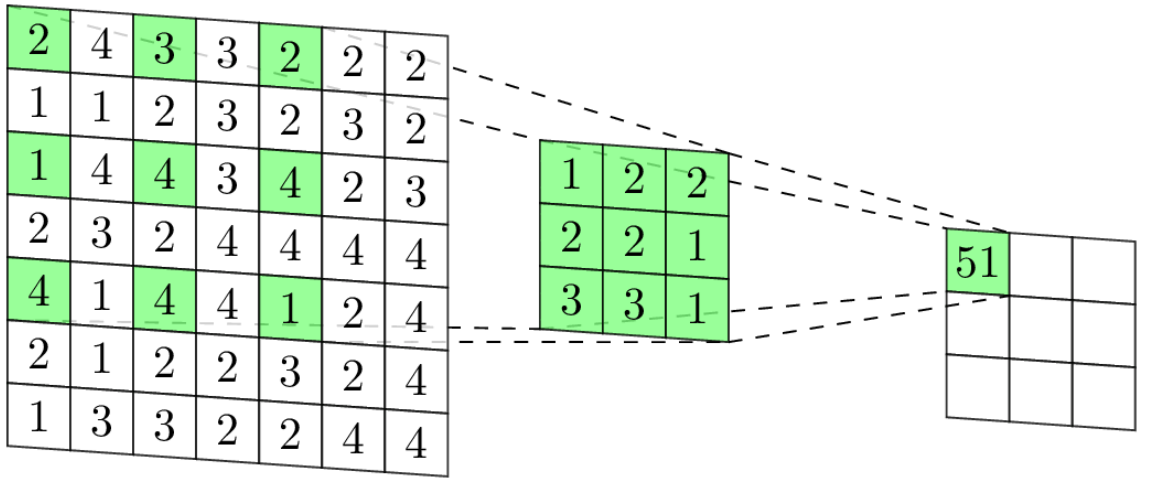
After applying dropout

Connections between consecutive layers are randomly dropped during training

- Forces the network to learn robust representations

Srivastava et al., JMLR 2014

Dilated Convolution



- Convolution is not calculated between the filter and a local input window, but a local **dilated** input window
- ➢ Increases the **field of view** of the convolution, i.e. the convolution can capture more of the input

References

- Ioffe S and Szegedy C. (2015) Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. ICML 37, 448–456
- He K, Zhang X, Ren S, and Sun J. (2016) Deep Residual Learning for Image Recognition. CVPR 2016, 770-778
- Srivastava N, Hinton GE, Krizhevsky A, Sutskever I, and Salakhutdinov R. (2014) Dropout: A Simple Way to Prevent Neural Networks from Overfitting. JMLR 15, 1929-1958

Questions ?

Deep Learning for Image Segmentation: U-Net

Kerem Celikay

Biomedical Computer Vision Group (BMCV)

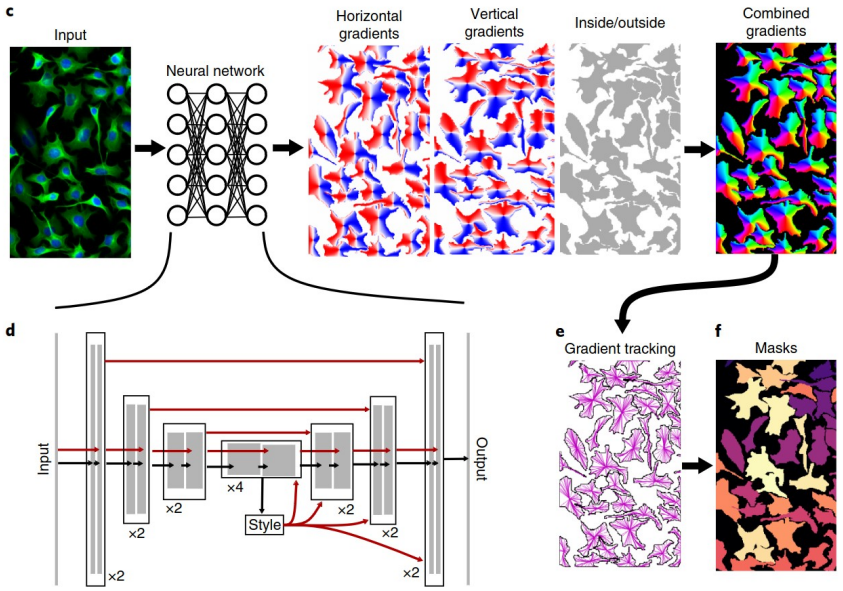
BioQuant, IPMB, Heidelberg University

Motivation

- Nowadays, U-Net is often used as a backbone for many different tasks
  - Instance/semantic image segmentation
  - Image reconstruction and denoising
  - Image generation (e.g., Diffusion models, GANs)
  - …

Motivation

- Nowadays, U-Net is often used as a backbone for many different tasks
  - ➔ Instance/semantic image segmentation
  - ➔ Image reconstruction and denoising
  - ➔ Image generation (e.g., Diffusion models, GANs)
  - ➔ …



Cellpose for instance cell segmentation
(Stringer et al., Nature Methods 2020)

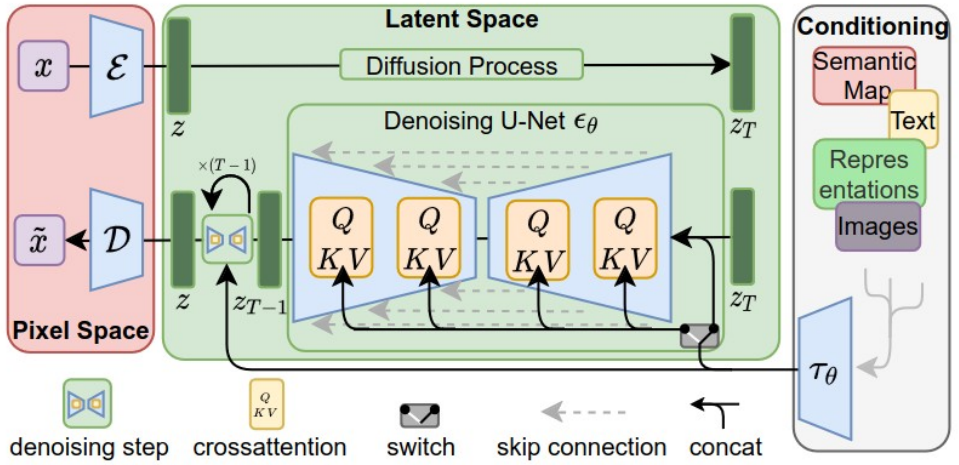
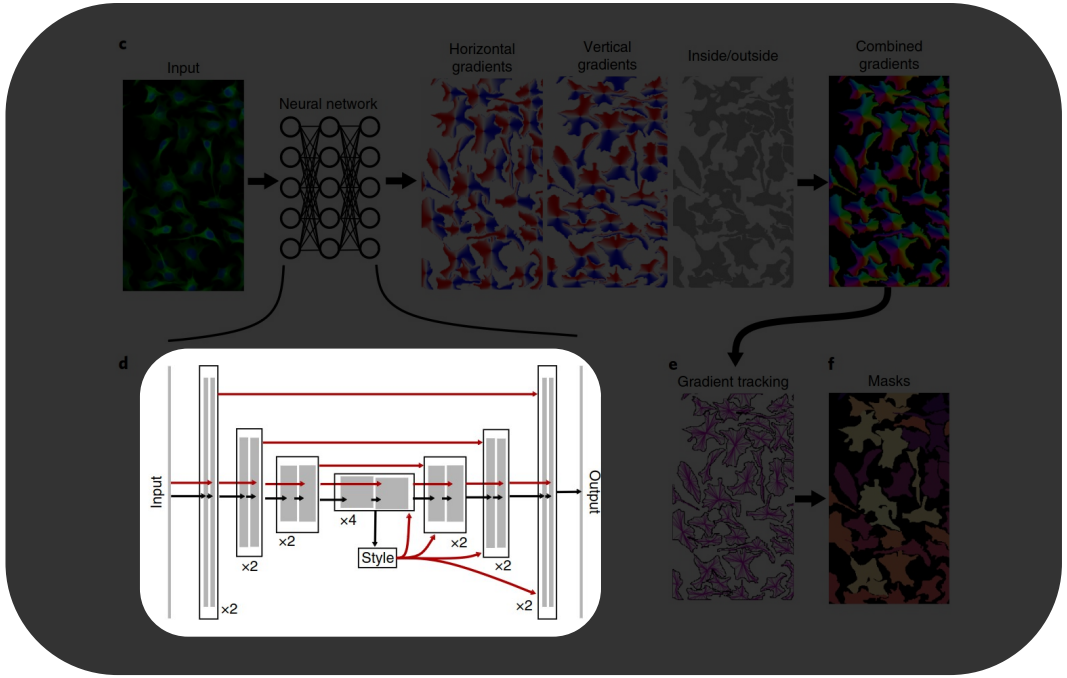


Image generation using diffusion models
(Rombach et al., CVPR 2022)

Motivation

- Nowadays, U-Net is often used as a backbone for many different tasks
  - ➔ Instance/semantic image segmentation
  - ➔ Image reconstruction and denoising
  - ➔ Image generation (e.g., Diffusion models, GANs)
  - ➔ …



Cellpose for instance cell segmentation
(Stringer et al., Nature Methods 2020)

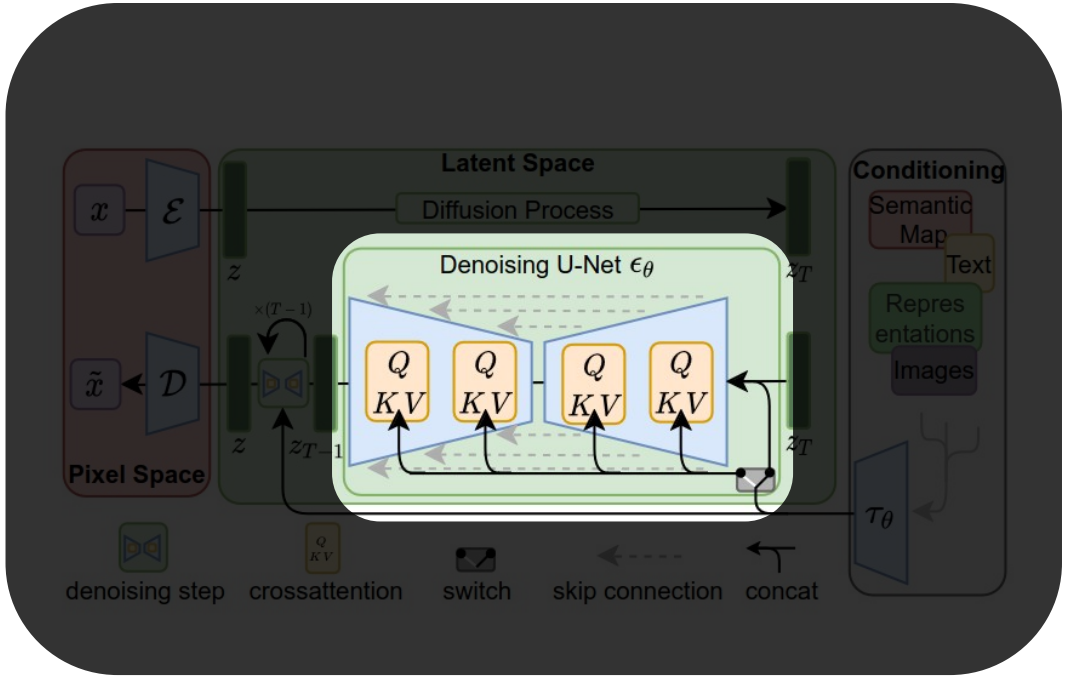
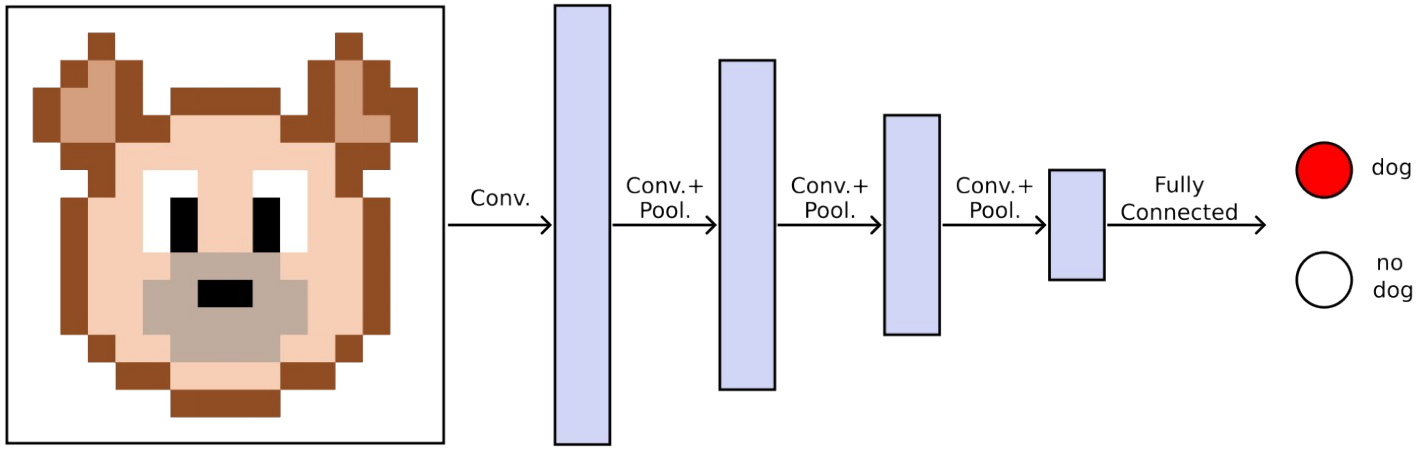


Image generation using diffusion models
(Rombach et al., CVPR 2022)

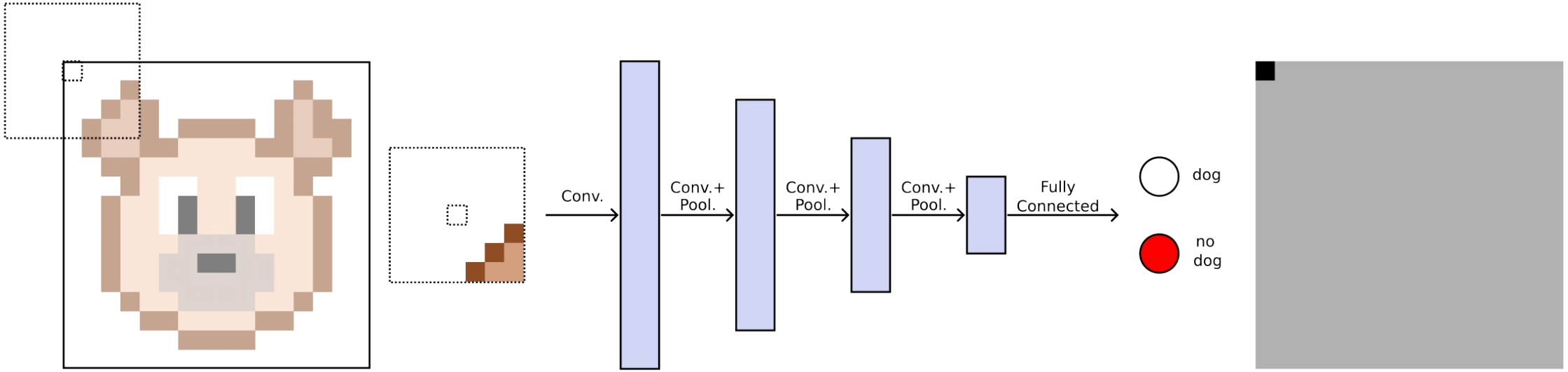
Motivation

- Convolutional Neural Networks (CNNs) have been proven to be good for image classification



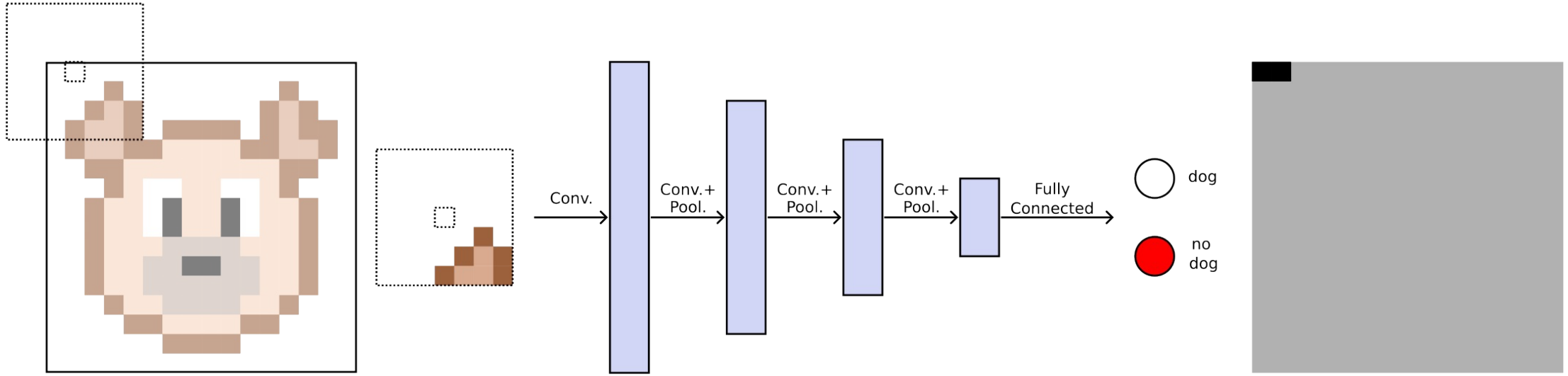
Motivation

- Convolutional Neural Networks (CNNs) have been proven to be good for image classification
- **Intuitive approach for image segmentation**: Use a CNN for classifying each individual pixel



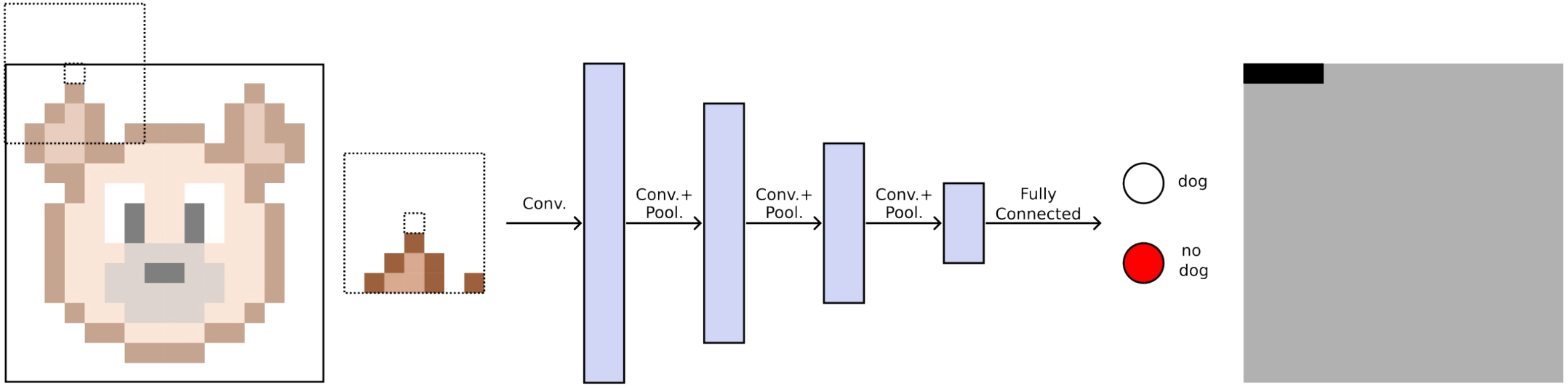
Motivation

- Convolutional Neural Networks (CNNs) have been proven to be good for image classification
- **Intuitive approach for image segmentation**: Use a CNN for classifying each individual pixel



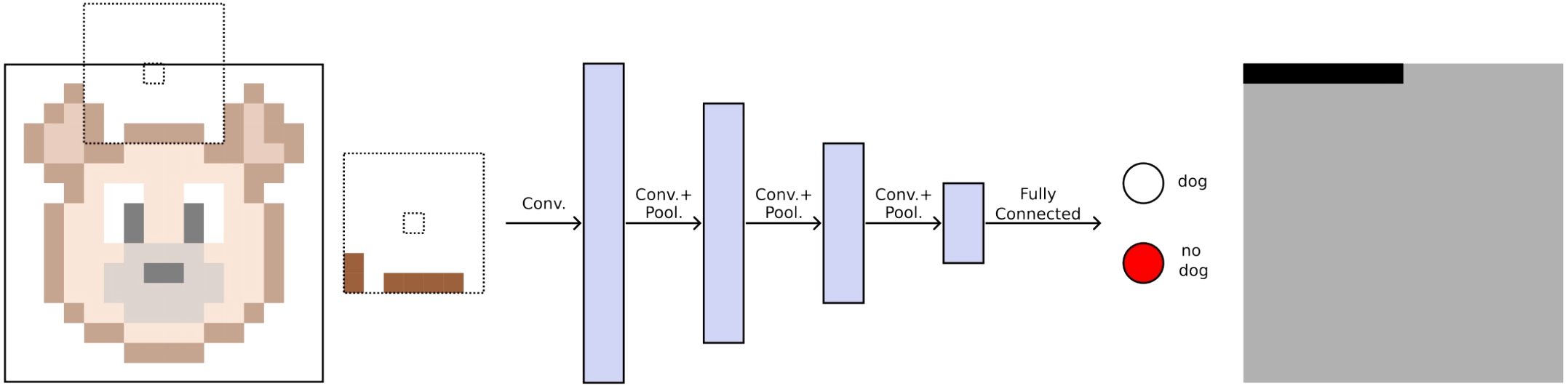
Motivation

- Convolutional Neural Networks (CNNs) have been proven to be good for image classification
- **Intuitive approach for image segmentation**: Use a CNN for classifying each individual pixel



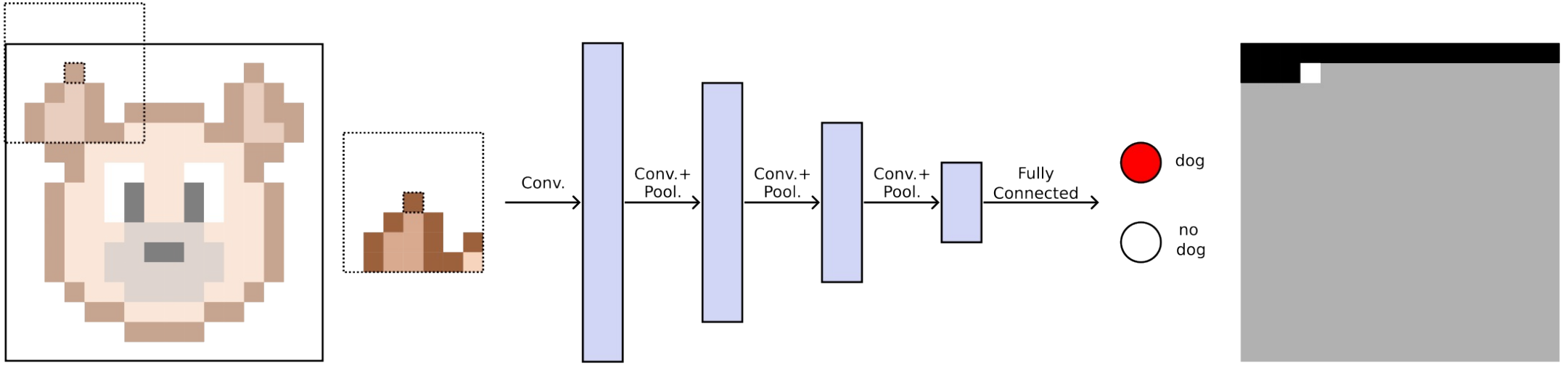
Motivation

- Convolutional Neural Networks (CNNs) have been proven to be good for image classification
- **Intuitive approach for image segmentation**: Use a CNN for classifying each individual pixel



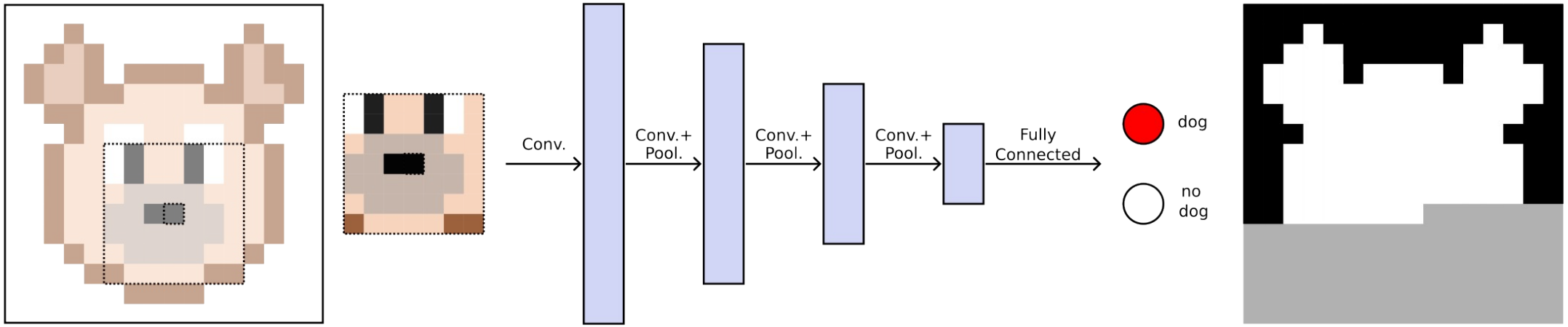
Motivation

- Convolutional Neural Networks (CNNs) have been proven to be good for image classification
- **Intuitive approach for image segmentation**: Use a CNN for classifying each individual pixel



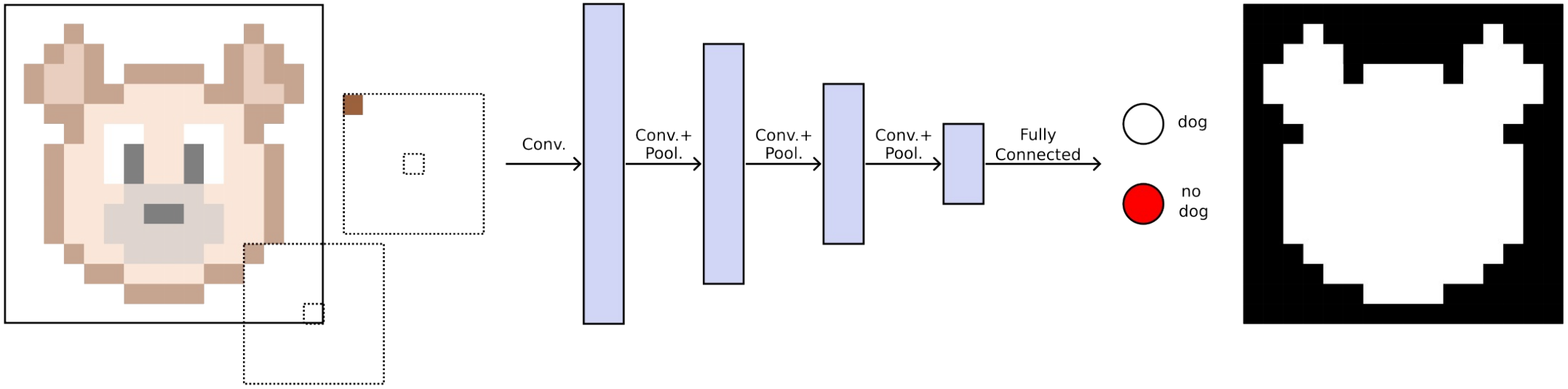
Motivation

- Convolutional Neural Networks (CNNs) have been proven to be good for image classification
- **Intuitive approach for image segmentation**: Use a CNN for classifying each individual pixel



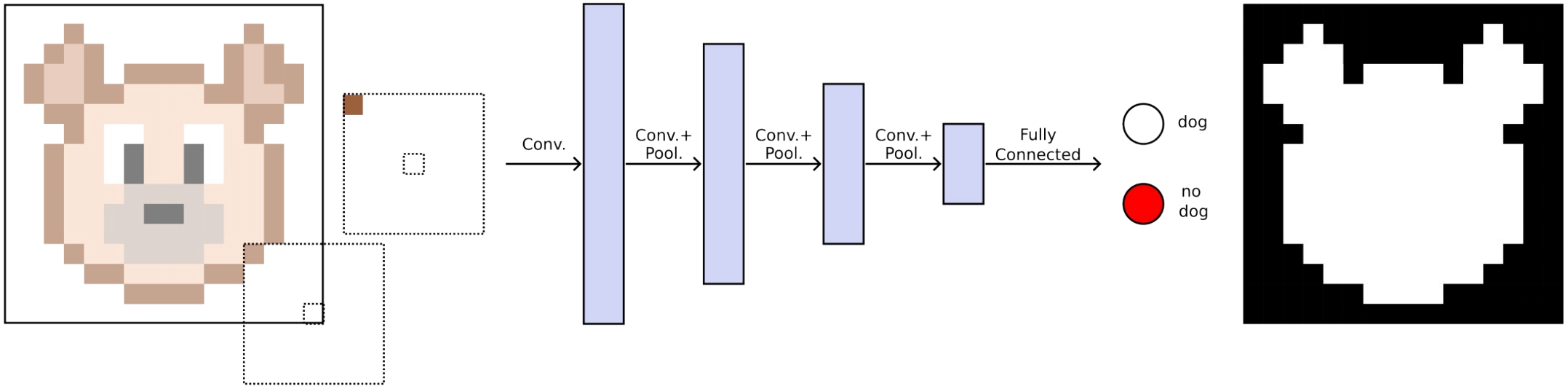
Motivation

- Convolutional Neural Networks (CNNs) have been proven to be good for image classification
- **Intuitive approach for image segmentation**: Use a CNN for classifying each individual pixel



Motivation

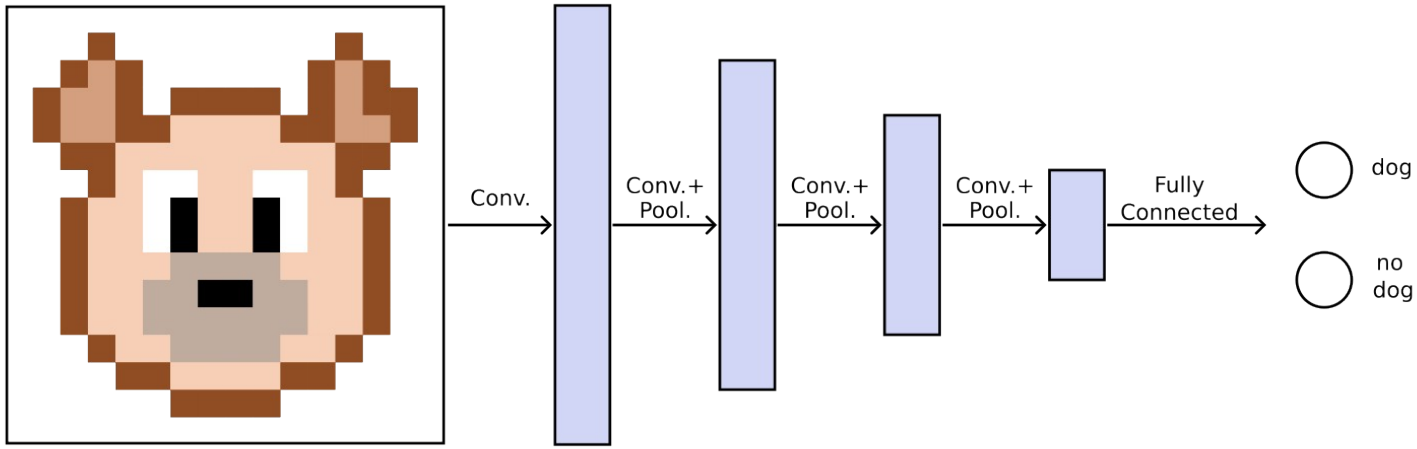
- Convolutional Neural Networks (CNNs) have been proven to be good for image classification
- **Intuitive approach for image segmentation**: Use a CNN for classifying each individual pixel

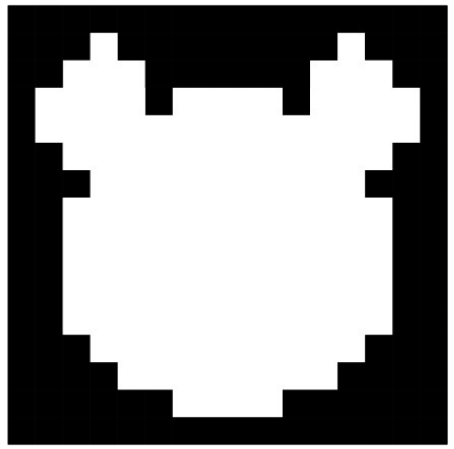


- Predicting each pixel individually becomes infeasible for larger images

Motivation

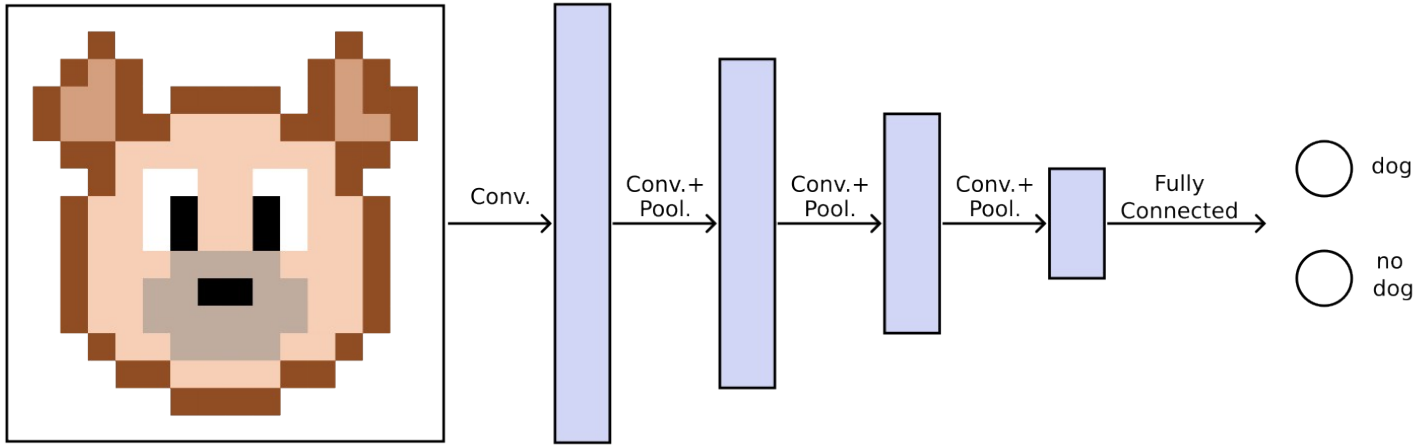
- **Alternative:** Predict the class of every pixel in a single forward pass

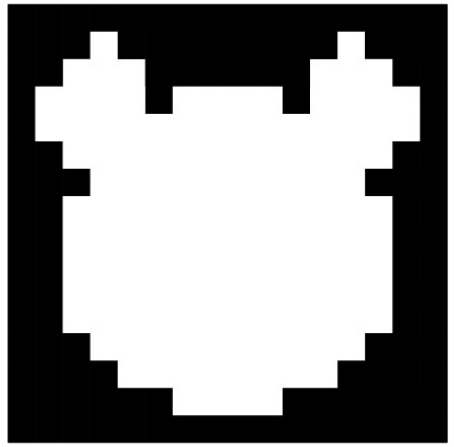




Motivation

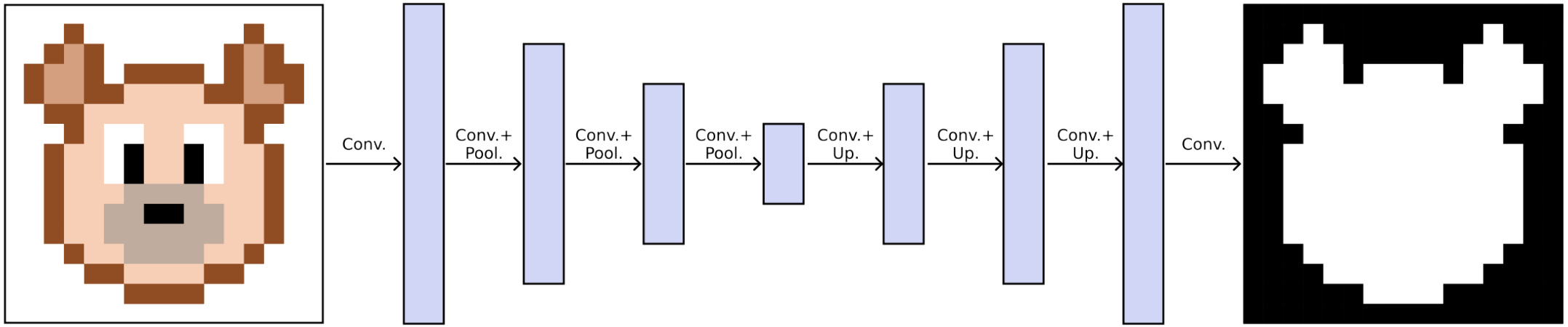
- **Alternative:** Predict the class of every pixel in a single forward pass
- **Requirement:** Output has to have the same dimensions as input





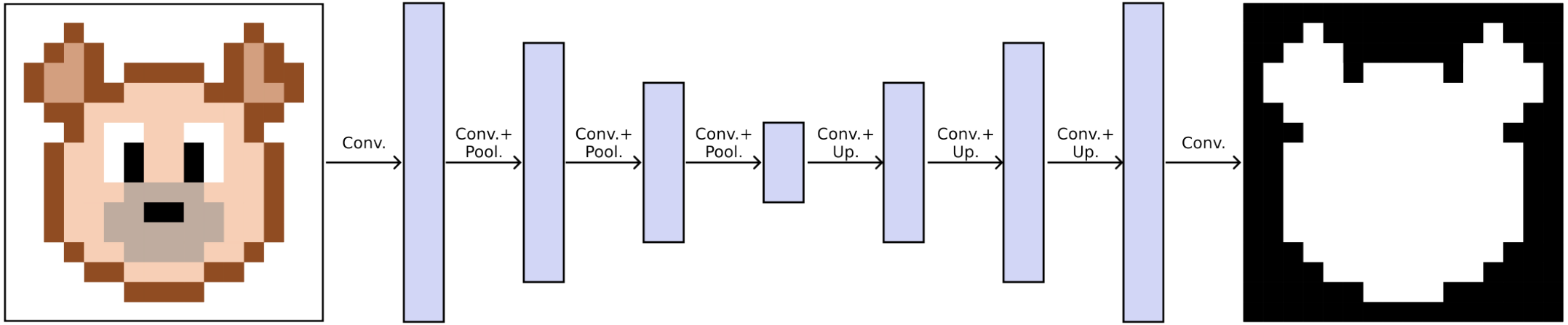
Motivation

- **Alternative:** Predict the class of every pixel in a single forward pass
- **Requirement:** Output has to have the same dimensions as input
  - ➔ Replace fully connected layers with upsampling layers



Motivation

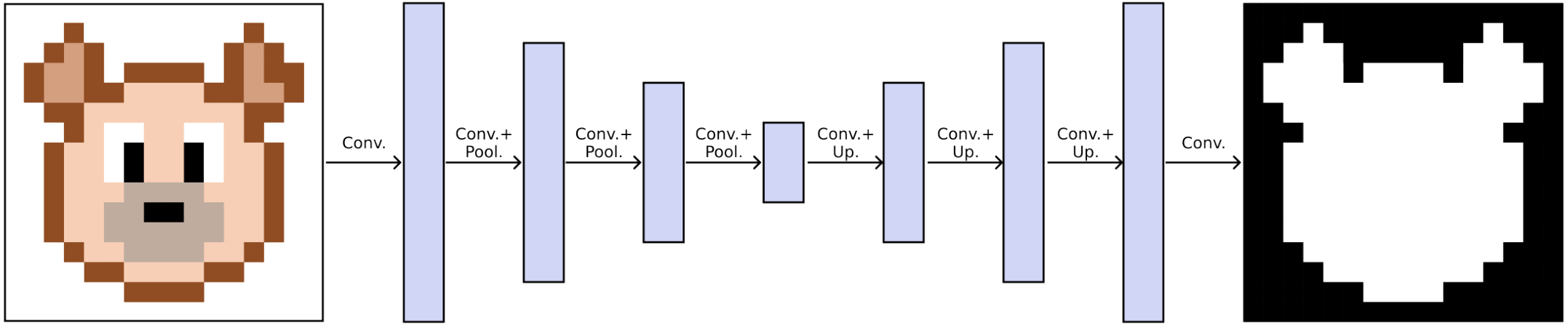
- **Alternative:** Predict the class of every pixel in a single forward pass
- **Requirement:** Output has to have the same dimensions as input
  - ➔ Replace fully connected layers with upsampling layers



Fully convolutional neural network

Motivation

- **Alternative:** Predict the class of every pixel in a single forward pass
- **Requirement:** Output has to have the same dimensions as input
  - ➔ Replace fully connected layers with upsampling layers



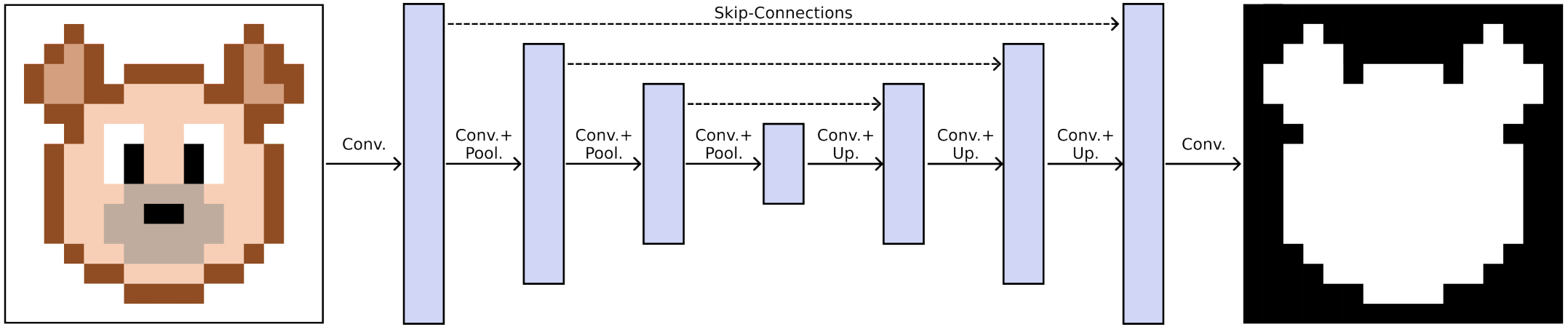
Fully convolutional neural network

Problems:

- Downsampling causes loss of spatial information
- Deep networks suffer from vanishing/exploding gradients

Motivation

- **Alternative:** Predict the class of every pixel in a single forward pass
- **Requirement:** Output has to have the same dimensions as input
  - ➔ Replace fully connected layers with upsampling layers



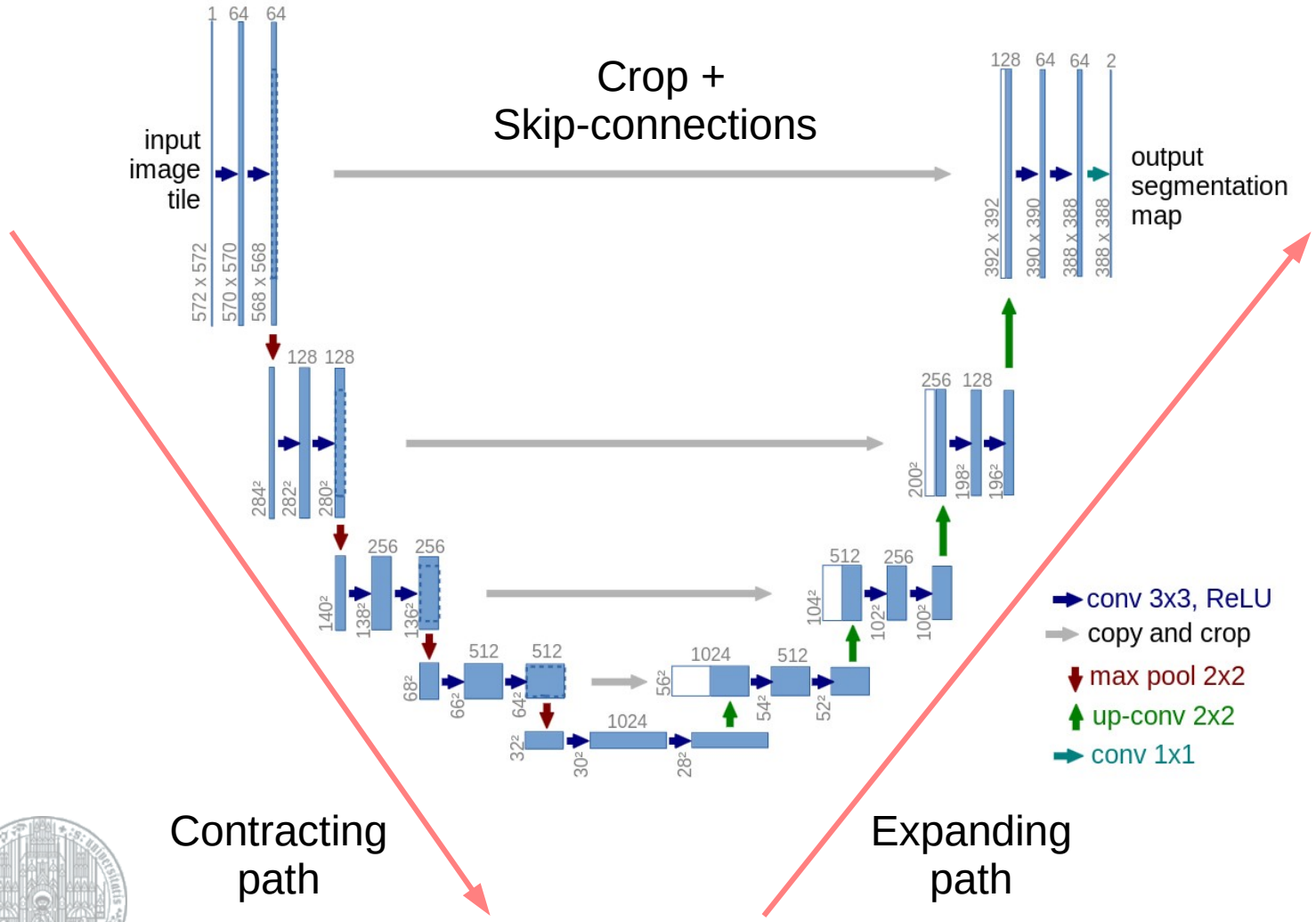
Fully convolutional neural network

Problems:

- Downsampling causes loss of spatial information
- Deep networks suffer from vanishing/exploding gradients

} Skip-connections help mitigate these problems

U-Net Architecture



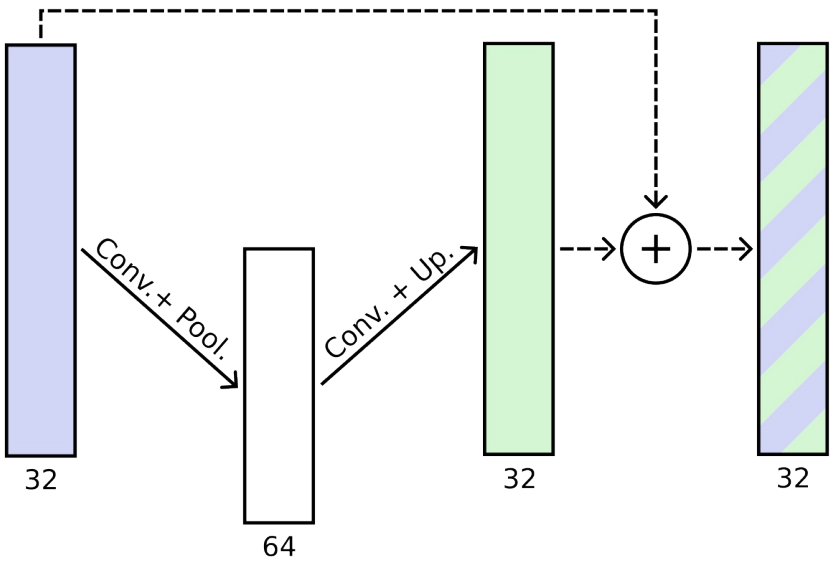
- **U-Net** architecture frequently used in many domains (e.g., image segmentation)
- **Contracting path** extracts high level features
- **Expanding path** restores the input resolution
- **Skip-connections** are used to retain spatial information

Ronneberger et al., MICCAI 2015

Skip-Connections and Image Cropping

- Two options for implementing skip-connections

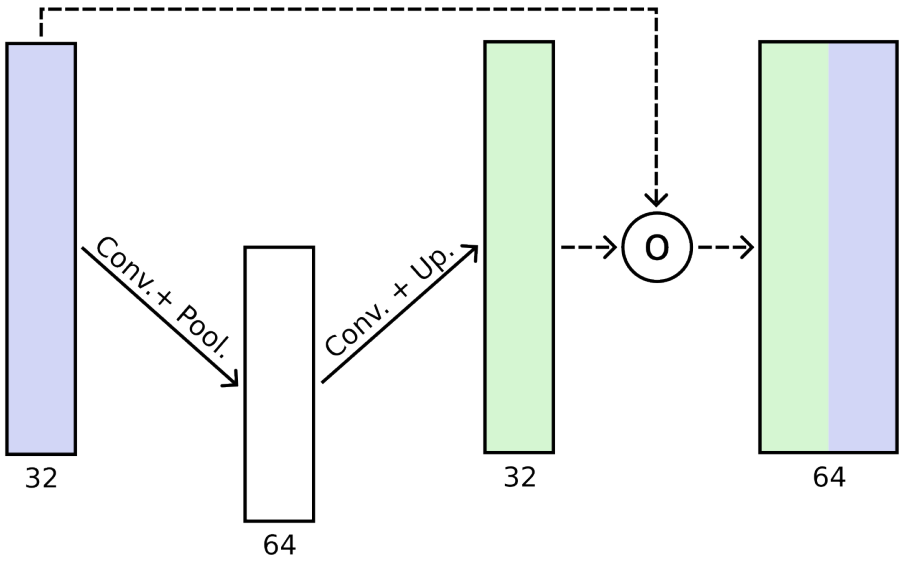
Addition



✔ No additional parameters needed
- ➔ Computationally efficient
- ➔ Less overfitting

✗ Loss/Mixing of information

Concatenation

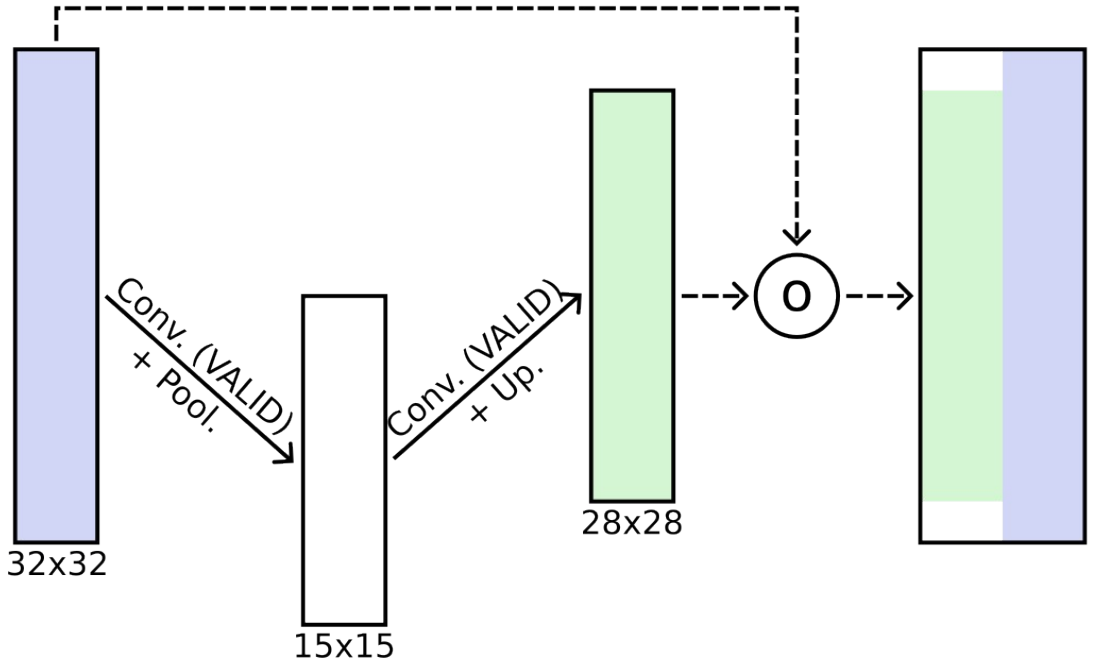


✔ Information is preserved

✗ More parameters needed
- ➔ Computationally less efficient
- ➔ Overfitting

Skip-Connections and Image Cropping

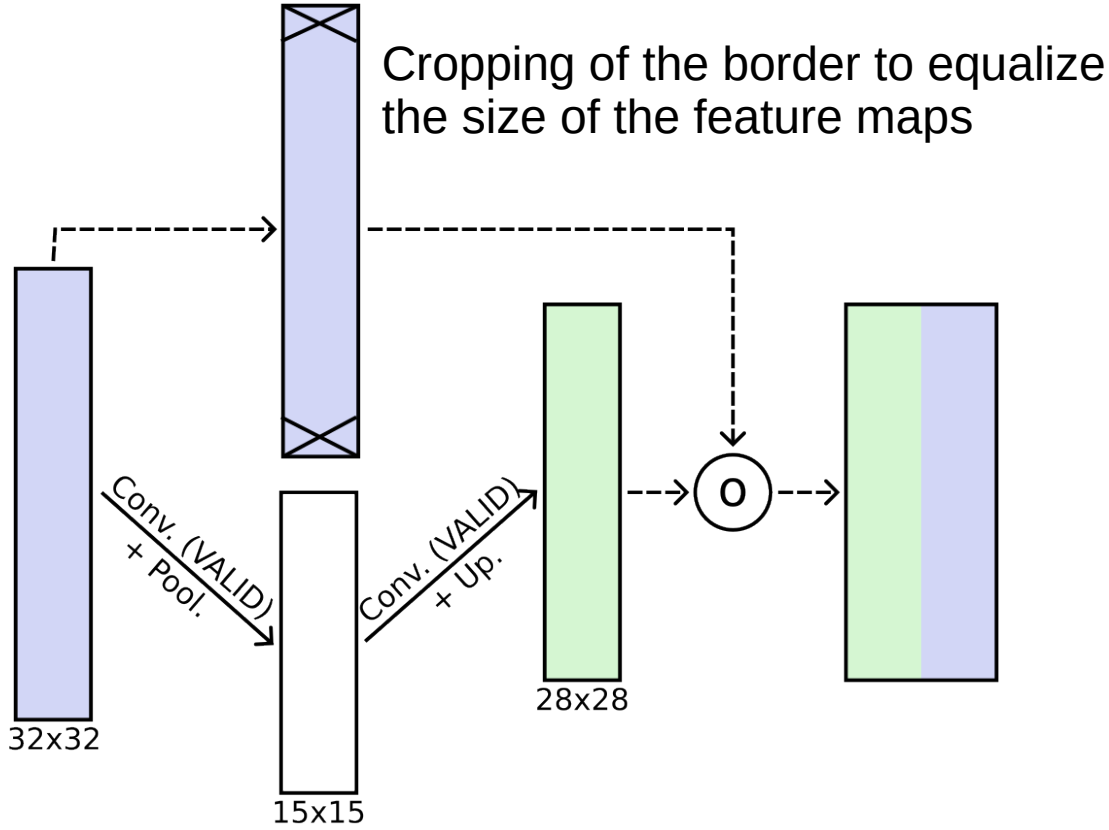
- Two options for implementing skip-connections
  - Addition
  - Concatenation
- When using convolution without padding (VALID), cropping is required



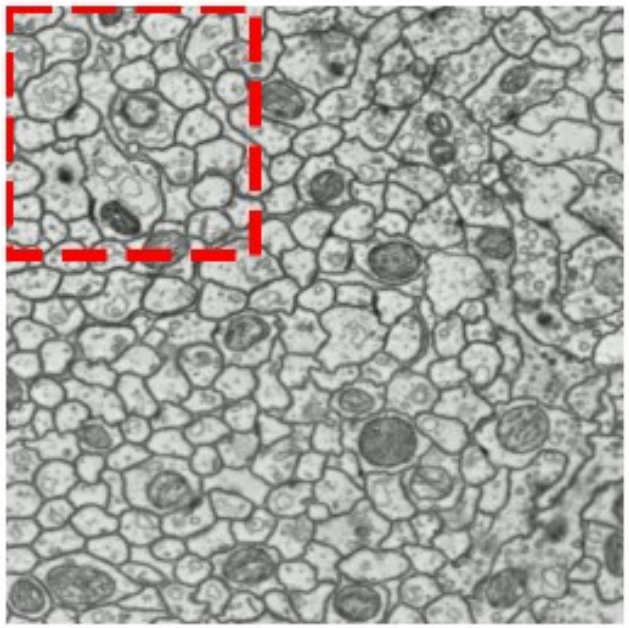
Feature maps have different spatial dimensions making addition/concatenation impossible

Skip-Connections and Image Cropping

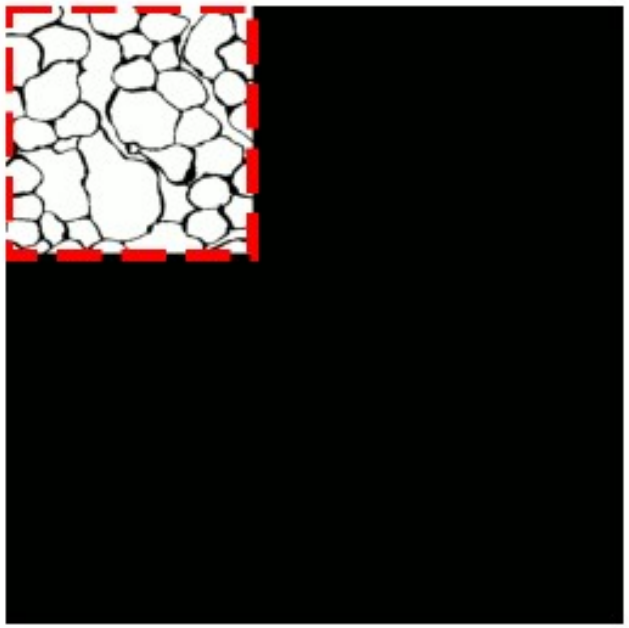
- Two options for implementing skip-connections
  - Addition
  - Concatenation
- When using convolution without padding (VALID), cropping is required



U-Net Training



Input image

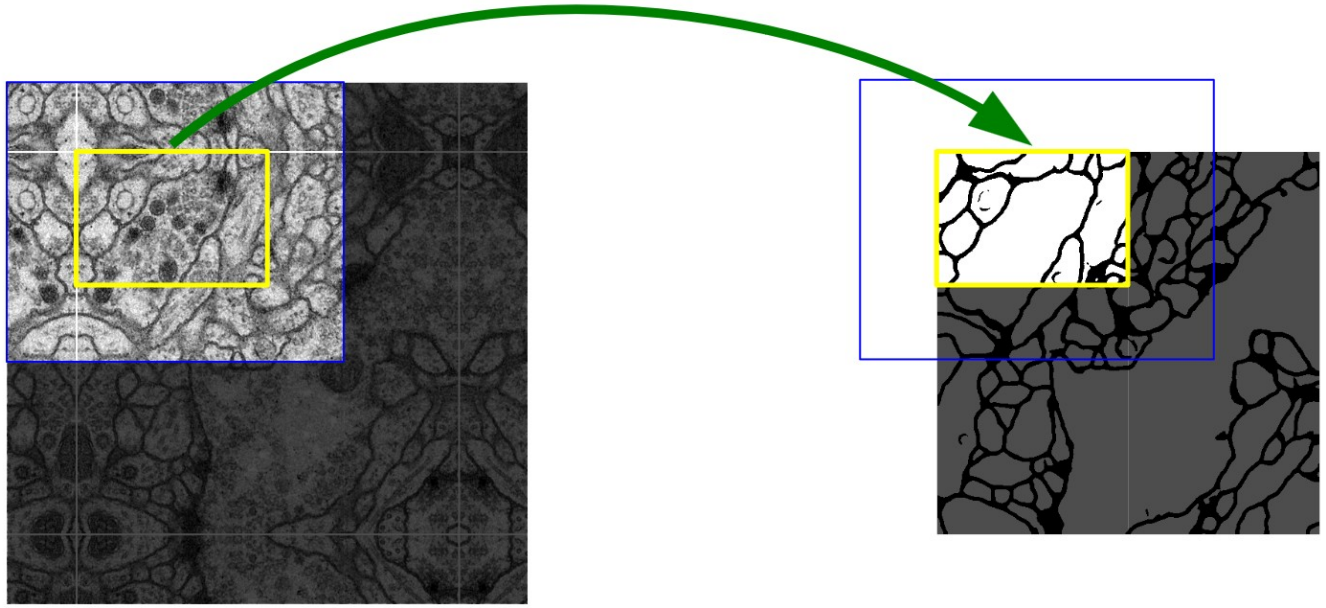


Prediction

- To handle arbitrary image sizes, prediction is performed using a sliding window approach with overlapping patches

CREMI, MICCAI 2016 Challenge

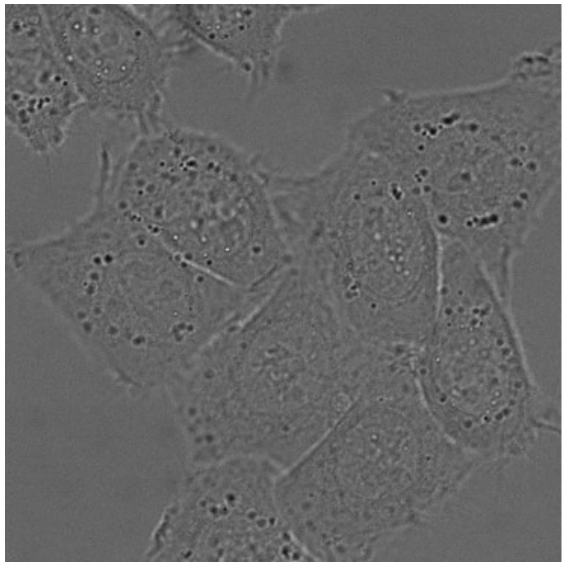
U-Net Training



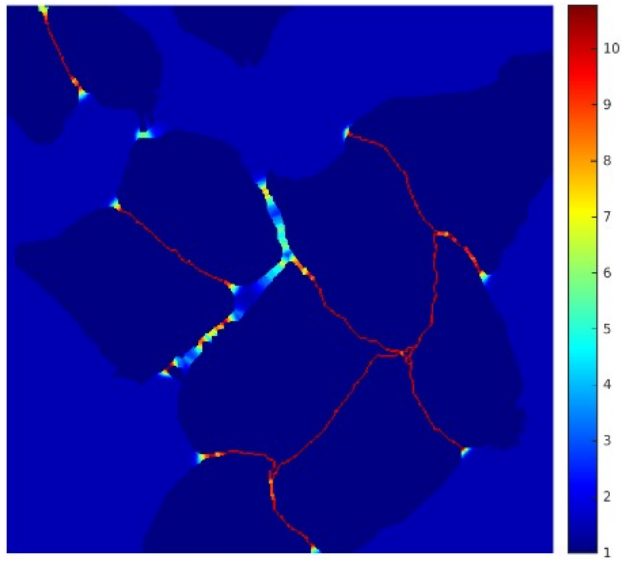
- To handle arbitrary image sizes, prediction is performed using a sliding window approach with overlapping patches
- Prediction of the yellow region uses context of the blue window
- Missing context at the borders is solved by extrapolation with mirroring

Ronneberger et al., MICCAI 2015

U-Net Training



Input image



Weight Map

$$w(\mathbf{x}) = w_c(\mathbf{x}) + w_0 \cdot \exp\left(-\frac{(d_1(\mathbf{x}) + d_2(\mathbf{x}))^2}{2\sigma^2}\right)$$

d_1 = distance to the closest cell
d_2 = distance to the second closest cell

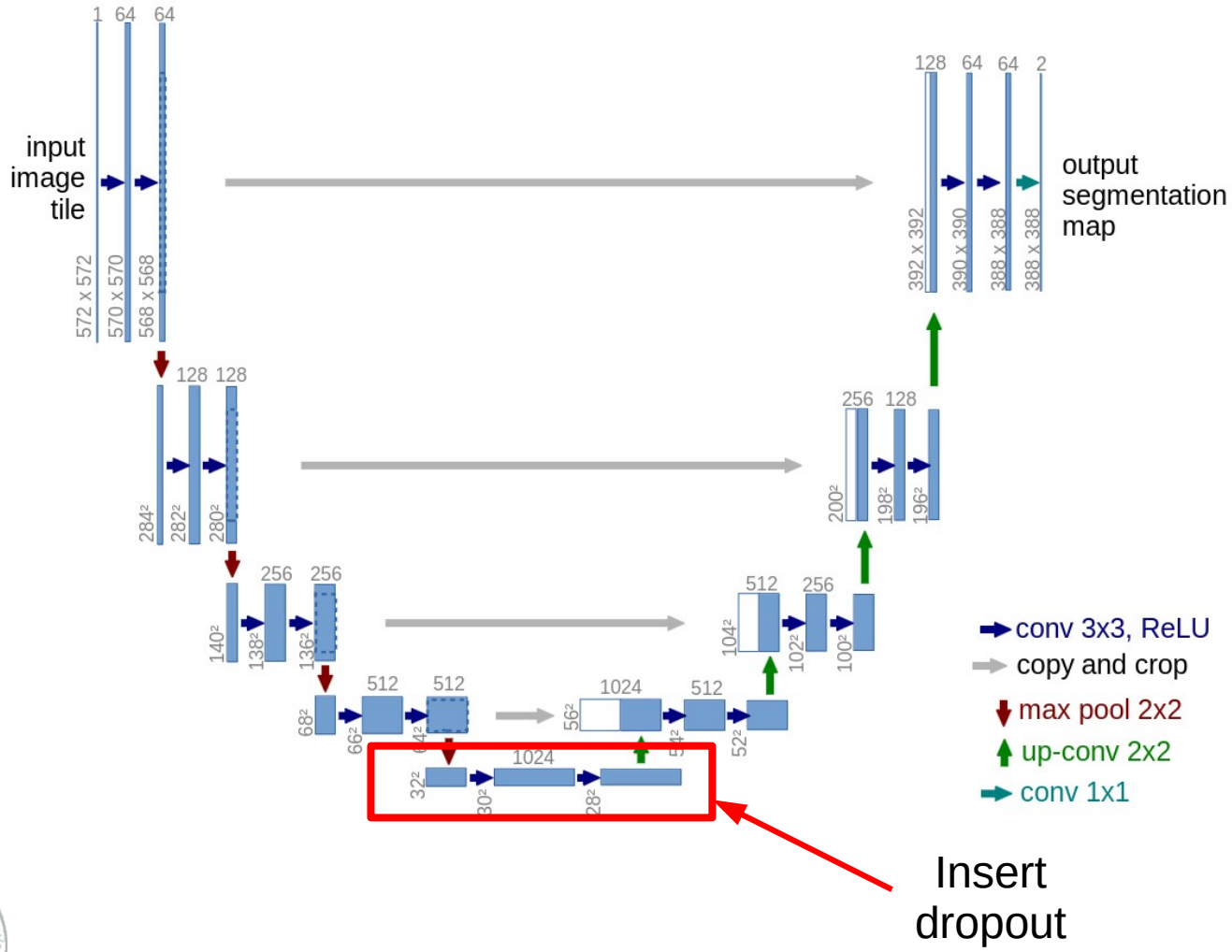
- Network is trained using weighted Cross-Entropy-Loss:

$$L_{CE} = -\frac{1}{N}\sum_{i}^{N} w_i \cdot \hat{y}_i \log(y_i)$$

- Weight function handles class imbalance
- Weight function forces the network to separate touching cells

Ronneberger et al., MICCAI 2015

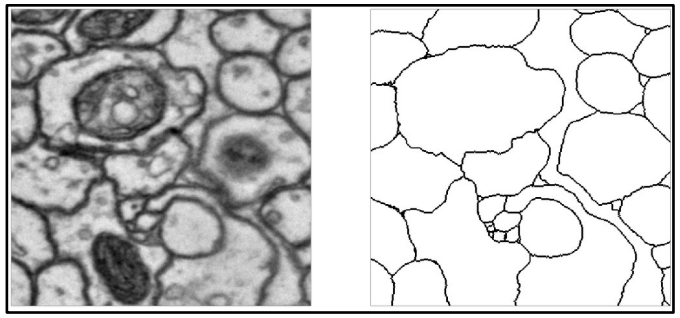
Training with Small Datasets



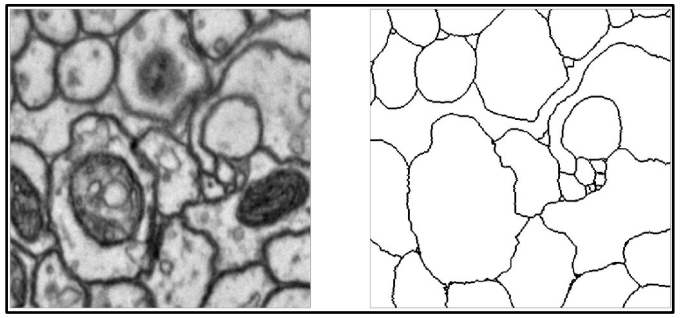
- Originally, dropout was used at the end of the contracting path

Ronneberger et al., MICCAI 2015

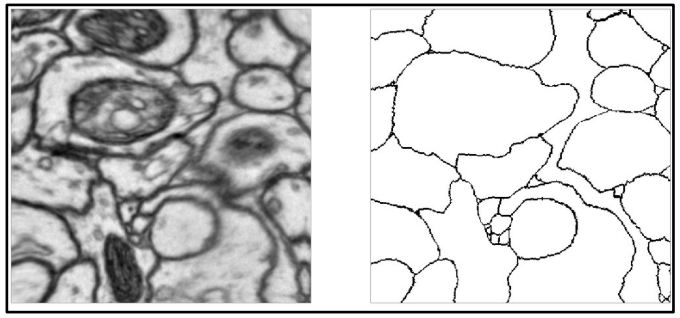
Training with Small Datasets



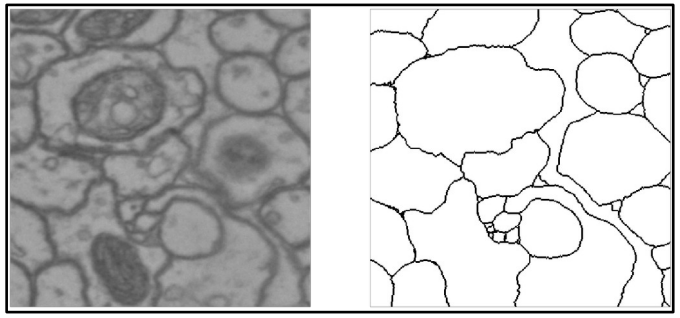
Original image



Random rotate



Random elastic deformations

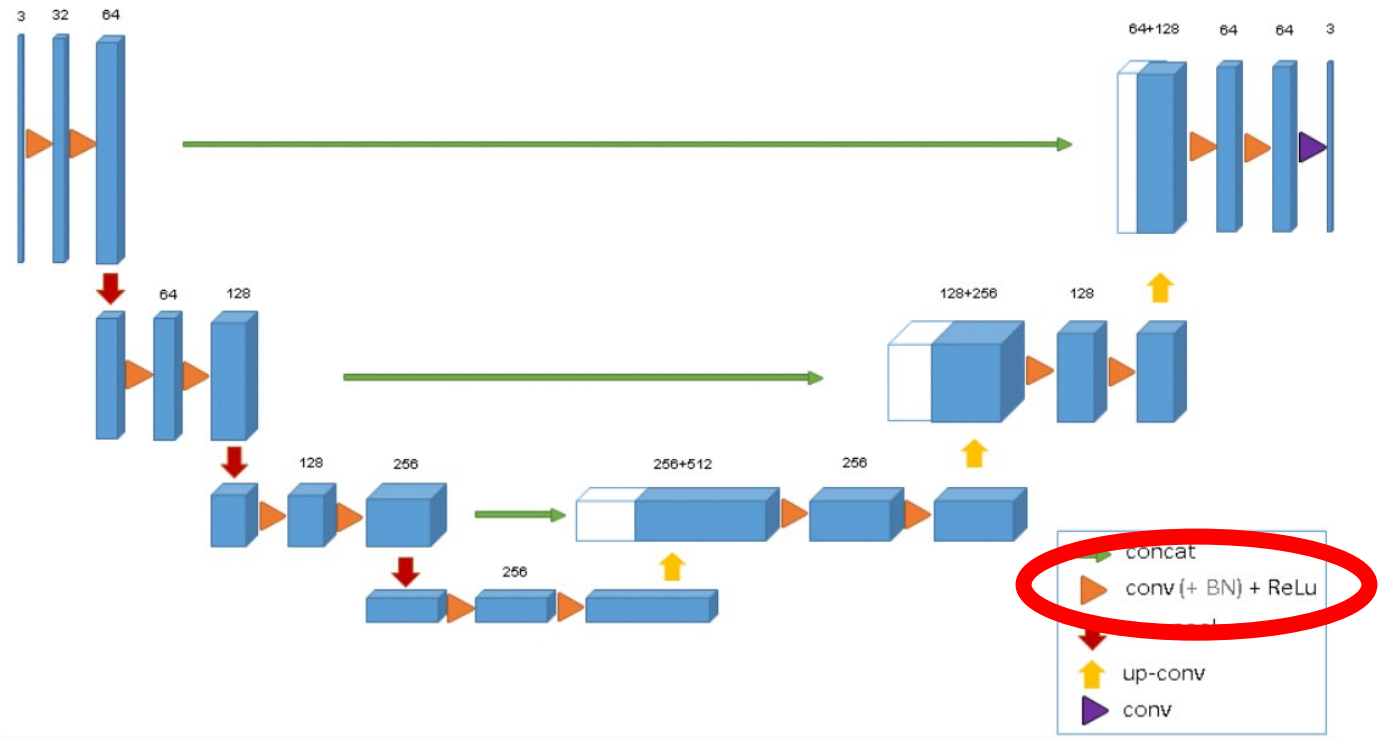


Random contrast + brightness

- Originally, dropout was used at the end of the contracting path
- Artificially increase training data by performing data augmentation, e.g.,
  - Rotation
  - Elastic deformations
  - Image intensity variations

CREMI, MICCAI 2016 Challenge

Training with Small Datasets



- Originally, dropout was used at the end of the contracting path
- Artificially increase training data by performing data augmentation, e.g.,
  - Rotation
  - Elastic deformations
  - Image intensity variations
- Nowadays, normalization layers (e.g., batch normalization) are common practice

Çiçek et al., MICCAI 2016

Pixel-Wise Performance Measures

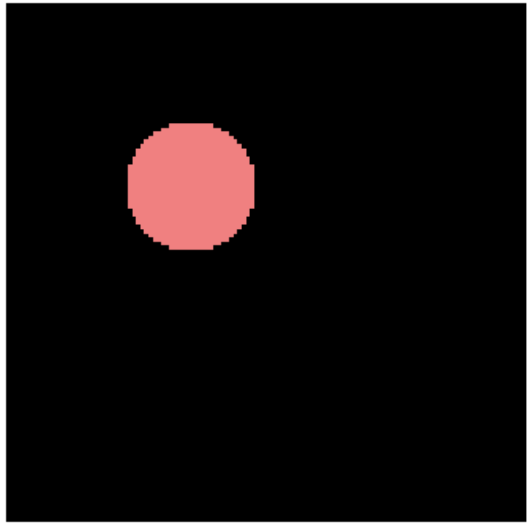
- Accuracy can be used for evaluating segmentation results

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$

Pixel-Wise Performance Measures

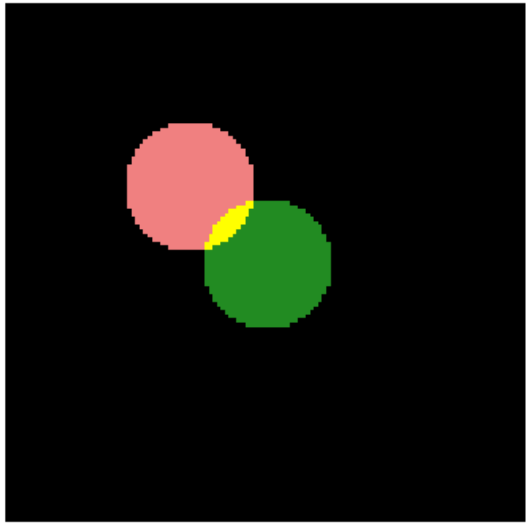
- Accuracy can be used for evaluating segmentation results

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$



Prediction

GT

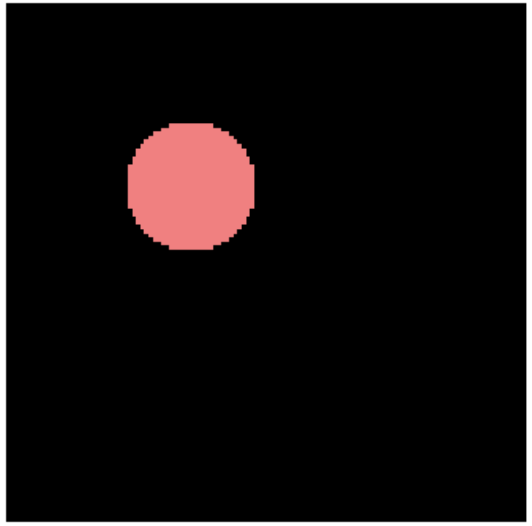


Overlay

Pixel-Wise Performance Measures

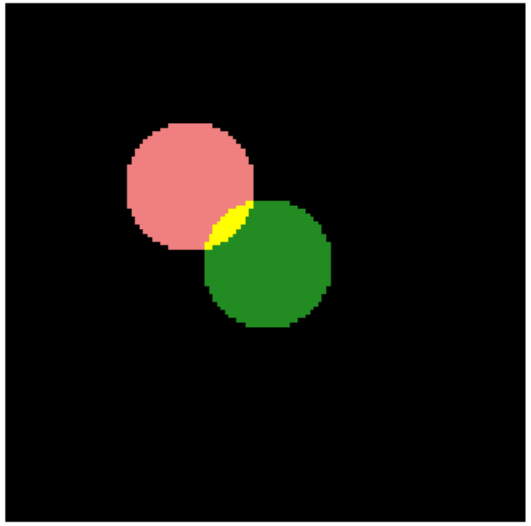
- Accuracy can be used for evaluating segmentation results

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$



Prediction

GT



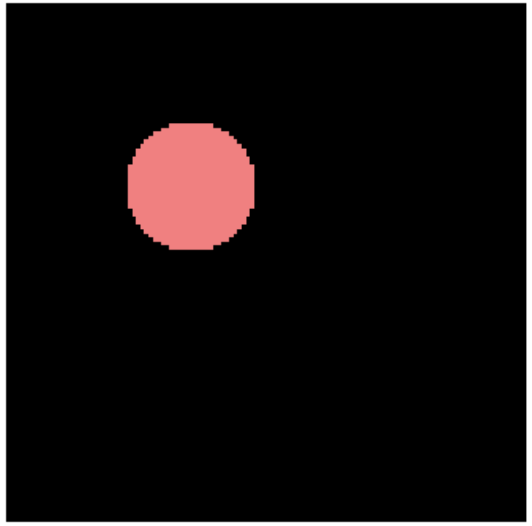
Overlay

$$\text{Accuracy} = 0.91$$

Pixel-Wise Performance Measures

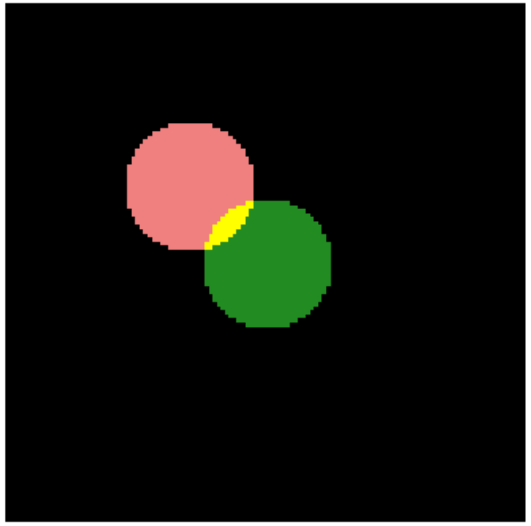
- Accuracy can be used for evaluating segmentation results

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$



Prediction

GT



Overlay

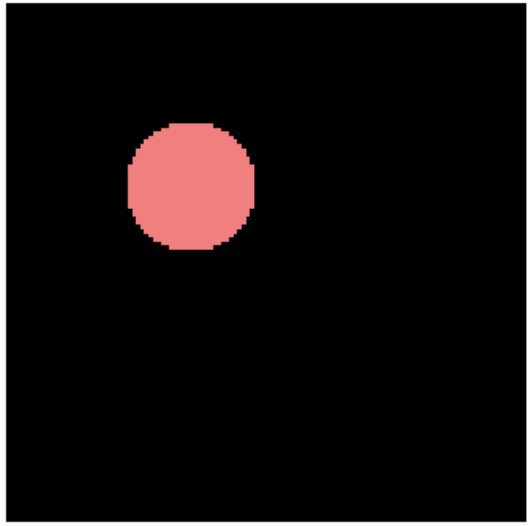
$$\text{Accuracy} = 0.91$$

Q: Why do we get a good accuracy despite poor segmentation results?

Pixel-Wise Performance Measures

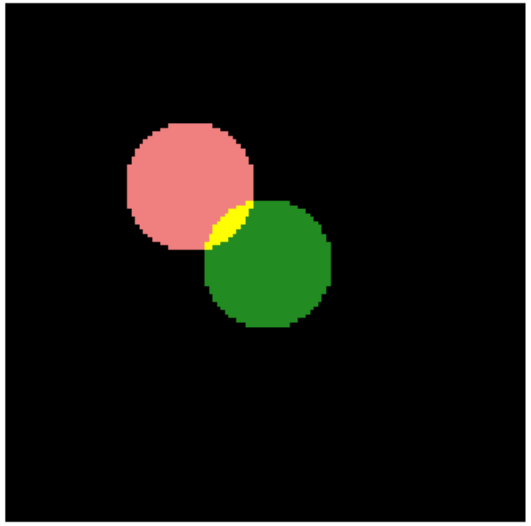
- Accuracy can be used for evaluating segmentation results

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$



Prediction

GT



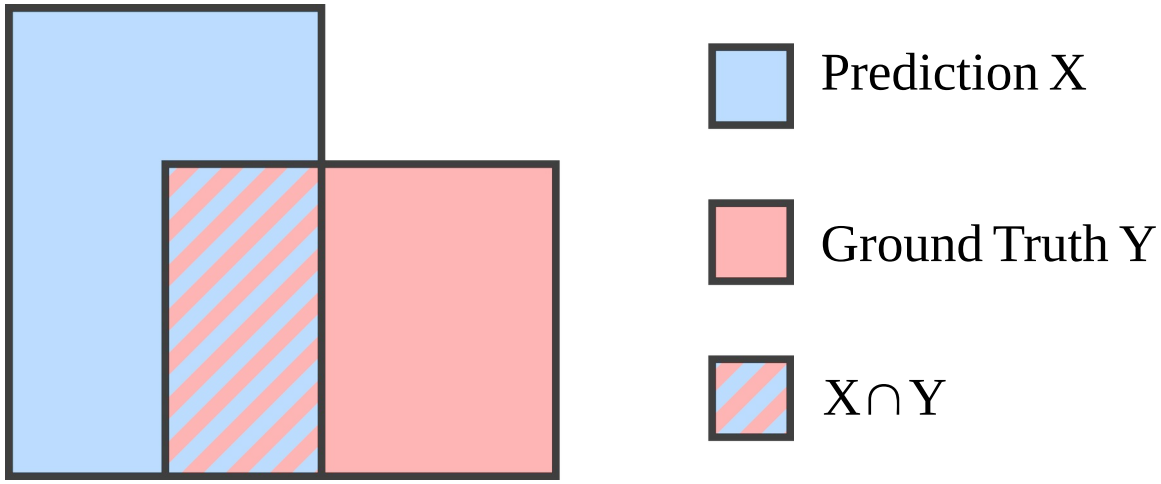
Overlay

$$\text{Accuracy} = 0.91$$

Q: Why do we get a good accuracy despite poor segmentation results?

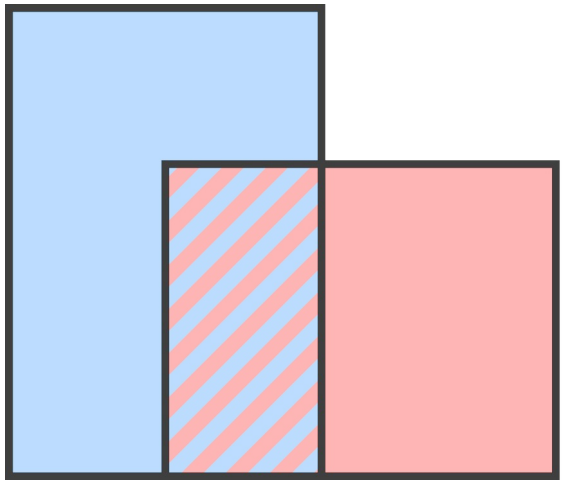
A: Class imbalance, i.e. correctly predicted background class dominates the accuracy!

Object-Wise Performance Measures



Adapted from Reinke et al. 2021

Object-Wise Performance Measures



Prediction X

Ground Truth Y

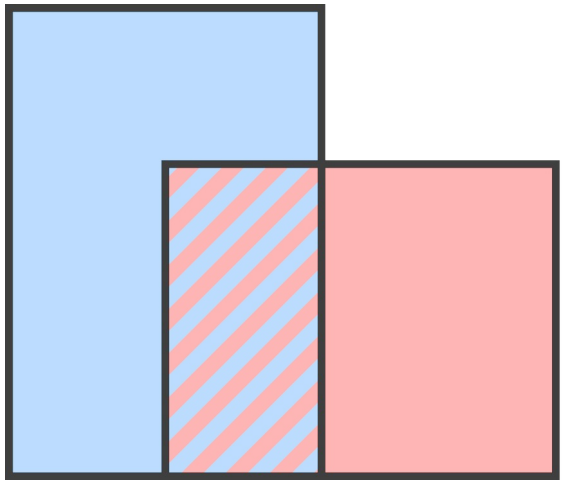
$X \cap Y$

$$IoU = \frac{\square}{\square + \square - \square}$$

$$Dice = \frac{2 \times \square}{\square + \square}$$

Adapted from Reinke et al. 2021

Object-Wise Performance Measures



Prediction X

Ground Truth Y

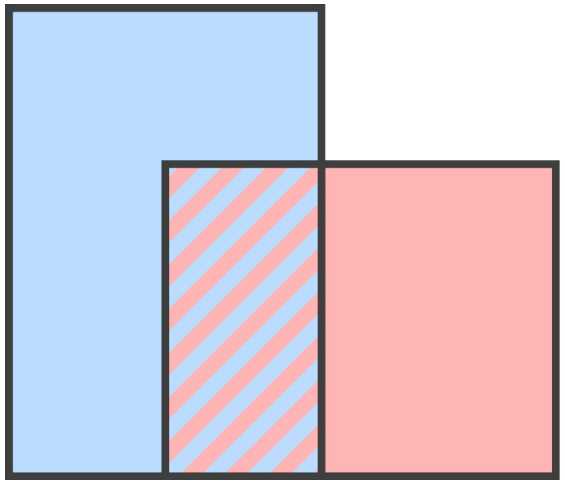
$X \cap Y$

$$IoU = \frac{\square}{\square + \square - \square} = \frac{|X \cap Y|}{|X \cup Y|}$$

$$Dice = \frac{2 \times \square}{\square + \square} = \frac{2 \cdot |X \cap Y|}{|X| + |Y|}$$

Adapted from Reinke et al. 2021

Object-Wise Performance Measures



Prediction X

Ground Truth Y

$X \cap Y$

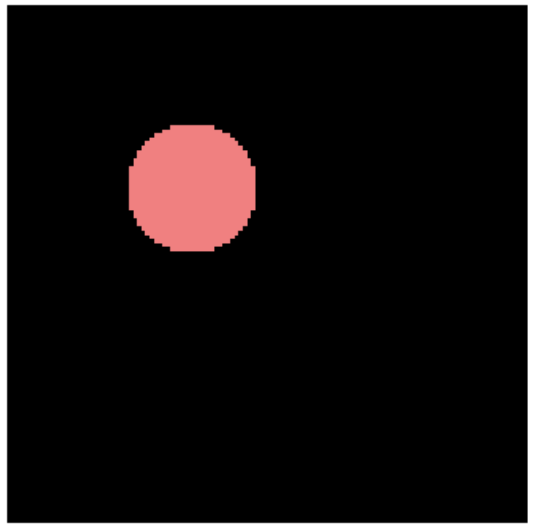
$$IoU = \frac{|X \cap Y|}{|X \cup Y|}$$

$$Dice = \frac{2 \cdot |X \cap Y|}{|X| + |Y|}$$

- IoU is equivalent to Jaccard Index
- Dice is equivalent to F1-Score

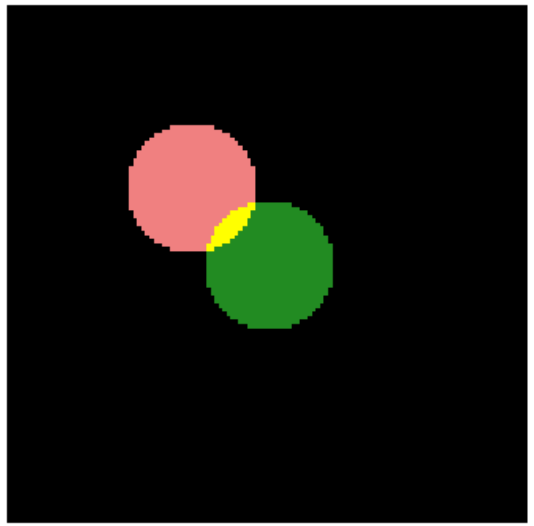
Adapted from Reinke et al. 2021

Performance Measures



Prediction

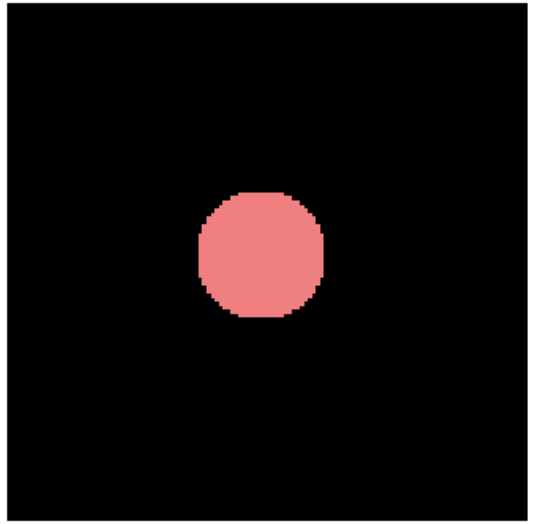
GT



Overlay

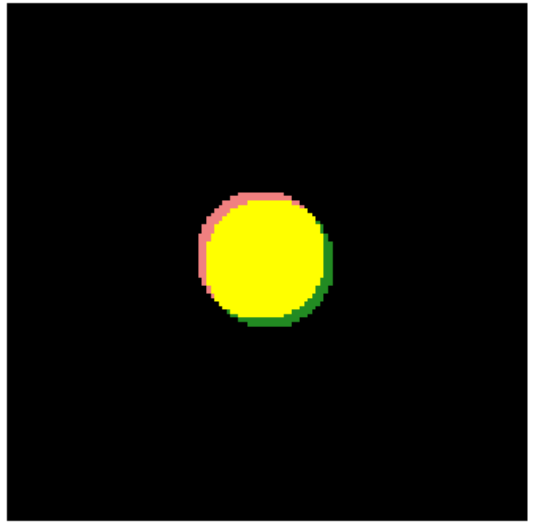
Accuracy	Dice	IoU
0.91	0.08	0.04

Performance Measures



Prediction

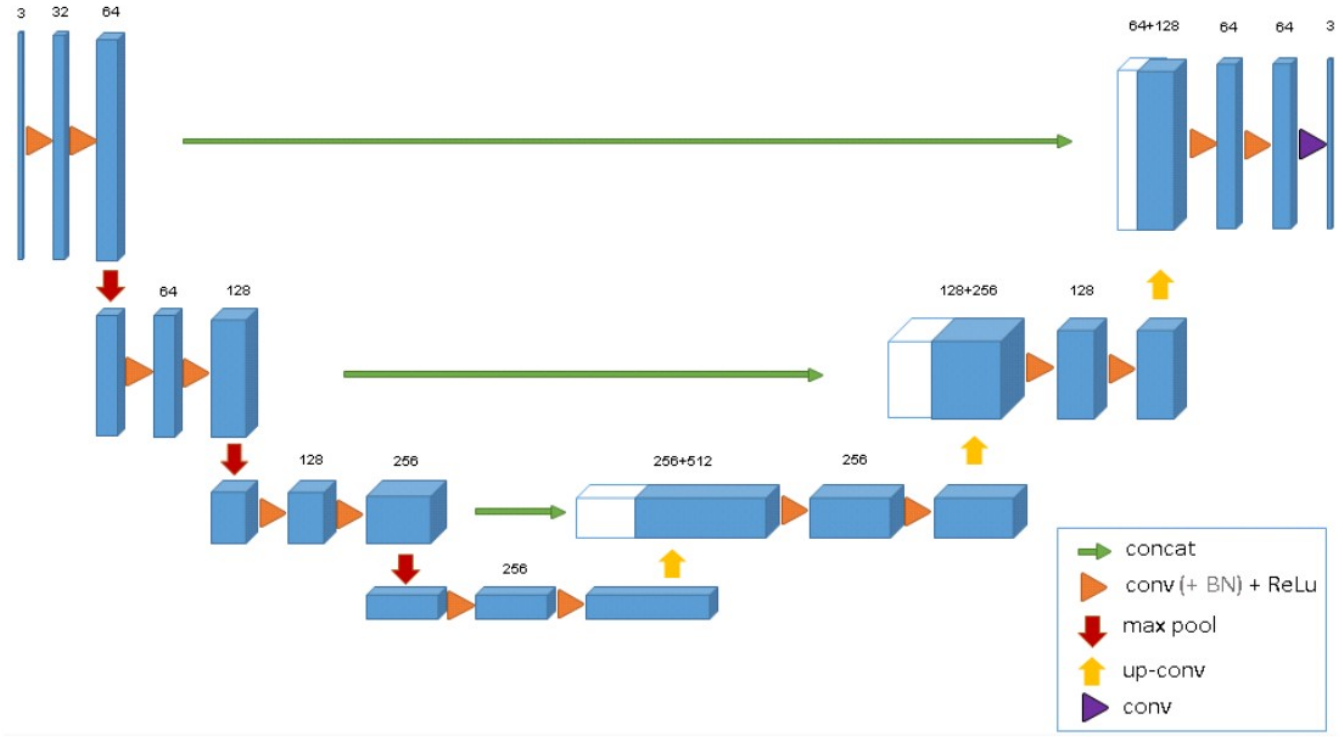
GT



Overlay

Accuracy	Dice	IoU
0.99	0.89	0.80

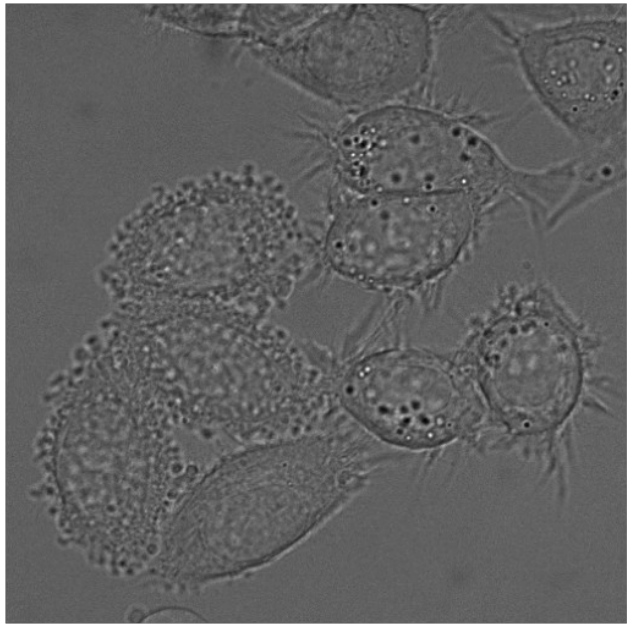
U-Net Extension to 3D



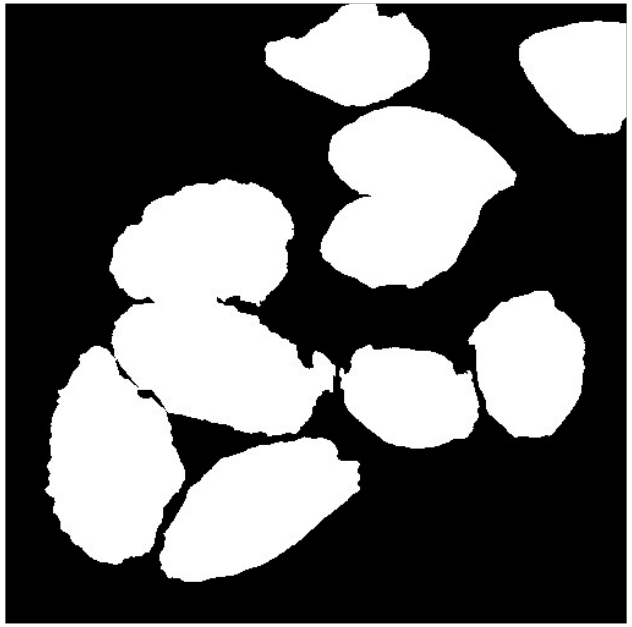
- 3D U-Net architecture for volumetric image segmentation
- General architecture remains the same
- Replace 2D operations with 3D operations (convolution, pooling, etc.)
- Add batch normalization (BN) in each layer for faster training

Çiçek et al., MICCAI 2016

Practical Exercises



Original image



Ground truth

- Implementation of the original 2D U-Net
- Application to HeLa cells
- Adapting to today's standards
- Implementation of data augmentation
- Testing different model configurations

Maška et al., Nature Methods 2023
celltrackingchallenge.net

References

- Stringer, C et al. "Cellpose: a generalist algorithm for cellular segmentation." Nature Methods 18 (2021): 100-106
- Rombach, R et al. "High-resolution image synthesis with latent diffusion models." CVRP 2022
- Ronneberger, O et al. "U-net: Convolutional networks for biomedical image segmentation." MICCAI 2015
- Çiçek, Ö et al. "3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation", MICCAI 2016
- Reinke, A et al. "Common limitations of image processing metrics: A picture story." arXiv:2104.05642 (2021)
- Maška, M et al. “The Cell Tracking Challenge: 10 years of objective benchmarking.” Nature Methods 20 (2023), 1010–1020
- CREMI, MICCAI Challenge on Circuit Reconstruction from Electron Microscopy Images, 2016, https://cremi.org/

Questions ?

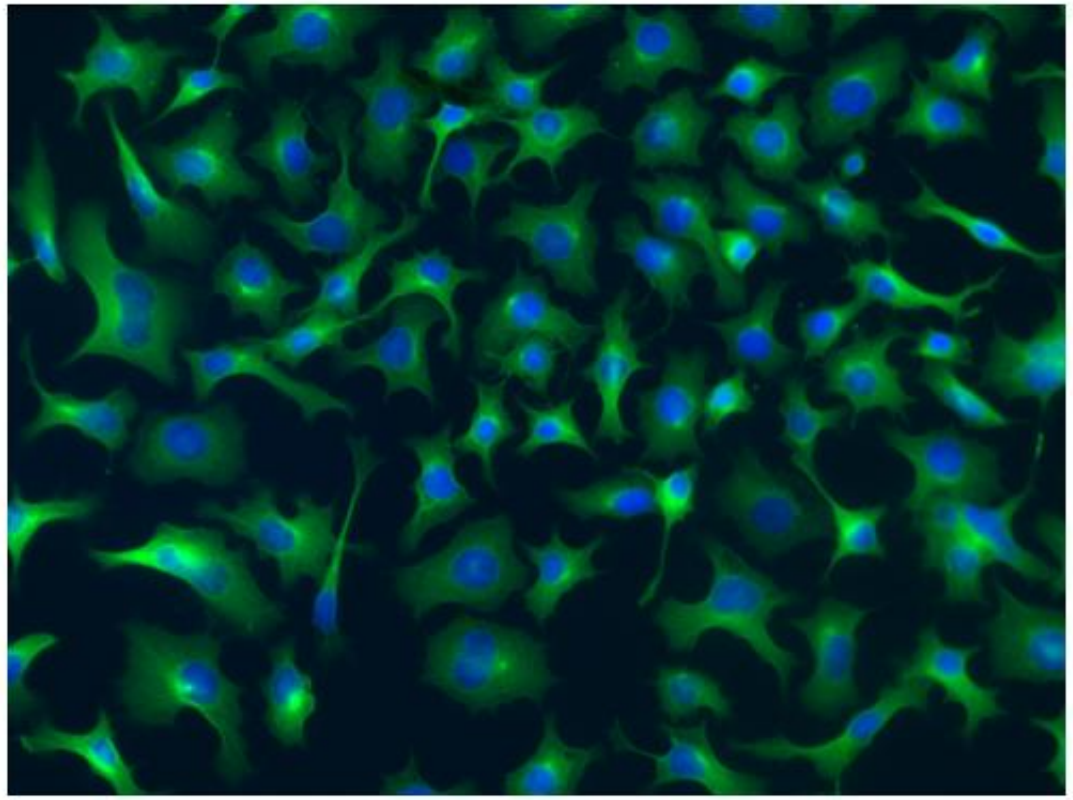
Deep Learning for Image Segmentation: Cellpose

Moritz Kunzmann

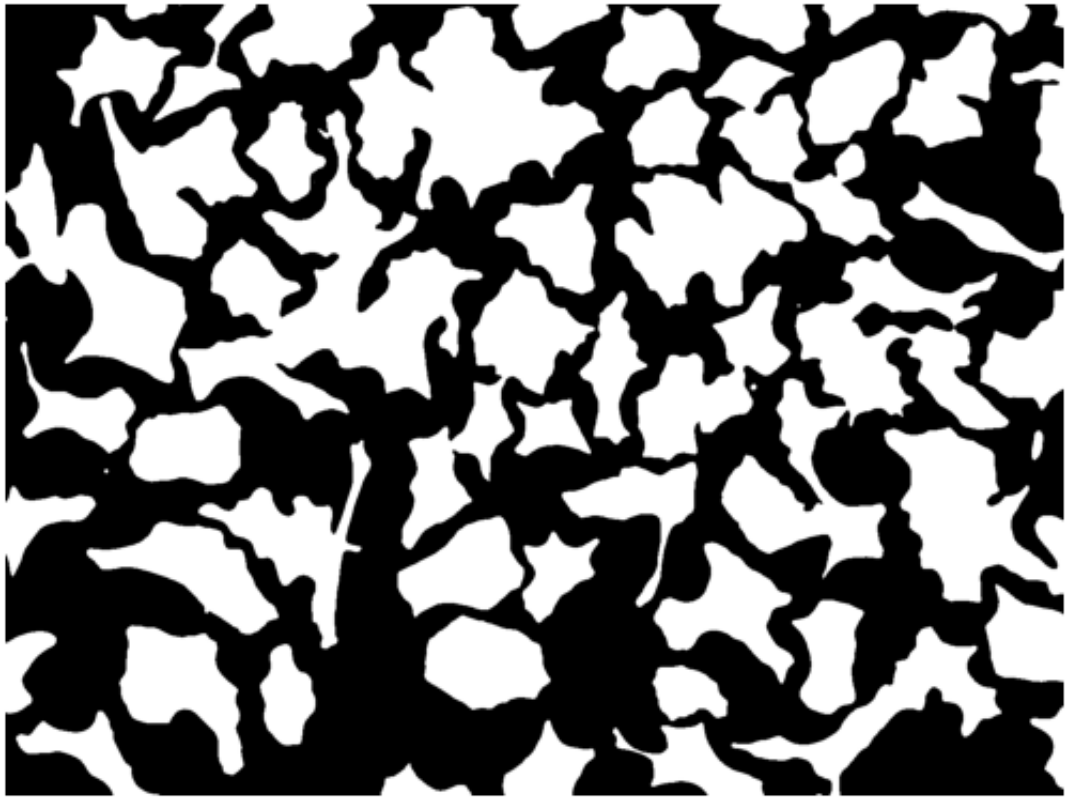
Biomedical Computer Vision Group (BMCV)
BioQuant, IPMB, Heidelberg University

Cell Instance Segmentation

Separating touching cells is challenging



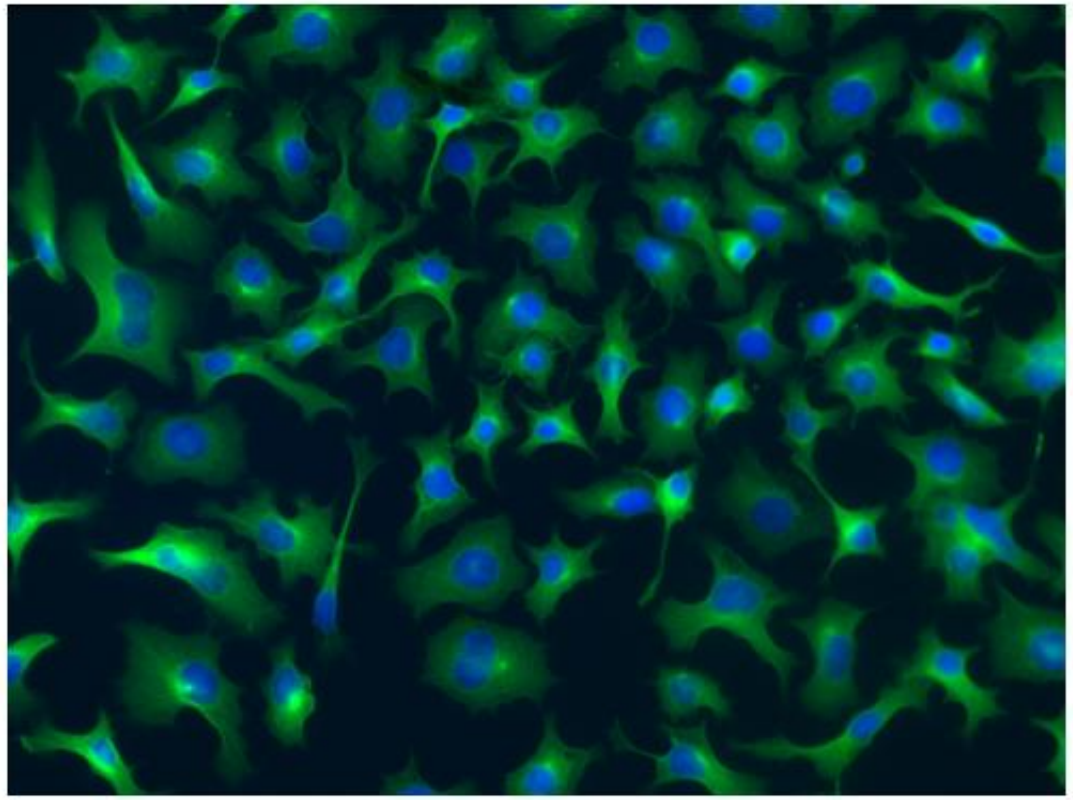
Fluorescence microscopy image
(Actin + Nuclei)



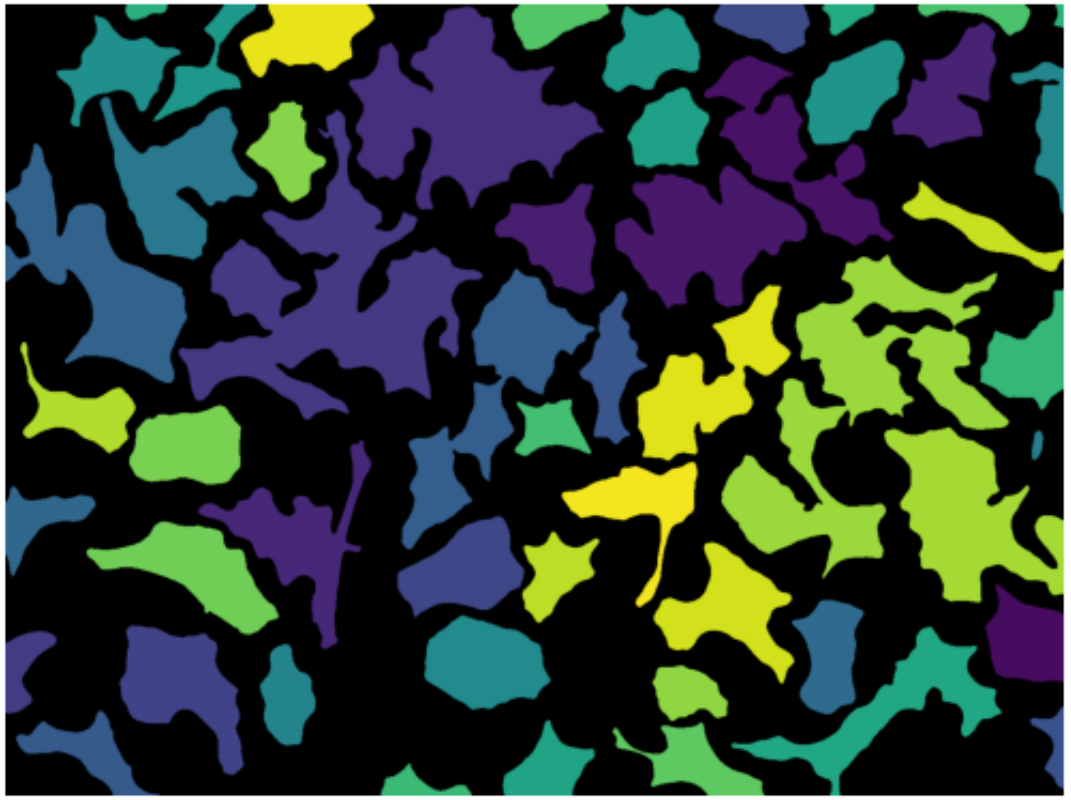
Binary segmentation

Cell Instance Segmentation

Separating touching cells is challenging

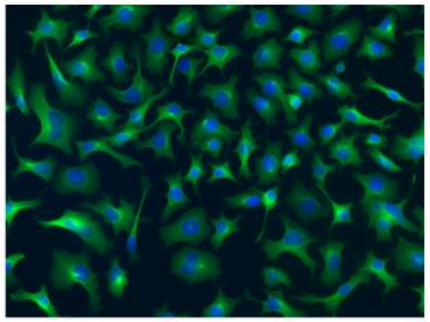


Fluorescence microscopy image
(Actin + Nuclei)

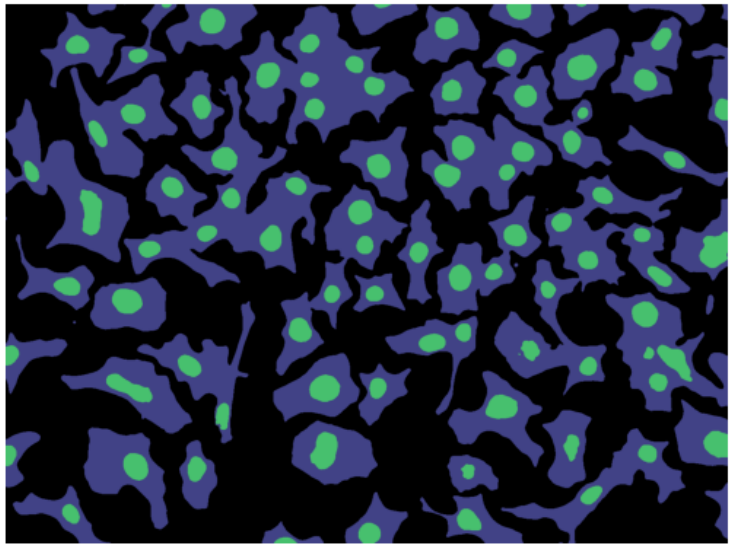


Binary segmentation +
Connected component labeling

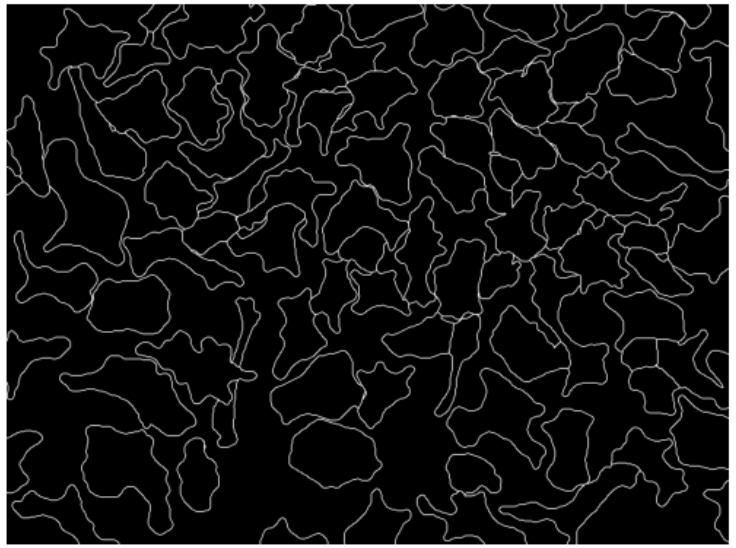
U-Net for Instance Segmentation



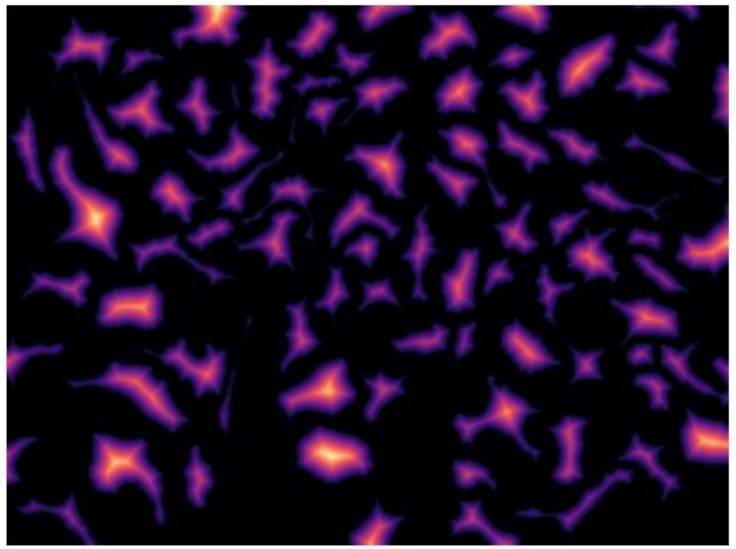
CNNs can predict more than binary outputs
→ Facilitate segmentation of object instances



Semantic segmentation
(Cells and nuclei)

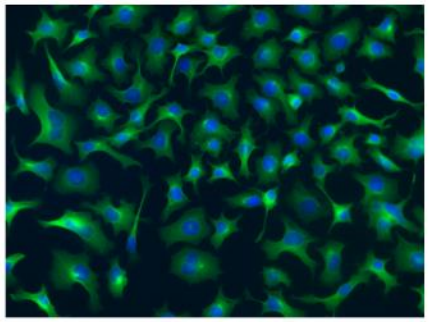


Cell boundaries

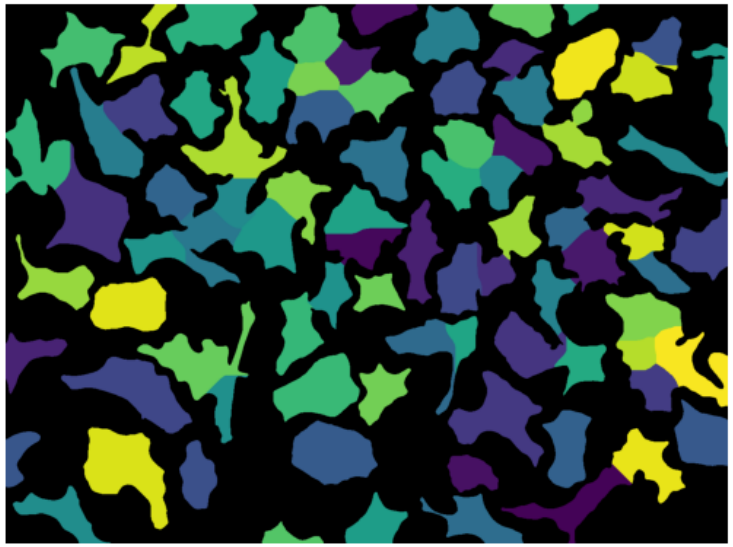


Distance to
cell boundaries

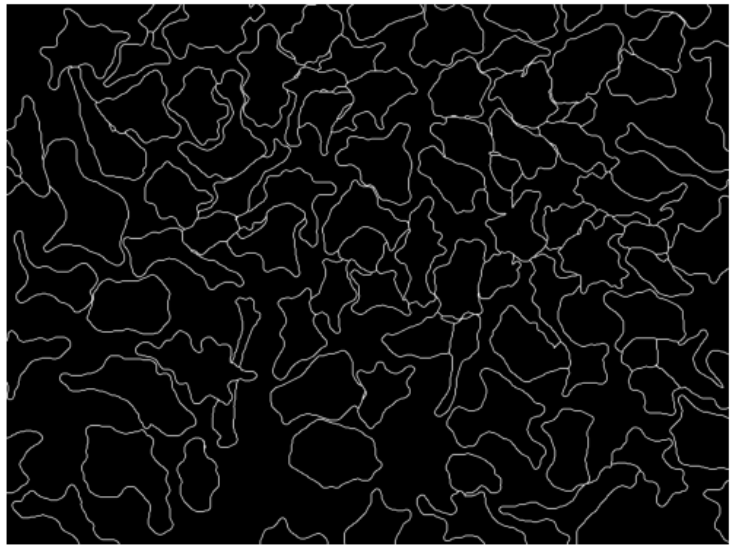
U-Net for Instance Segmentation



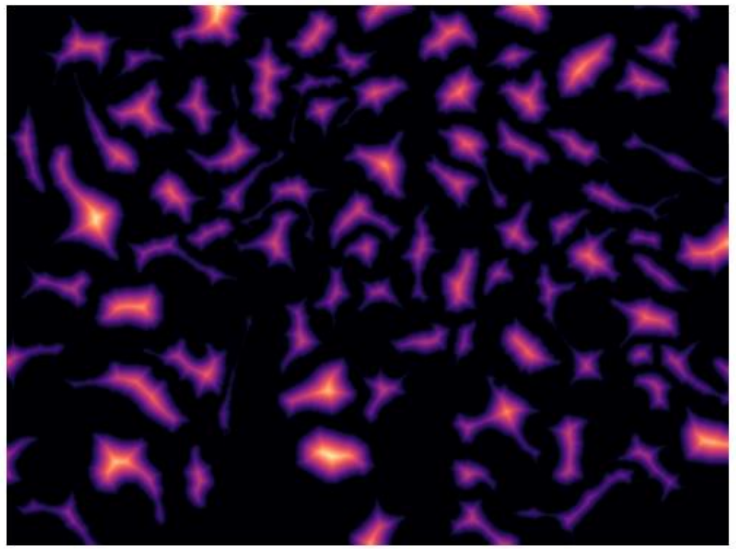
CNNs can predict more than binary outputs
→ Facilitate segmentation of object instances



Semantic segmentation
(Cells and nuclei)
↓
Marker-based Watershed

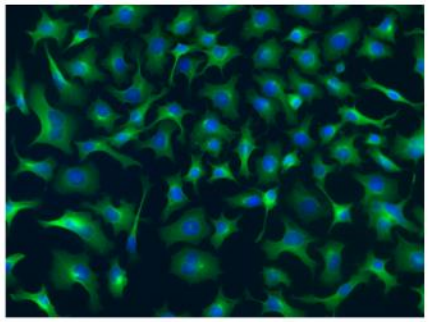


Cell boundaries

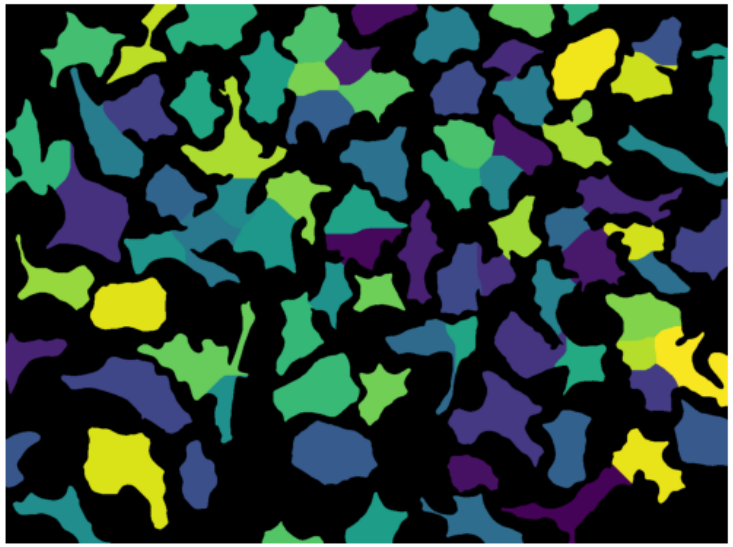


Distance to
cell boundaries

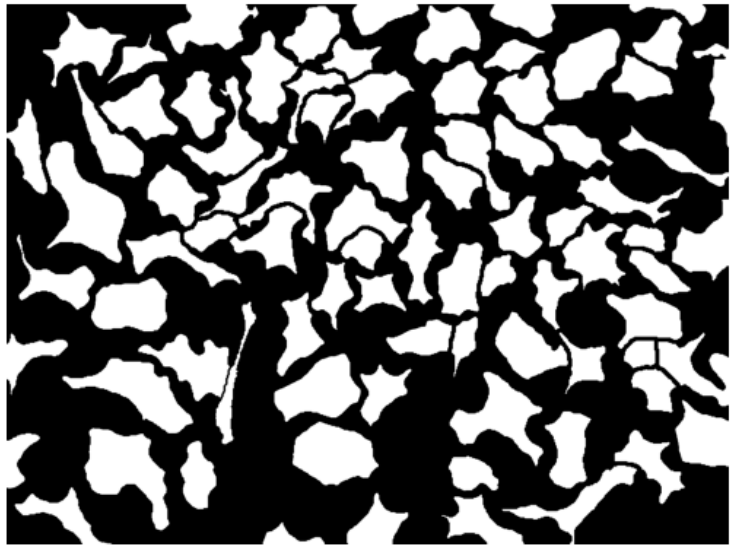
U-Net for Instance Segmentation



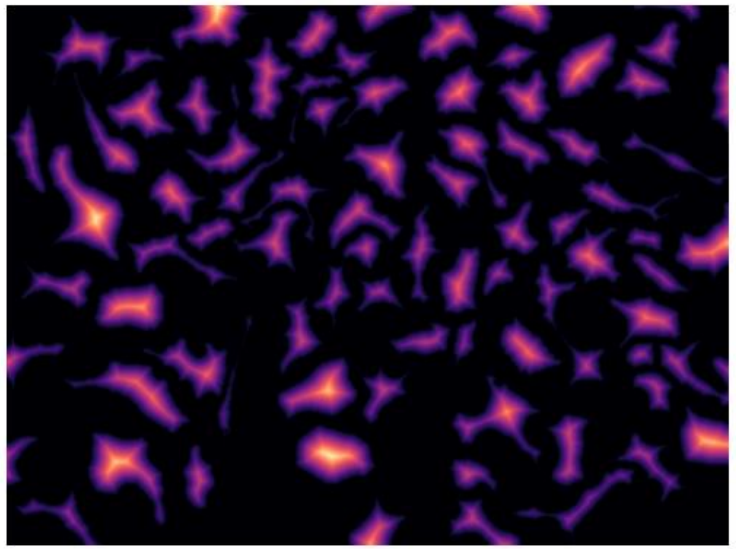
CNNs can predict more than binary outputs
→ Facilitate segmentation of object instances



Semantic segmentation
(Cells and nuclei)
↓
Marker-based Watershed

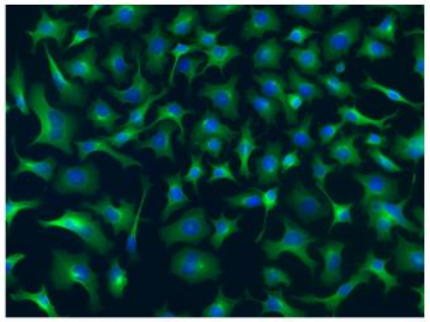


Cell boundaries
↓
Subtraction from binary segmentation

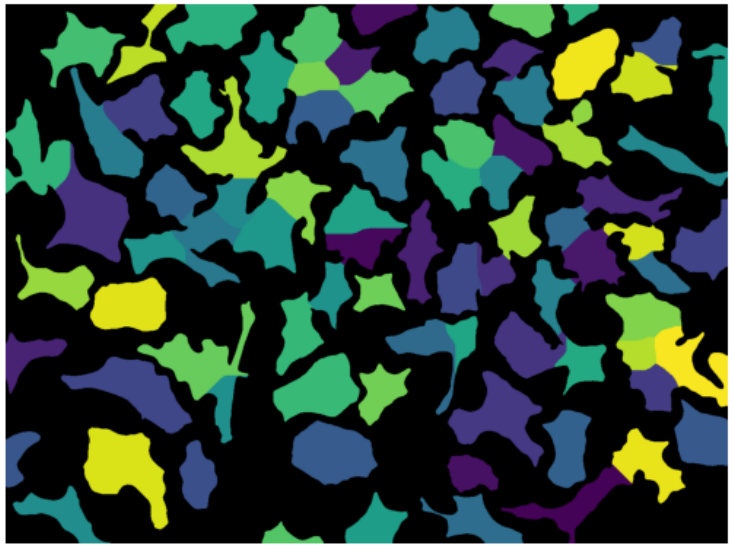


Distance to
cell boundaries

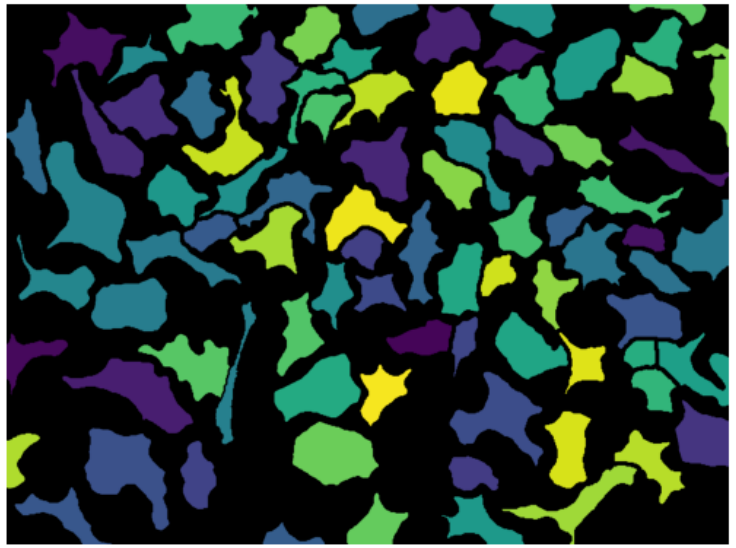
U-Net for Instance Segmentation



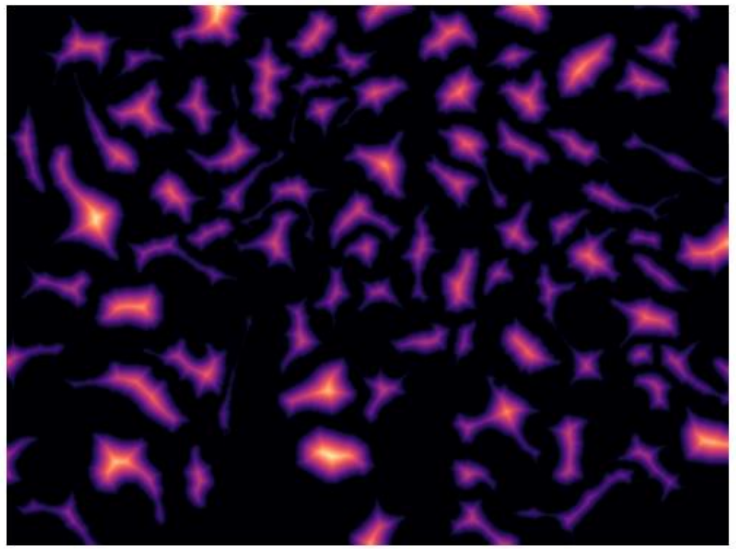
CNNs can predict more than binary outputs
→ Facilitate segmentation of object instances



Semantic segmentation
(Cells and nuclei)
↓
Marker-based Watershed

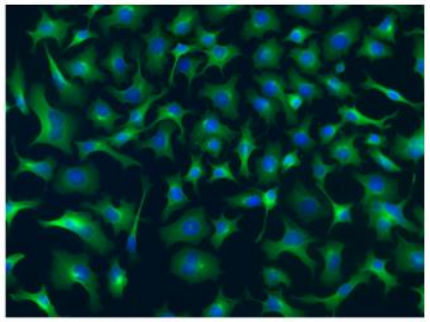


Cell boundaries
↓
Subtraction from binary segmentation
↓
Connected component labeling

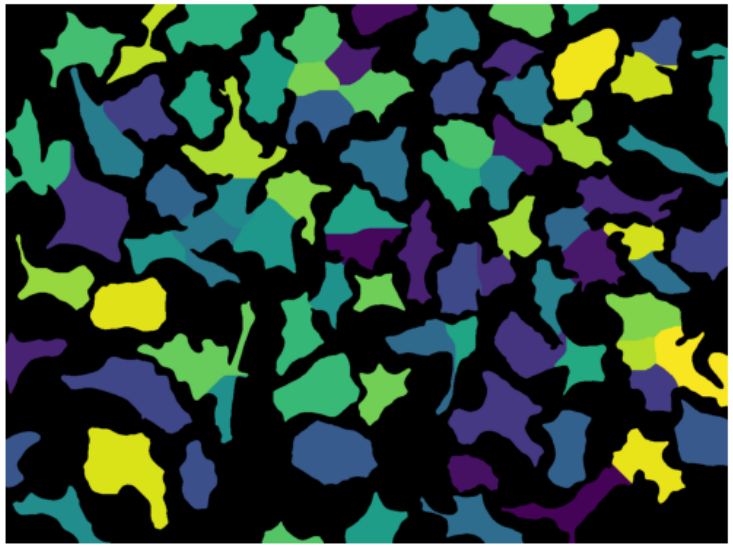


Distance to
cell boundaries

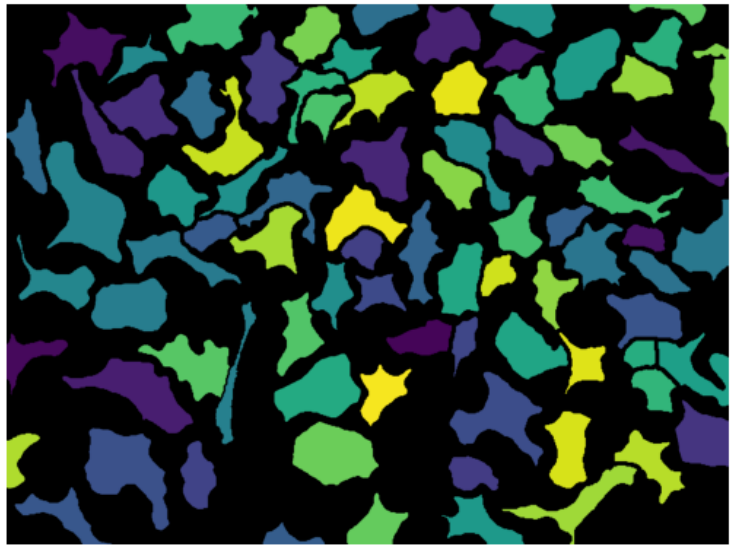
U-Net for Instance Segmentation



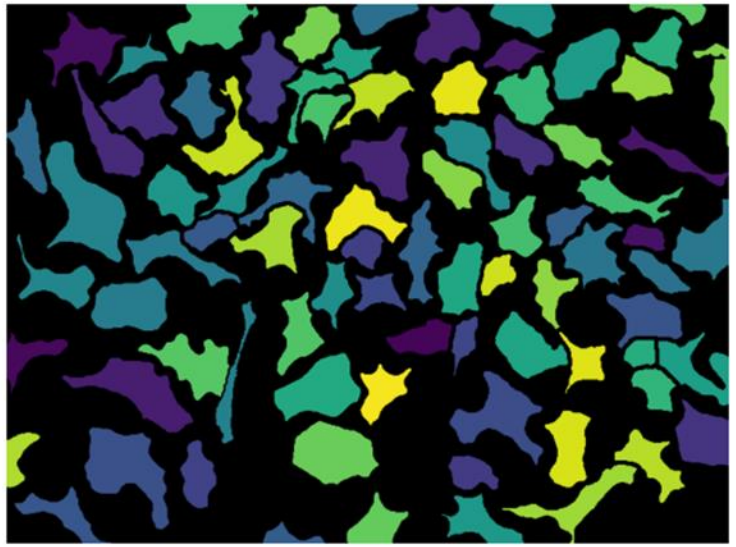
CNNs can predict more than binary outputs
→ Facilitate segmentation of object instances



Semantic segmentation
(Cells and nuclei)
↓
Marker-based Watershed



Cell boundaries
↓
Subtraction from binary segmentation
↓
Connected component labeling

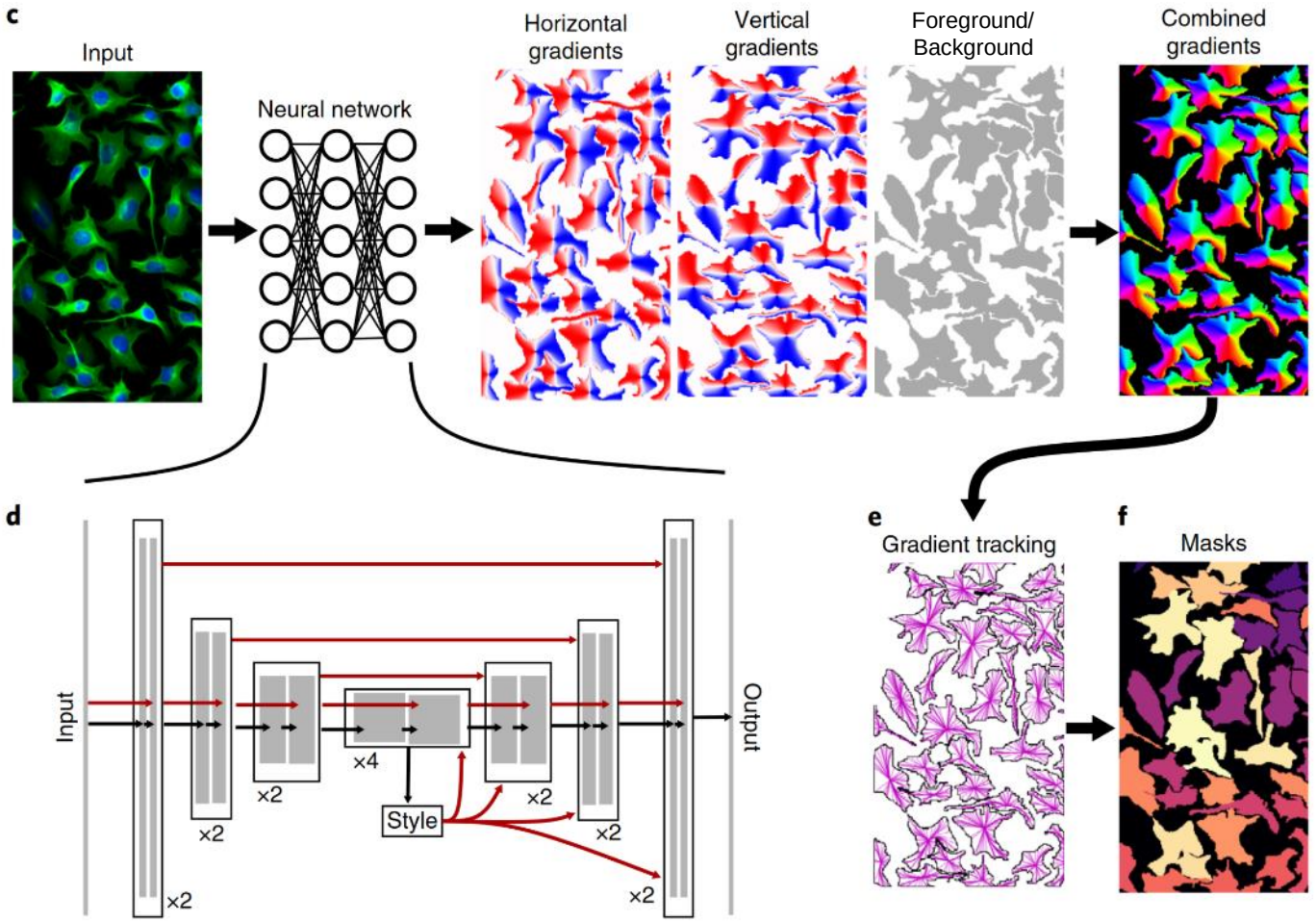


Distance to
cell boundaries
↓
Non-maximum suppression
↓
Marker-based Watershed

Cellpose

Segmentation based on gradient flow

- **Heat gradients** from center of cell towards boundaries
- All gradients within a cell point towards center
- Following gradient separates touching cells

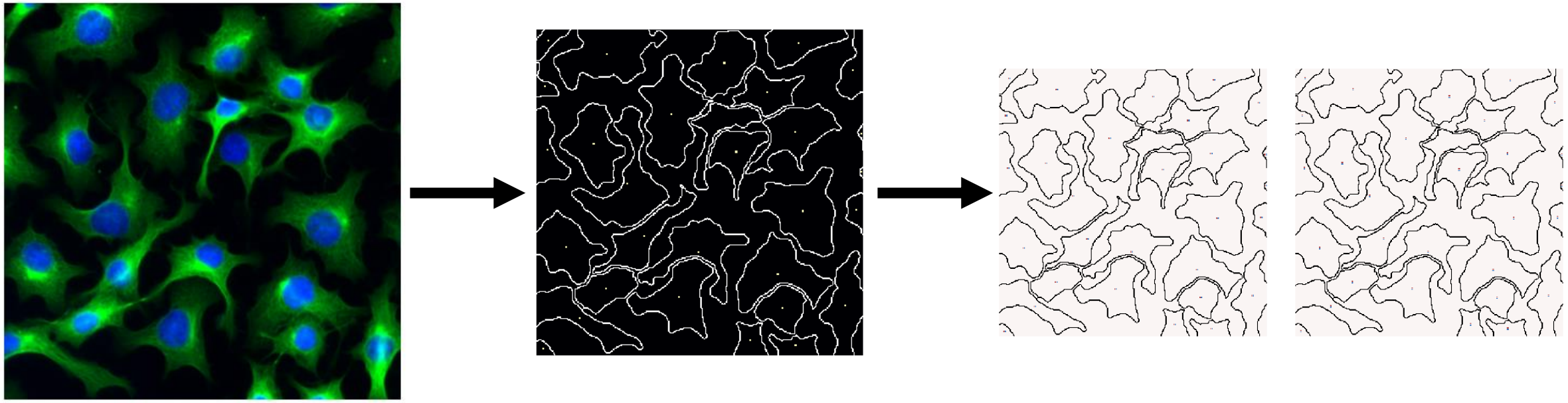


Stringer et al., Nat. Methods 2021

Training Data: Simulated Heat Gradients

Segmentation based on gradient flow

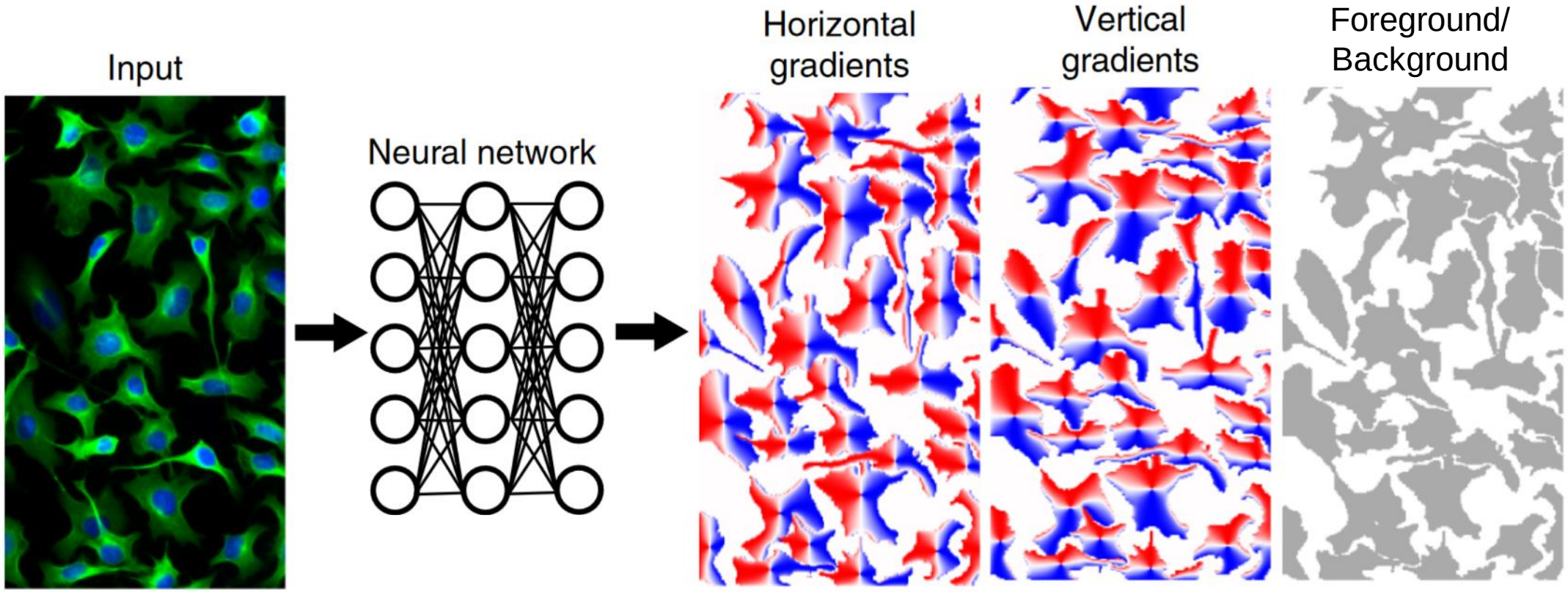
- Simulated **heat diffusion** from cell center (heat source) to boundaries
- Flow: Horizontal and vertical heat gradients



Intensity image **Heat diffusion** **Horizontal and vertical gradients**

Network Outputs

Prediction of foreground probability and heat diffusion gradients

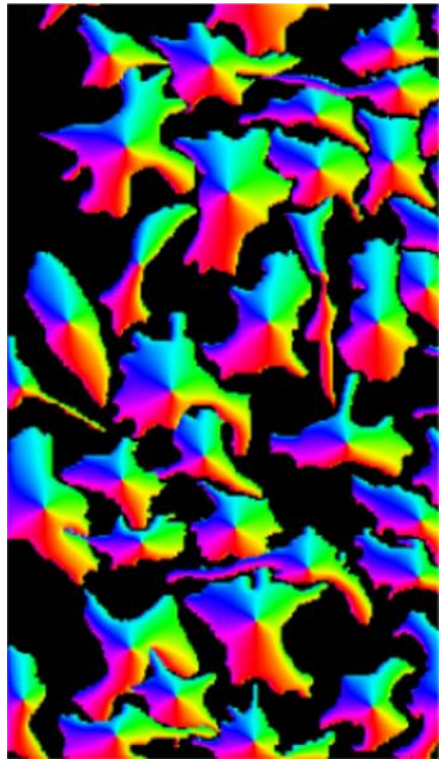


Stringer et al., Nat. Methods 2021

Instance Segmentation

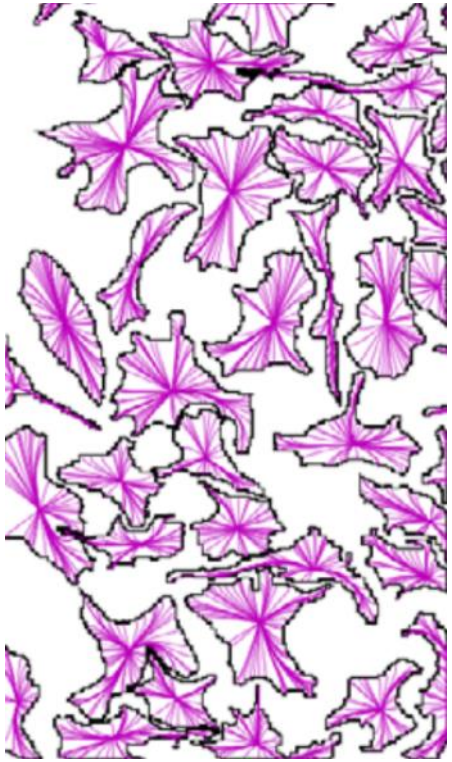
Gradient tracking to identify cell instances

Combined gradients



Gradients are iteratively followed towards heat source

Flow towards center



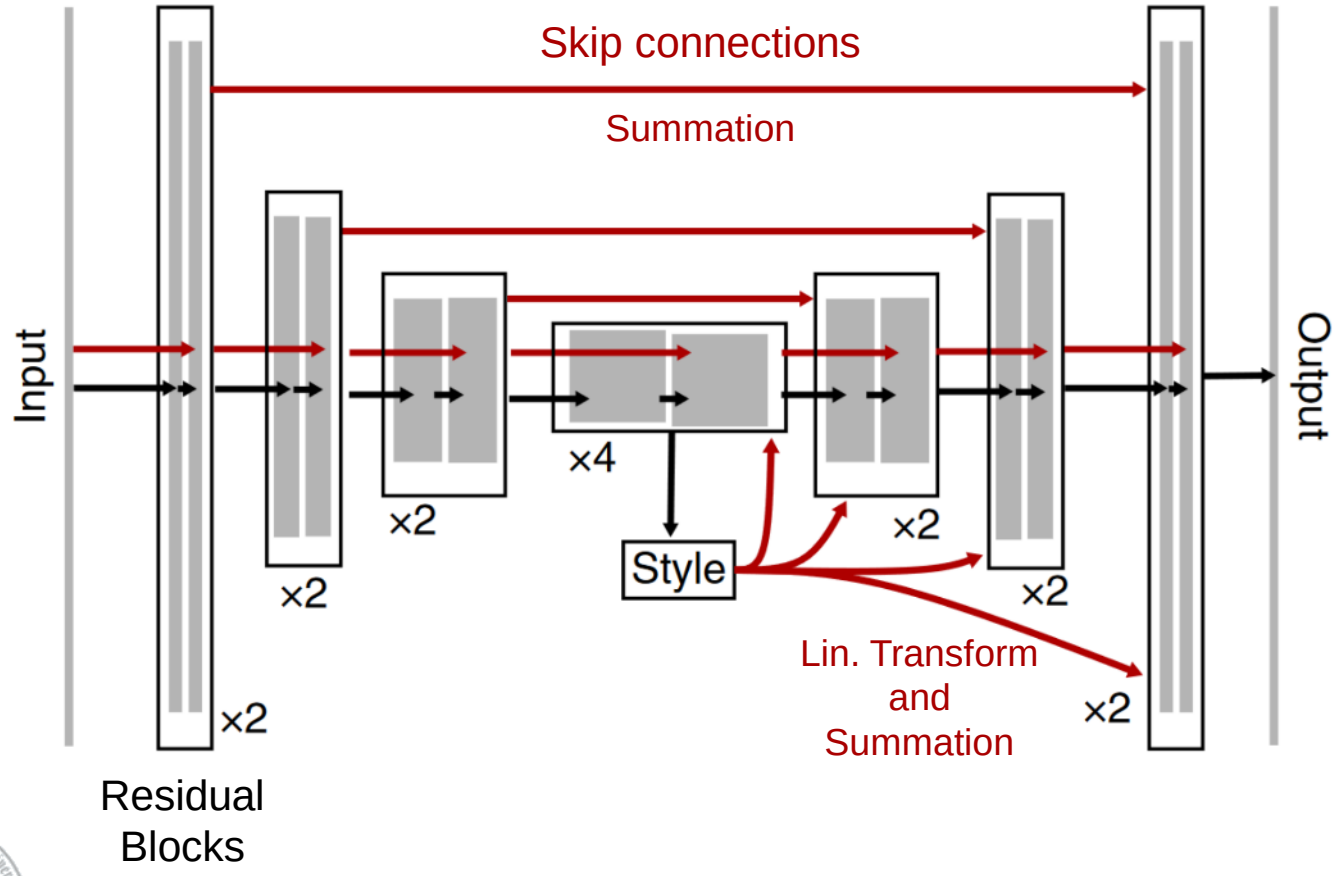
Flow target determines cell instance

Instance segmentation

Stringer et al., Nat. Methods 2021

Cellpose Network Architecture

Cellpose model is based on U-Net

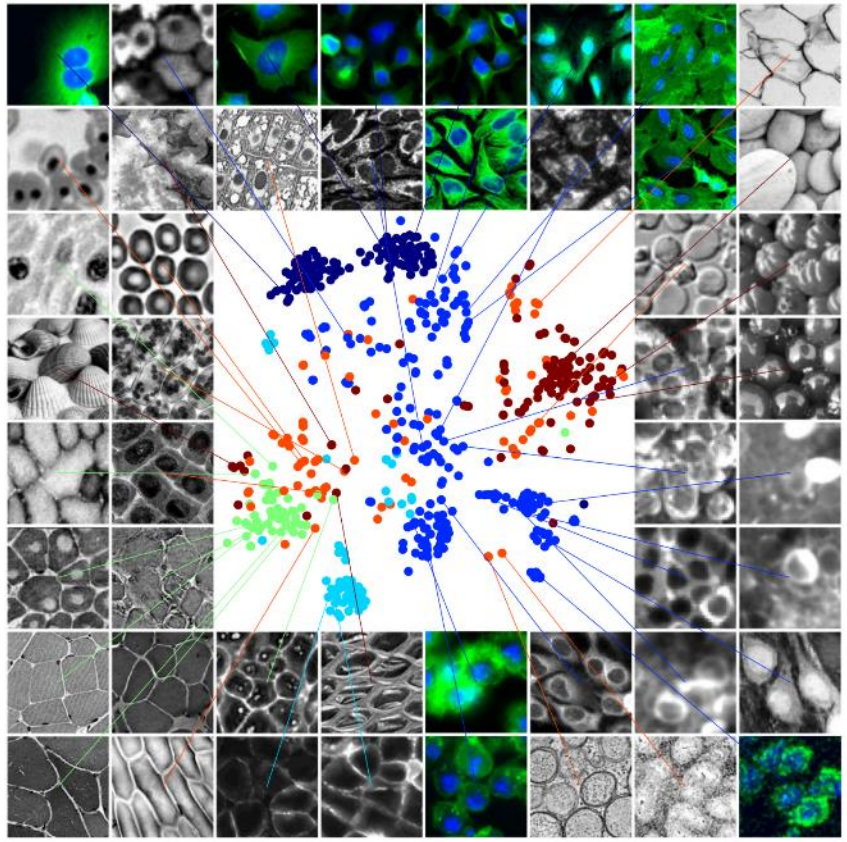


Adaptions compared to U-Net

- Residual blocks
  → better performance
- Skip connections use summation instead of concatenation
  → reduces number of parameters
- Style vectors
  → information on image modalities

Stringer et al., Nat. Methods 2021

Style vectors



t-SNE of style vectors from different image modalities

Motivation:

- Adapt model behavior for diverse images
- Increase performance for unseen data

Implementation:

- 256d style vectors are computed through max-pooling at bootleneck
- Styles are transformed linearly and added to feature maps during upsampling

Style vectors of similar image modalities are close in feature space

Stringer et al., Nat. Methods 2021

Training and Data Augmentation

Trained on image patches

- **Augmentation:** Rescale, Crop, Rotation

- **Rescaling:** Images are rescaled so that median cell size is the same for all images in batch
- **Crop:** Random patches, rescaled to 224 x 224 pixels
  → Aspect ratio and cell size are changed

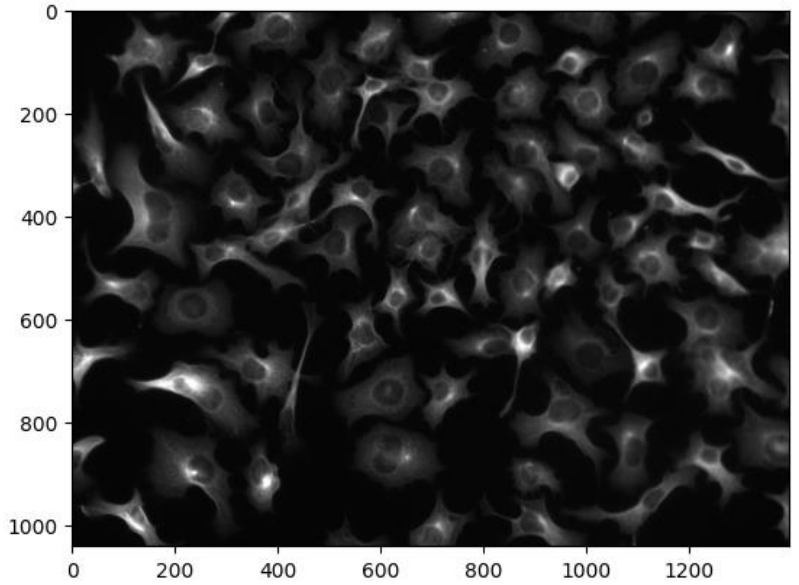
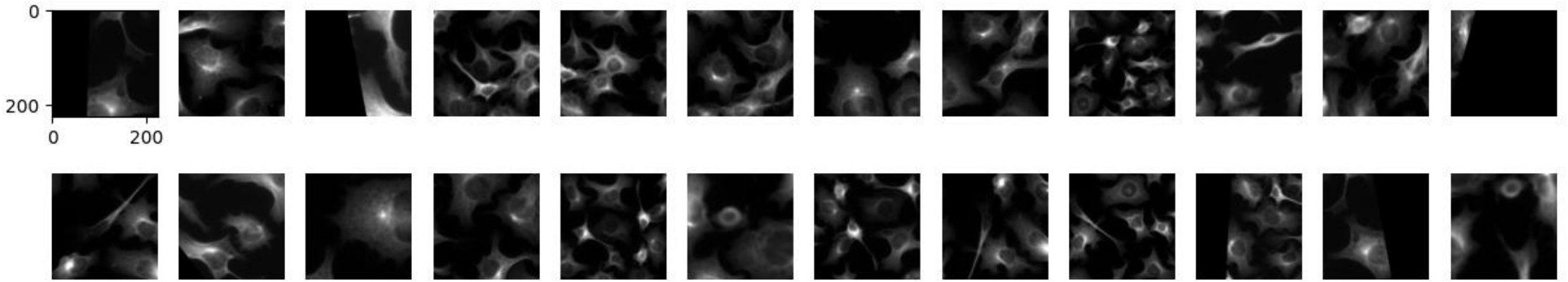


Image before augmentation



Augmented image patches

Loss Function

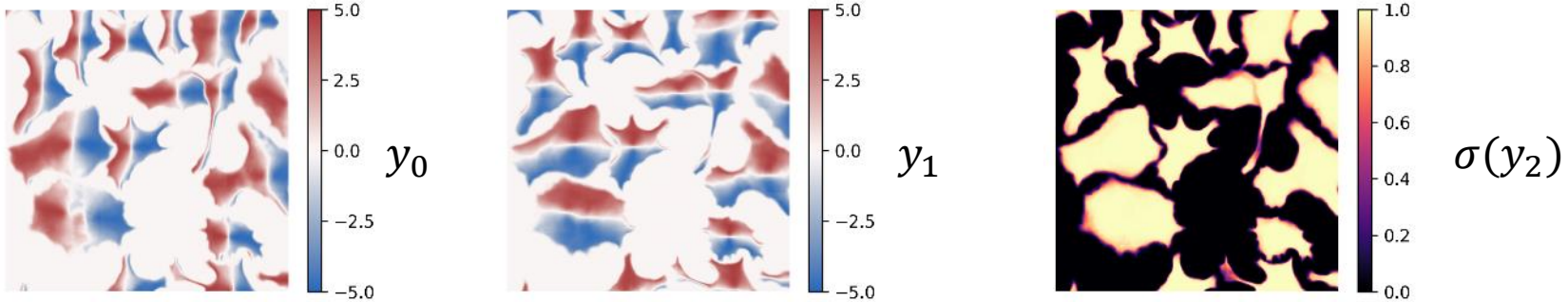
L2 loss **BCE loss**

$$Loss = \left|\left|y_0 - 5H\right|\right|^2 + \left|\left|y_1 - 5V\right|\right|^2 + L(\sigma(y_2), P)$$

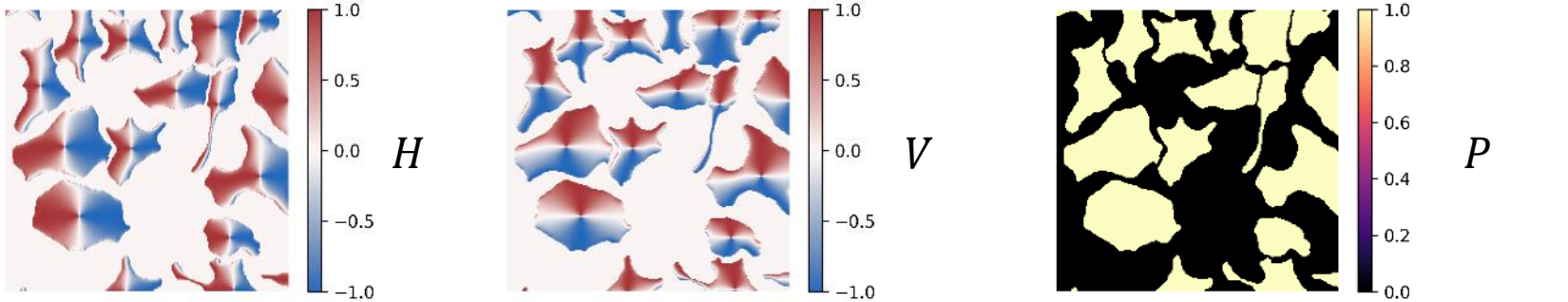
Gradients **Foreground/ Background**

Horizontal Vertical

Predictions



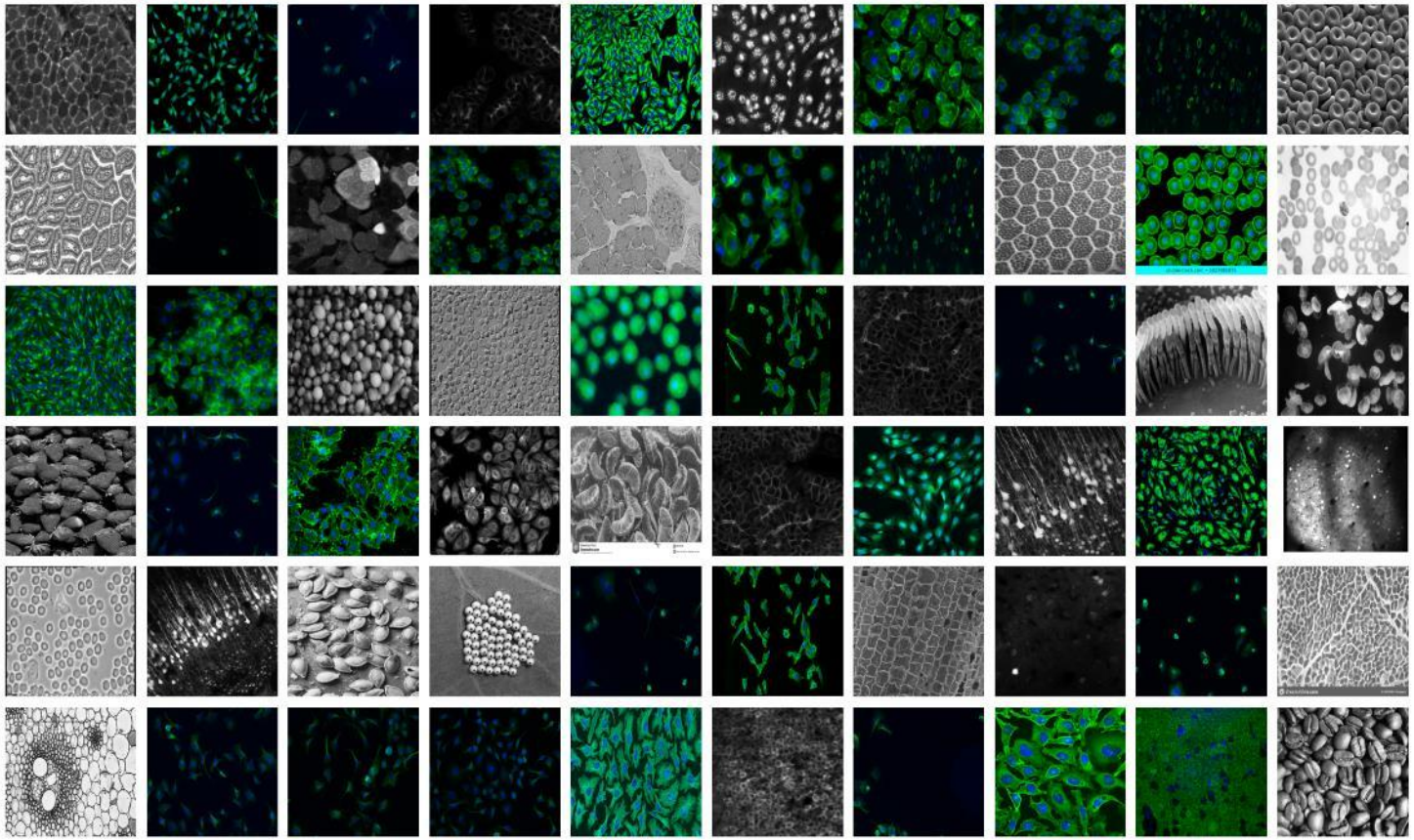
Ground Truth



Stringer et al., Nat. Methods 2021

Generalized Dataset: *cyto*

Cellpose dataset includes various image modalities



Images from *cyto* dataset

Specialized (published)
Cell Image Library ($n = 100$)

Generalized (aggregated from various sources)

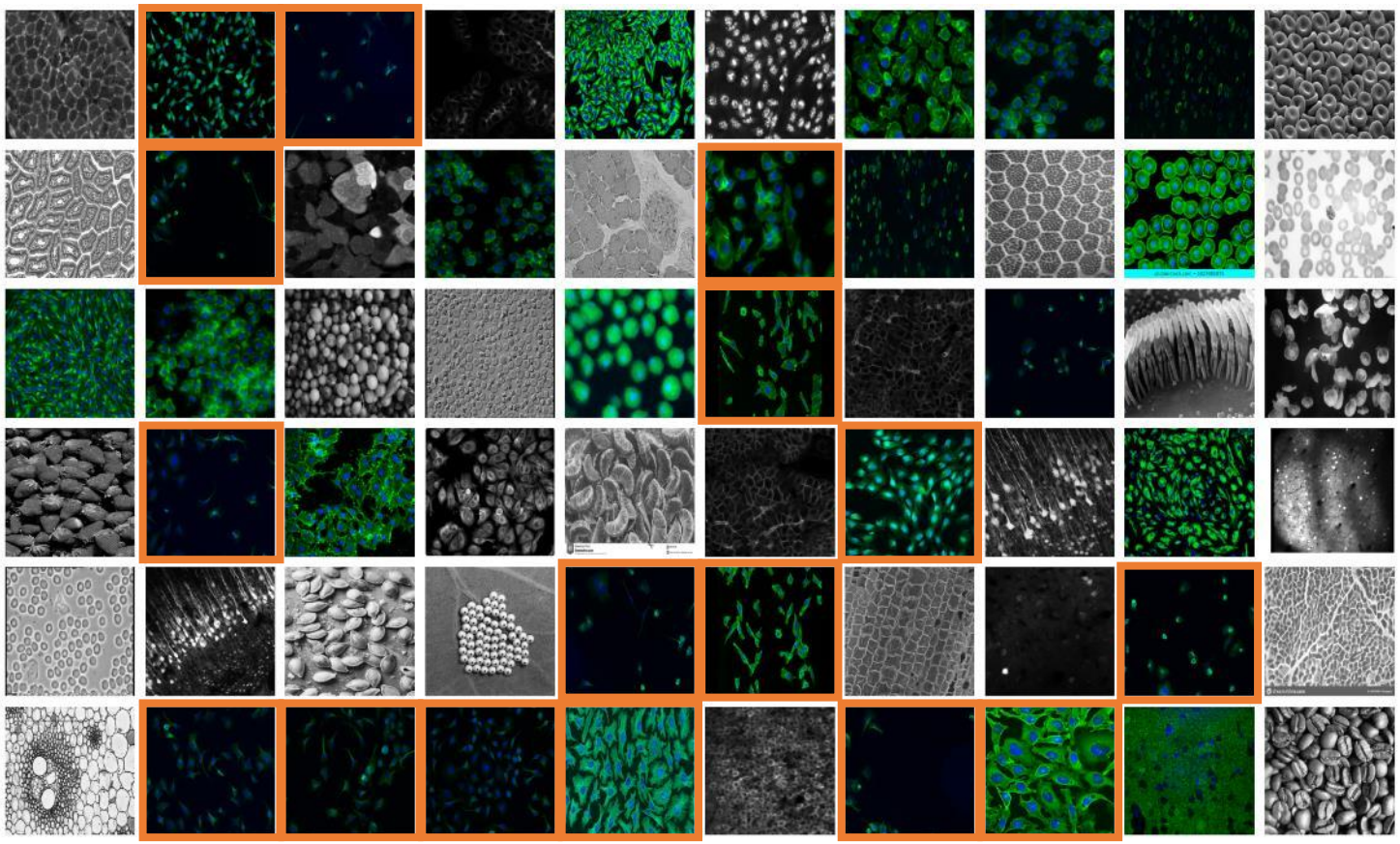
- Cell Image Library ($n = 100$)
- Cells, fluorescent ($n = 216$)
- Cells, non-fluorescent ($n = 50$)
- Cells, membranes ($n = 58$)
- Microscopy, other ($n = 86$)
- Nonmicroscopy ($n = 98$)

Additional datasets have been included in newer versions (e.g., user-generated, bacteria, worms)

Stringer et al., Nat. Methods 2021
Stringer et al., Nat. Methods 2022
Stringer and Pachitariu, bioRxiv 2024

Generalized Dataset: *cyto*

Cellpose dataset includes various image modalities



Images from *cyto* dataset

Specialized (published)
Cell Image Library ($n = 100$)

Generalized (aggregated from various sources)

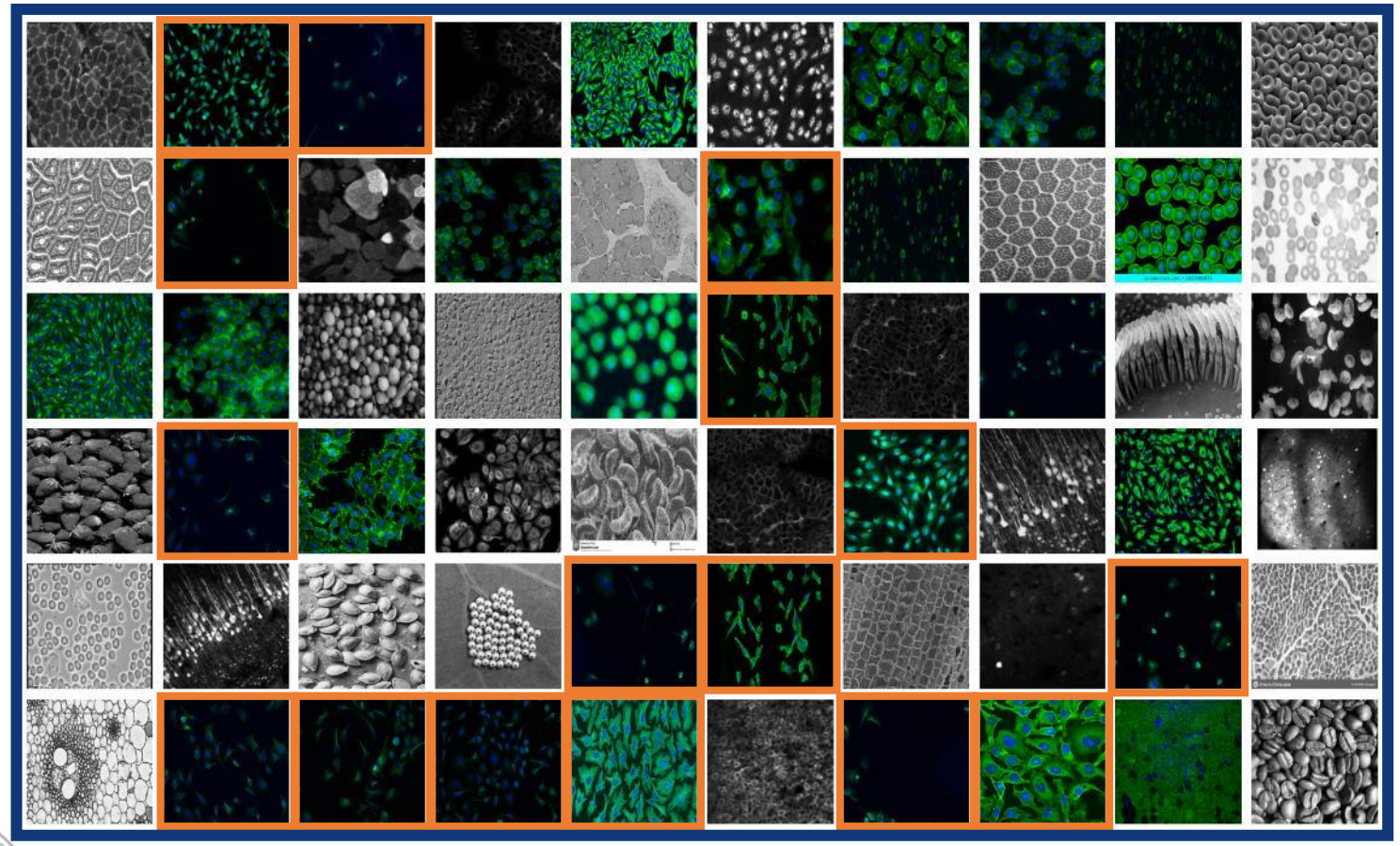
- Cell Image Library ($n = 100$)
- Cells, fluorescent ($n = 216$)
- Cells, non-fluorescent ($n = 50$)
- Cells, membranes ($n = 58$)
- Microscopy, other ($n = 86$)
- Nonmicroscopy ($n = 98$)

Additional datasets have been included in newer versions (e.g., user-generated, bacteria, worms)

Stringer et al., Nat. Methods 2021
Stringer et al., Nat. Methods 2022
Stringer and Pachitariu, bioRxiv 2024

Generalized Dataset: *cyto*

Cellpose dataset includes various image modalities



Images from *cyto* dataset

Specialized (published)
Cell Image Library ($n = 100$)

Generalized (aggregated from various sources)

- Cell Image Library ($n = 100$)
- Cells, fluorescent ($n = 216$)
- Cells, non-fluorescent ($n = 50$)
- Cells, membranes ($n = 58$)
- Microscopy, other ($n = 86$)
- Nonmicroscopy ($n = 98$)

Additional datasets have been included in newer versions (e.g., user-generated, bacteria, worms)

Stringer et al., Nat. Methods 2021
Stringer et al., Nat. Methods 2022
Stringer and Pachitariu, bioRxiv 2024

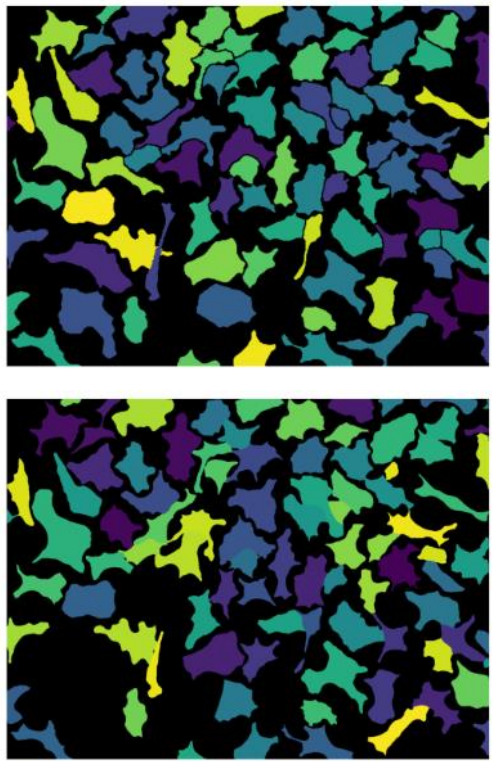
Average Precision

Segmentation performance evaluated at multiple matching thresholds

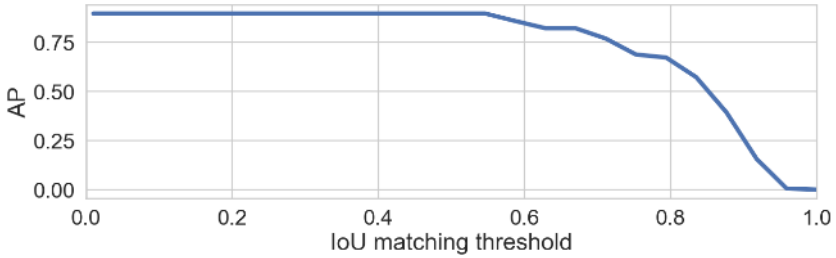
- Matching criterion: **Intersection over Union**

GT

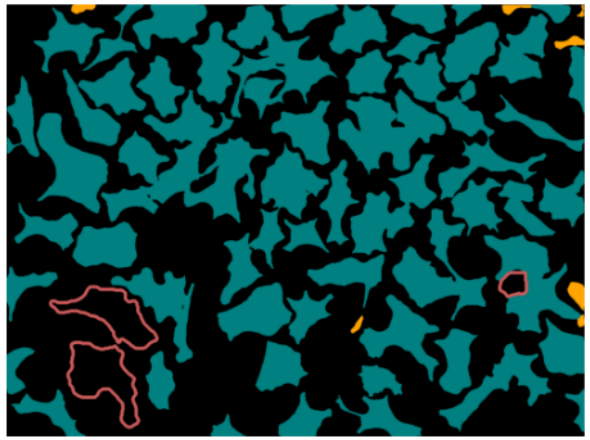
Pred



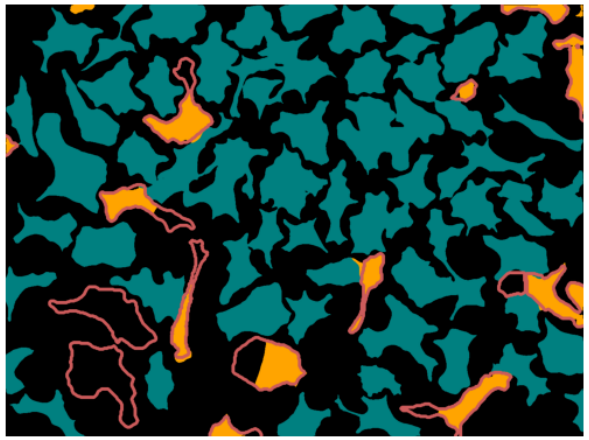
$$AP_{IoU} = \frac{TP}{TP + FP + FN}$$



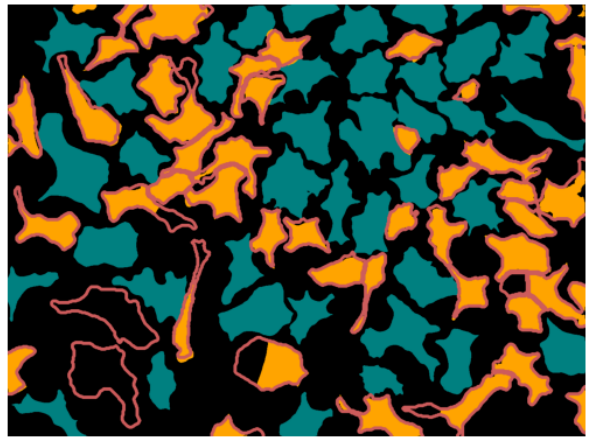
$\text{IoU} \geq 0.10$



$\text{IoU} \geq 0.75$



$\text{IoU} \geq 0.90$



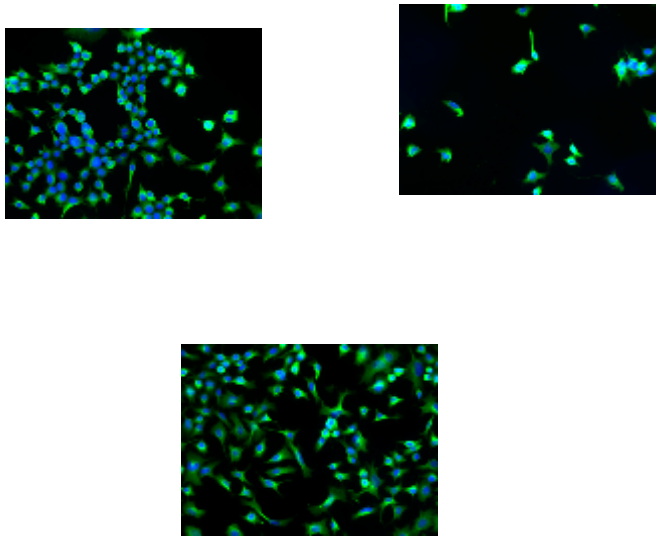
True Positive, False Positive, False Negative segmentations

Segmentation Performance

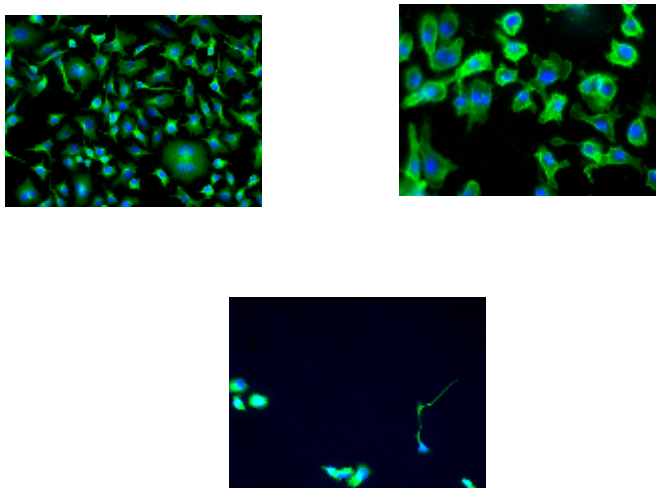
Experimental Results

- Broad range of data benefits performance for **all image modalities**
- Cellpose outperforms competitor methods

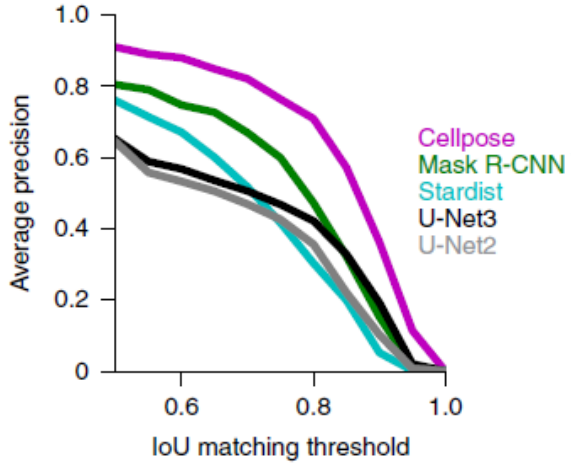
Training Data
Specialized



Testing Data
Specialized



Specialist model/ Specialized data



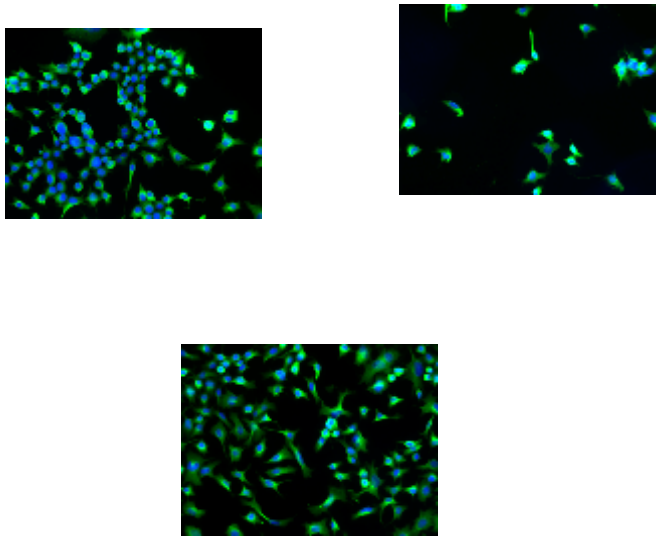
Stringer et al., Nat. Methods 2021

Segmentation Performance

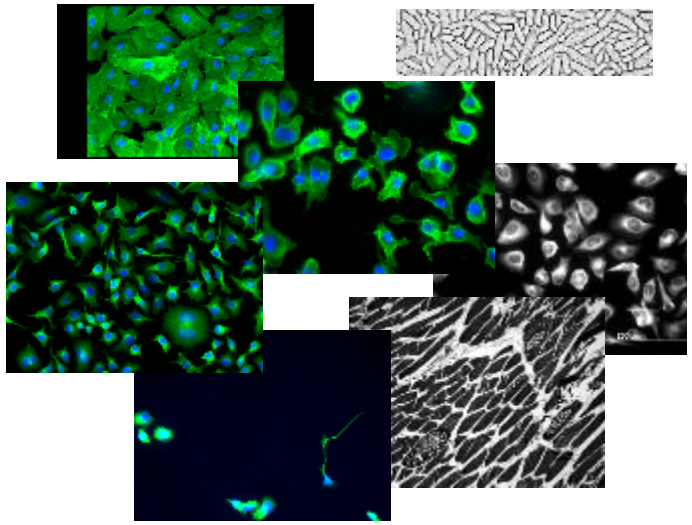
Experimental Results

- Broad range of data benefits performance for **all image modalities**
- Cellpose outperforms competitor methods

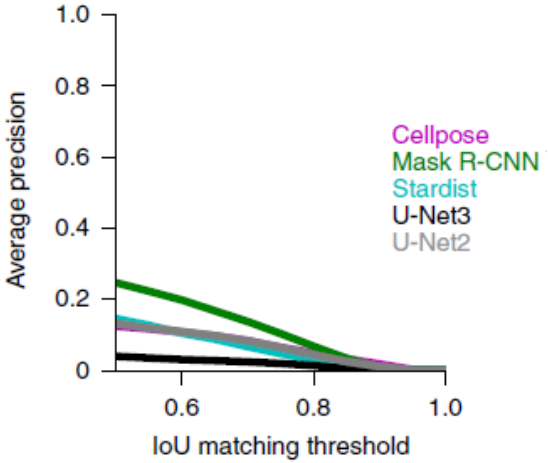
Training Data
Specialized



Testing Data
Generalized



Specialist model/ Generalized data



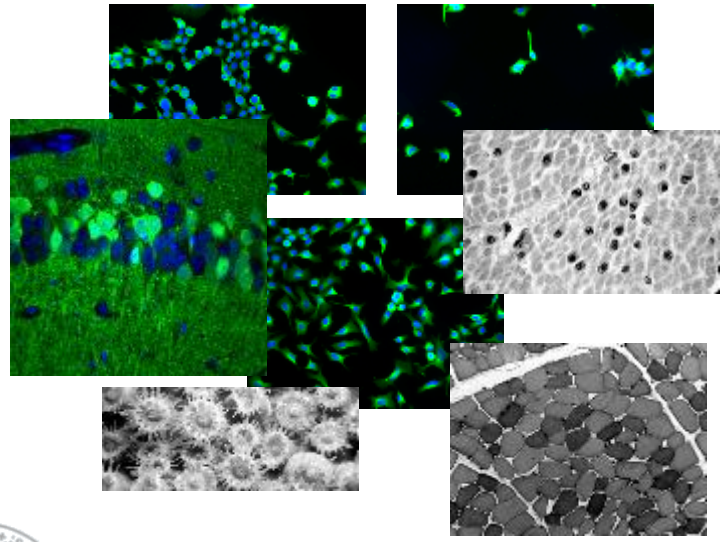
Stringer et al., Nat. Methods 2021

Segmentation Performance

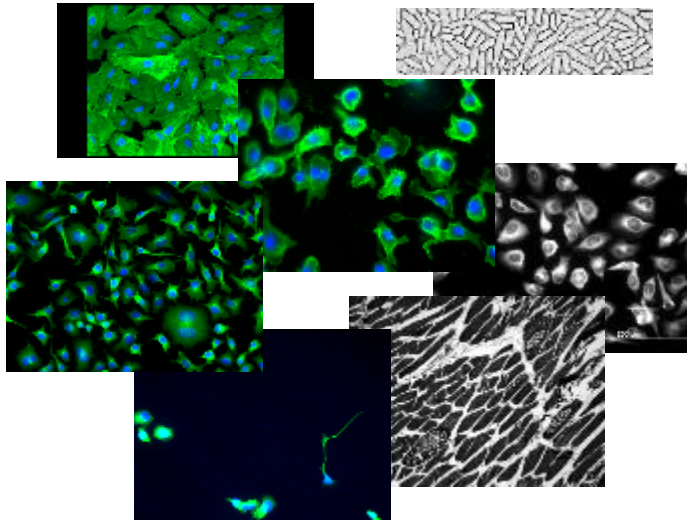
Experimental Results

- Broad range of data benefits performance for **all image modalities**
- Cellpose outperforms competitor methods

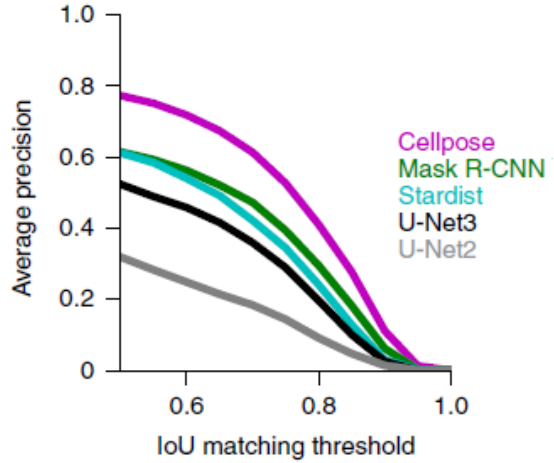
Training Data
Generalized



Testing Data
Generalized



Generalist model/ Generalized data



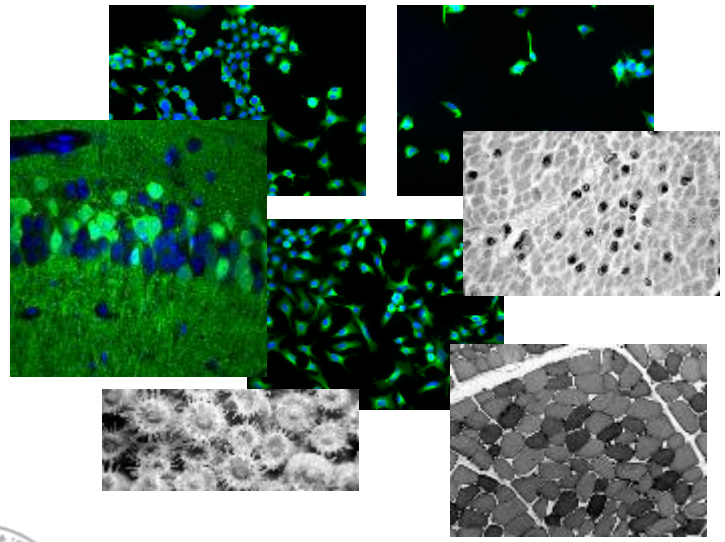
Stringer et al., Nat. Methods 2021

Segmentation Performance

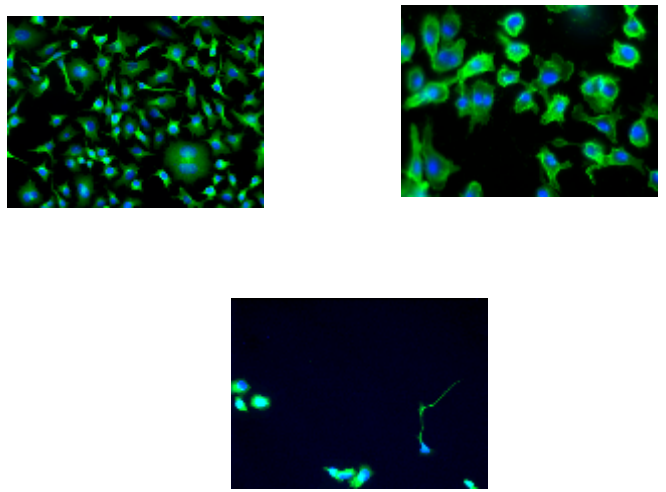
Experimental Results

- Broad range of data benefits performance for **all image modalities**
- Cellpose outperforms competitor methods

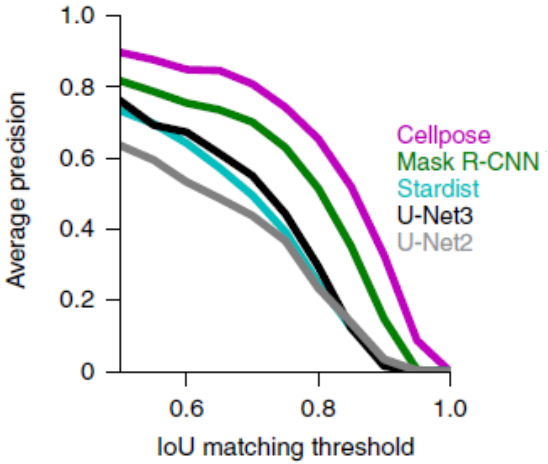
Training Data
Generalized



Testing Data
Specialized



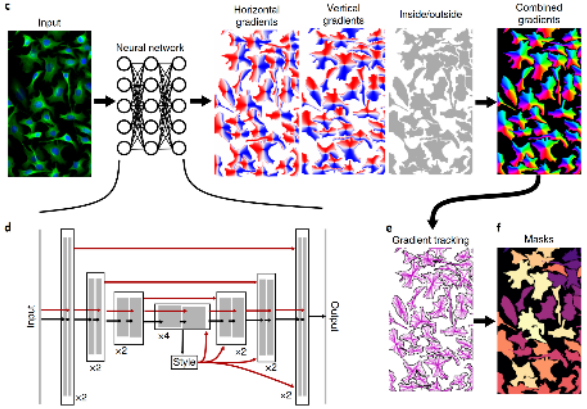
Generalist model/ Specialized data



Stringer et al., Nat. Methods 2021

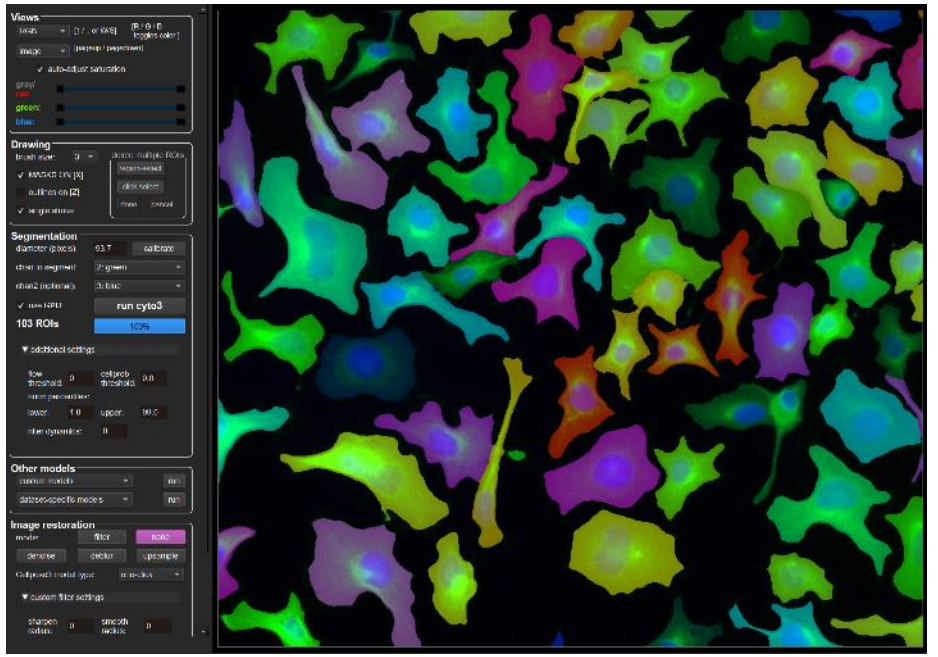
Summary

- Cell instance segmenation based on **predicted flows**
- **Generalist** model allows out-of-the-box segmentation of many image types



Method overview

- **Easy to use:**
  - GUI
  - Integration with other software e.g., ImageJ, Napari
- Continuous development with new models and data
  - 3 major publications since 2021, various adaptions/extensions



Cellpose GUI

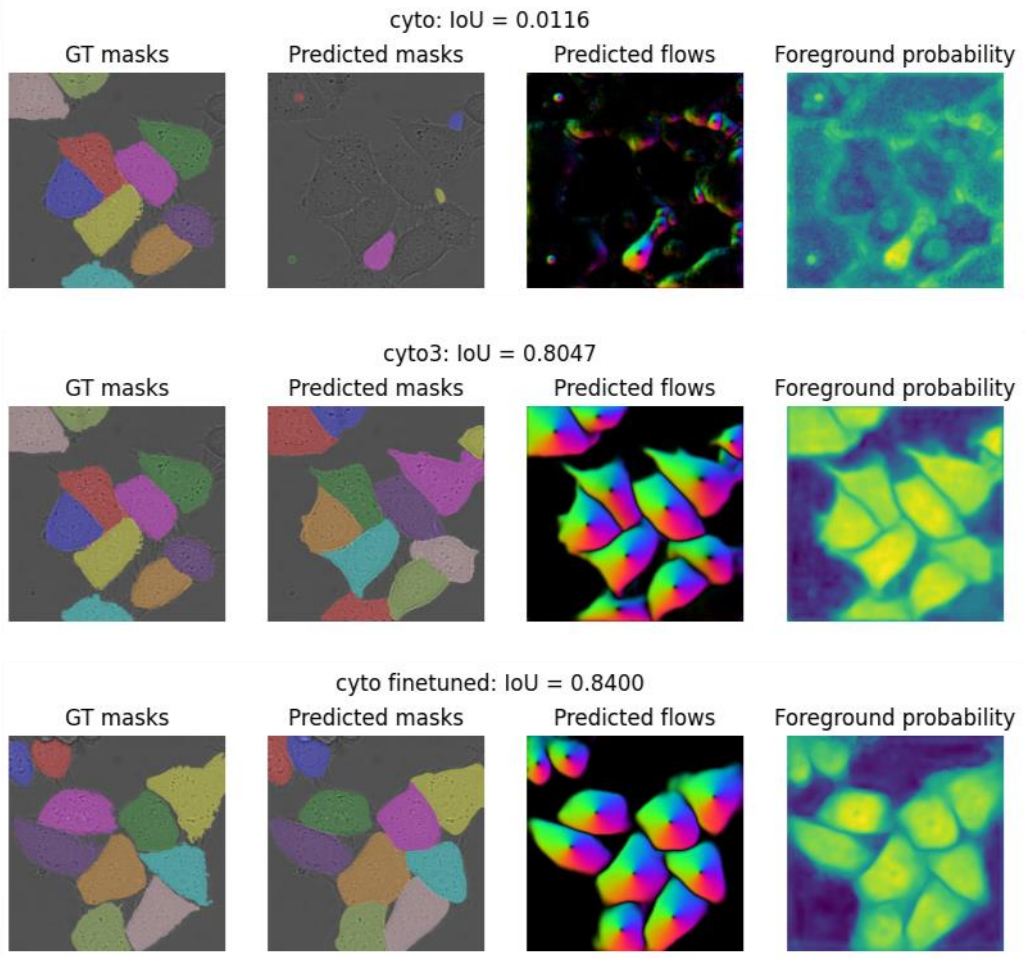
Stringer et al., Nat. Methods 2021

Practical Exercises

Default parameters

Optimized parameters

Fine-tuned parameters



- Visualization of Cellpose outputs
- Using built-in Cellpose models for segmentation of new data
- Adapting segmentation parameters
- Implementing/using Average Precision for evaluation of instance segmentation
- Fine-tuning built-in Cellpose models for new data

Questions ?