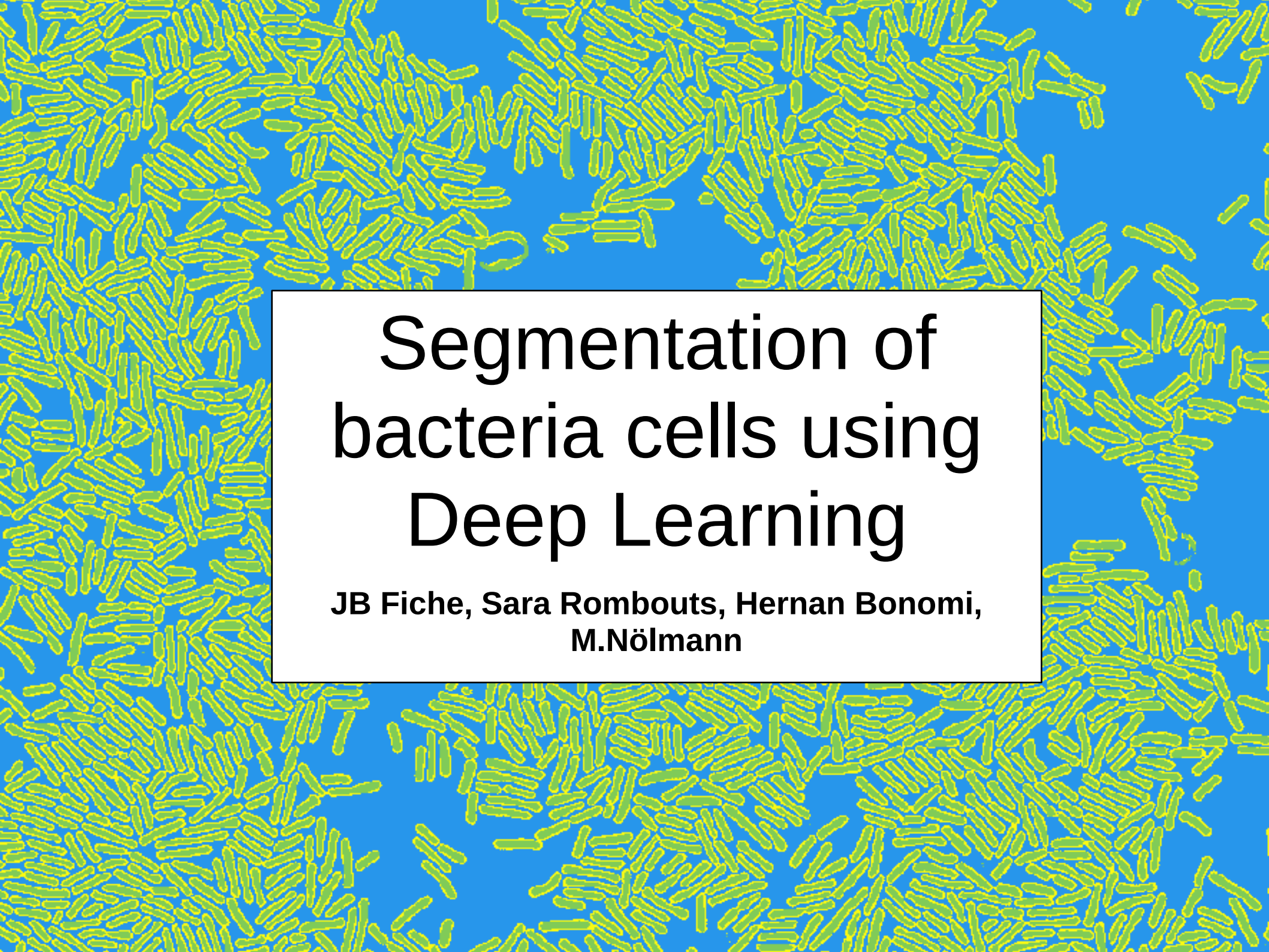


Segmentation of bacteria cells using Deep Learning

JB Fiche, Sara Rombouts, Hernan Bonomi, M.Nölmann

Outline

I. **Why do we want to used deep learning?**

II. Brief history of deep learning

III. Short introduction on neural network (NN)

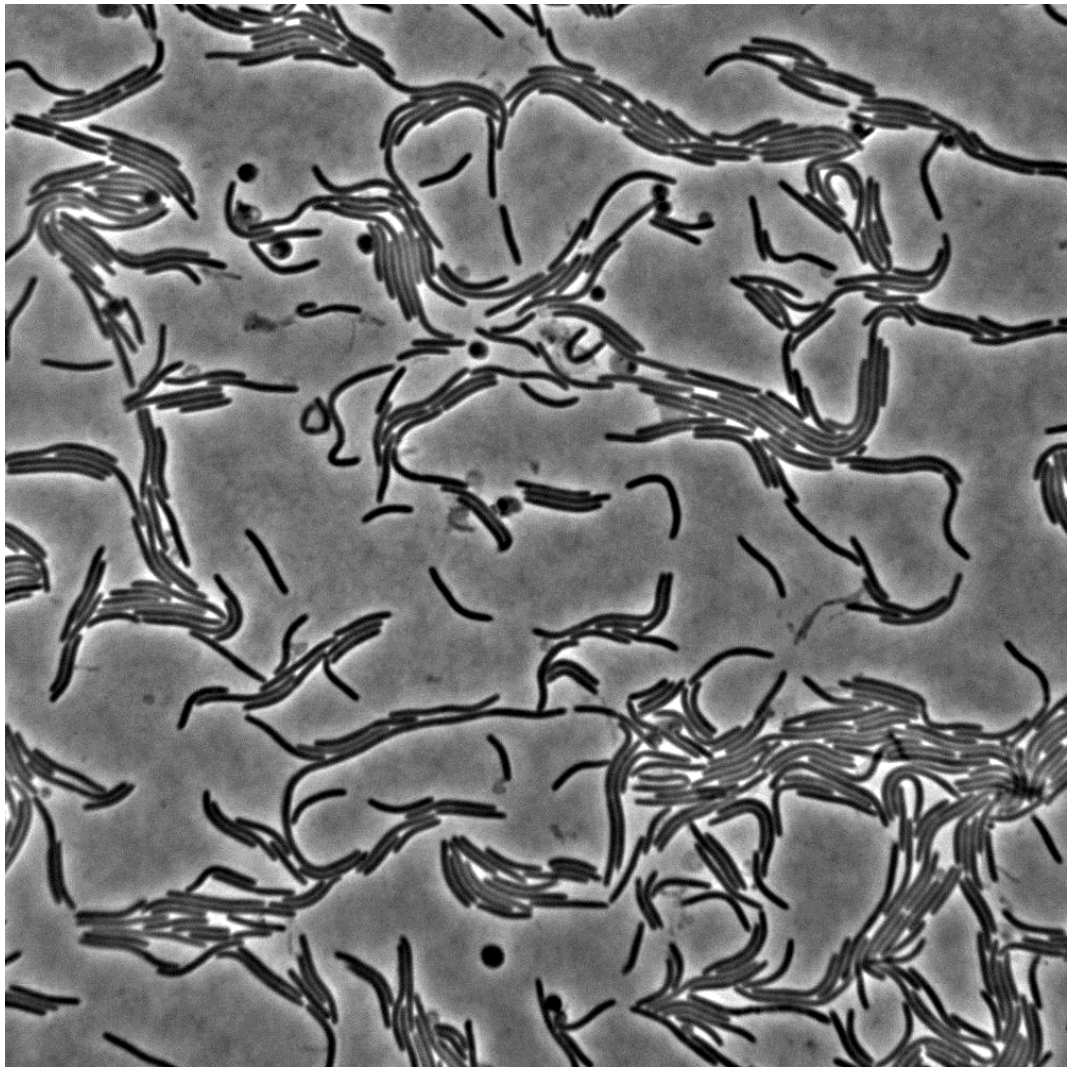
IV. Convolutional neural network (CNN) for image classification

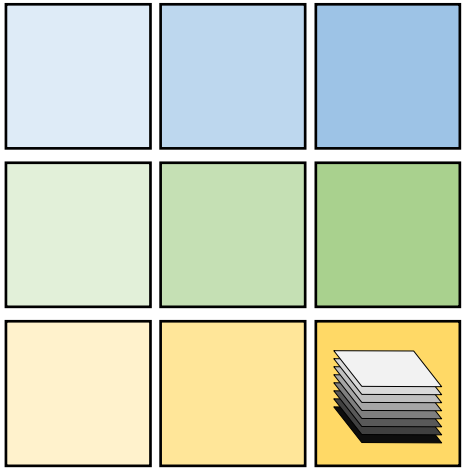
V. Van-Valen network for cell segmentation

VI. Results

VII. Conclusion & perspectives

Tracking *Myxococcus xanthus* cells on an agar pad



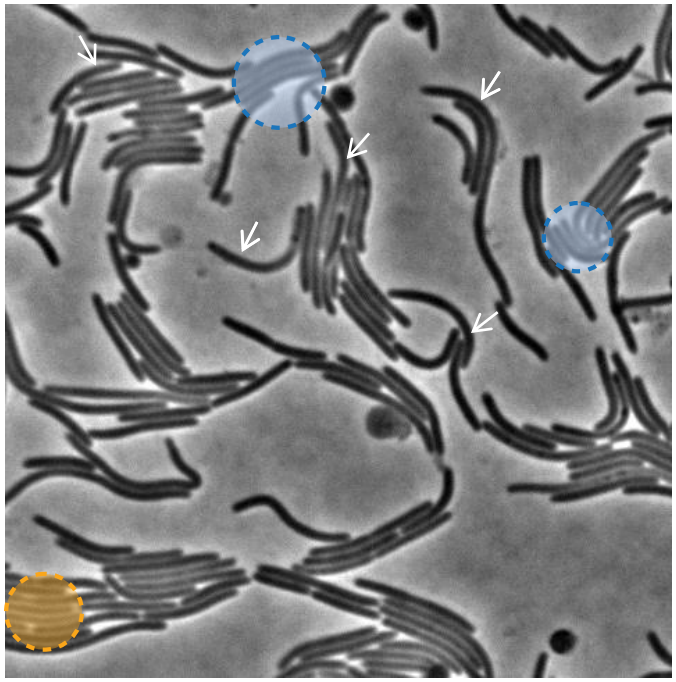


3x3 ROI for a total of ~600x600µm
Stack of 7 images
~40-50s to complete a cycle

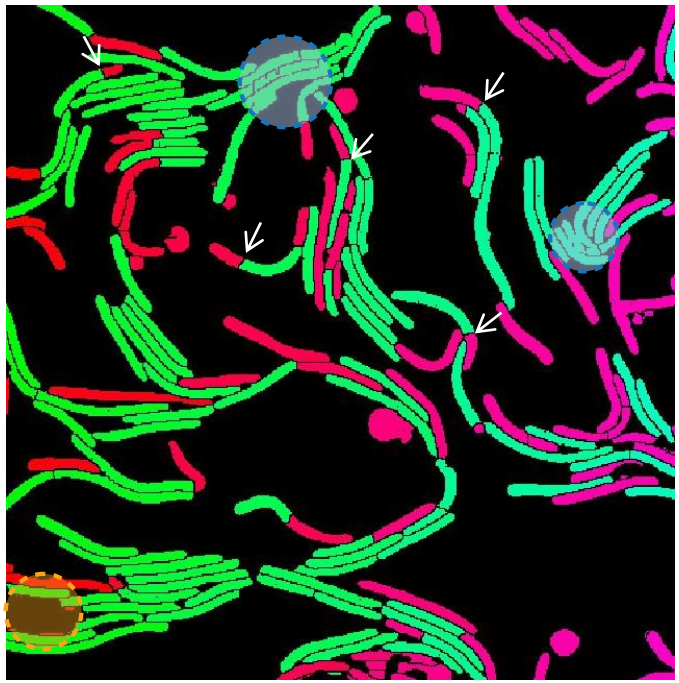
Myxococcus xanthus gliding assay on a 1,5% agar pad. We need a reliable method to segment the cells and track them over time :

- *predation behavior*
- *cell division*
- *etc.*

Segmentation of *Myxococcus xanthus* cells using super-segger



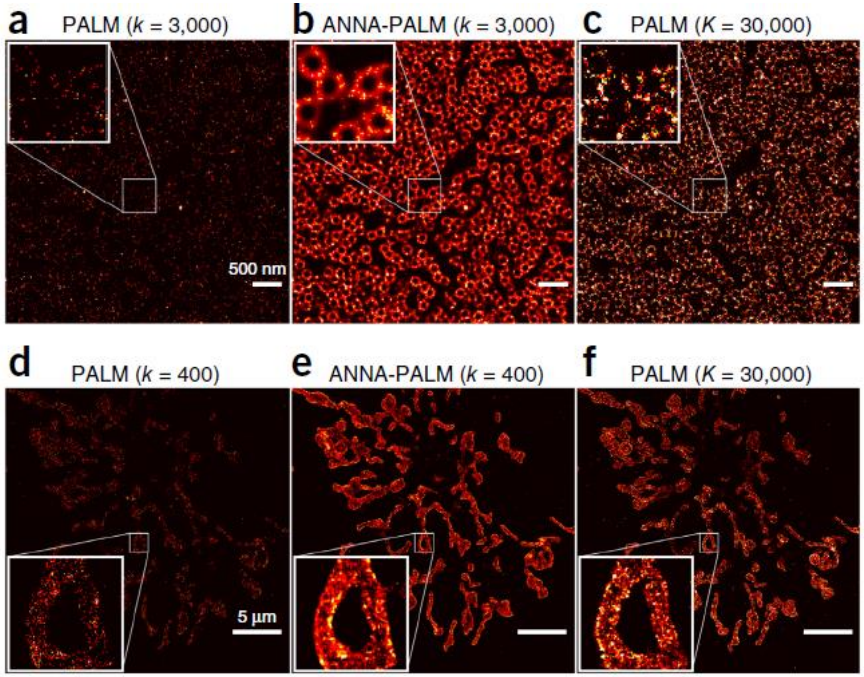
Super-segger



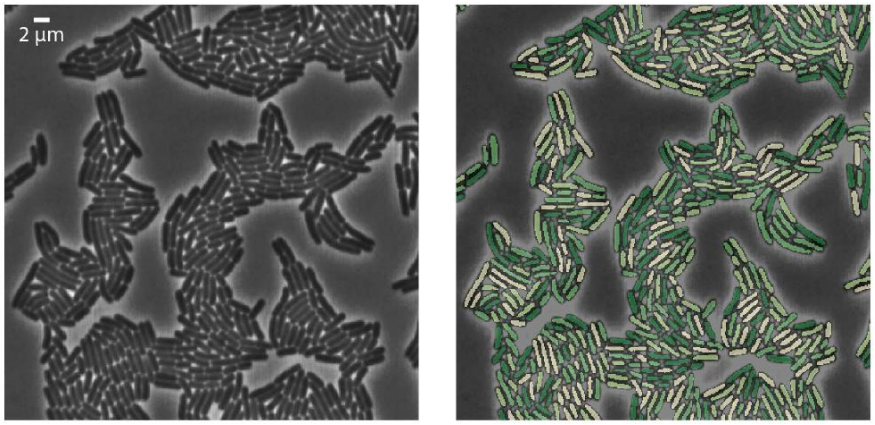
Even on a good quality phase-contrast image, >5% of the cells are not properly segmented using SuperSegger.

Stylianidou et al. 2016. Mol. Microbiol.

Deep-Learning applications for microscopy



Weigert et al. 2017. BioRxiv - Ouyang et al. 2018. Nat. Biotech.



Van Valen et al. 2016. PLOS Computational Biol.

Deep learning is a type of machine learning that has been applied to microscopy in order to :

- Improve super-resolution PALM/STORM image reconstruction
- Segment cells
- Perform filtering on low-signal images
- etc.

Outline

I. Why do we want to used deep learning?

II. Brief history of deep learning

III. Short introduction on neural network (NN)

IV. Convolutional neural network (CNN) for image classification

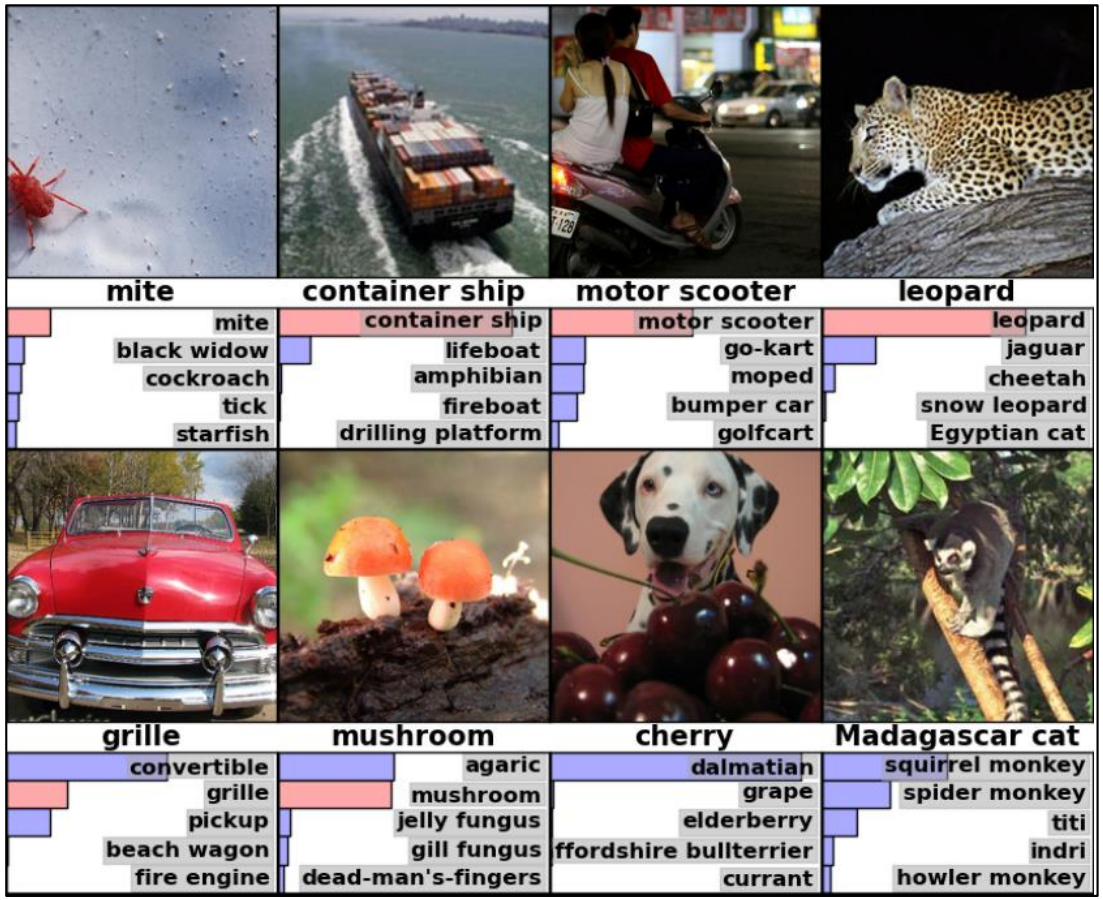
V. Van-Valen network for cell segmentation

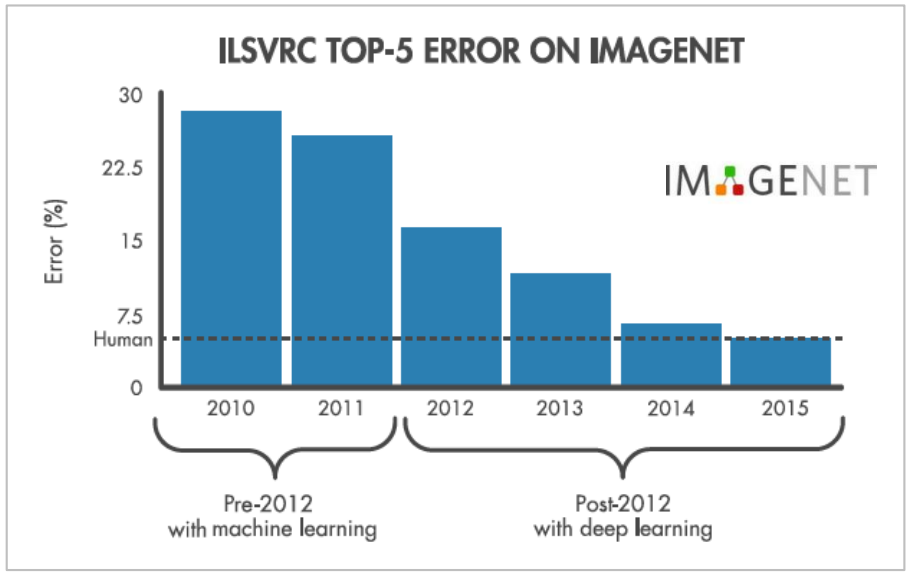
VI. Results

VII. Conclusion & perspectives

Image classification using Deep Learning

Since 2012, several deep learning algorithms based on CNN have been designed for image classification (AlexNet, GoogleNet, VGGNet, etc.) and features extraction.

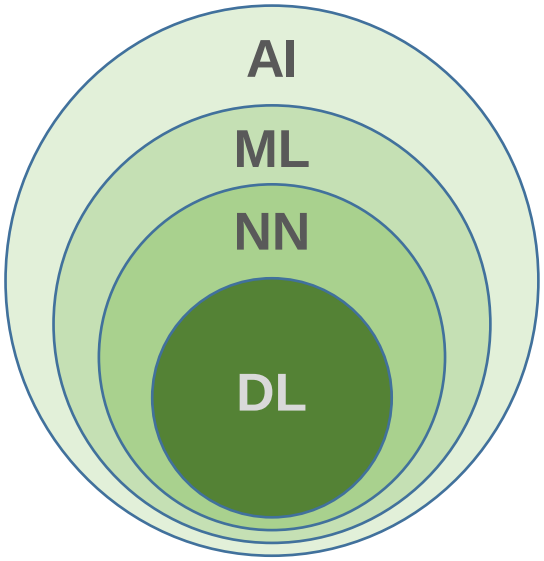




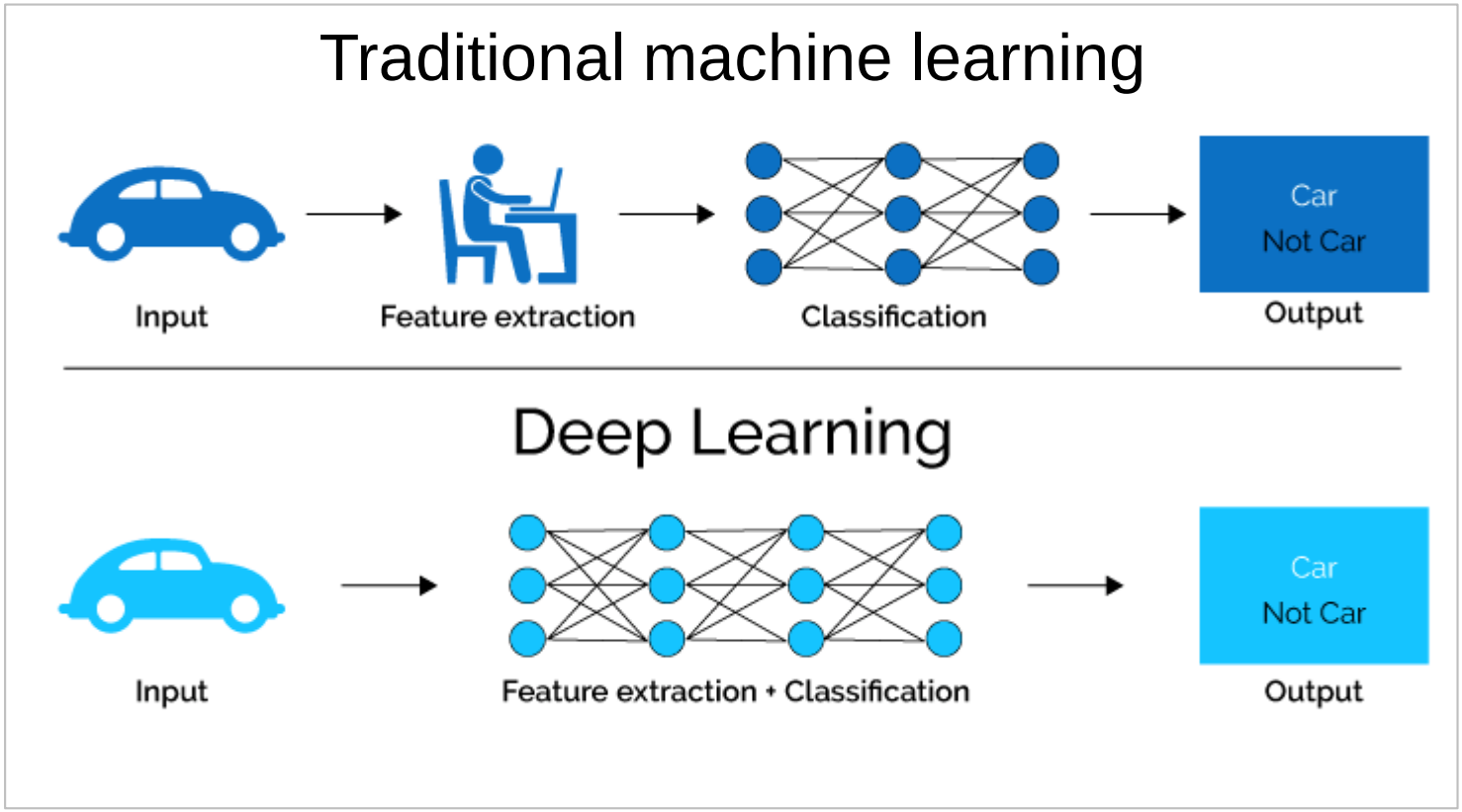
http://image-net.org/index

Krizhevsky – 2012 ImageNet classification with deep convolution neural networks – NIPS 2012

Deep Learning vs. Machine Learning



AI = artificial intelligence
ML = machine learning
NN = neural network
DL = deep learning



For image classification, objects recognition and localization, most algorithms use a Convolutional Neural Network (CNN).

Convolutional Neural Network

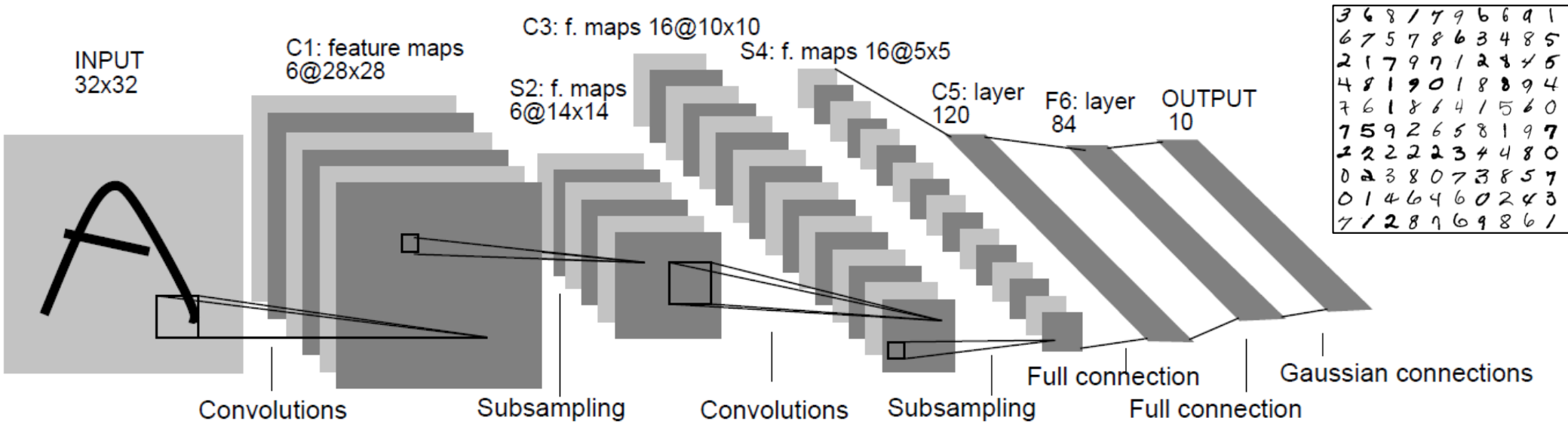
1943 – First description of Artificial Neural Network

1985 – Description of the back-propagation algorithm for training multi-layers NN

1988 – First description of the Convolutional NN

1998 – Application of CNN for digit recognition on ZIP codes

2012 – CNN AlexNet wins the 2012 ImageNet contest



LeCun – 1998 Proc of the IEEE

Outline

I. Why do we want to used deep learning?

II. Brief history of deep learning

III. Short introduction on neural network (NN)

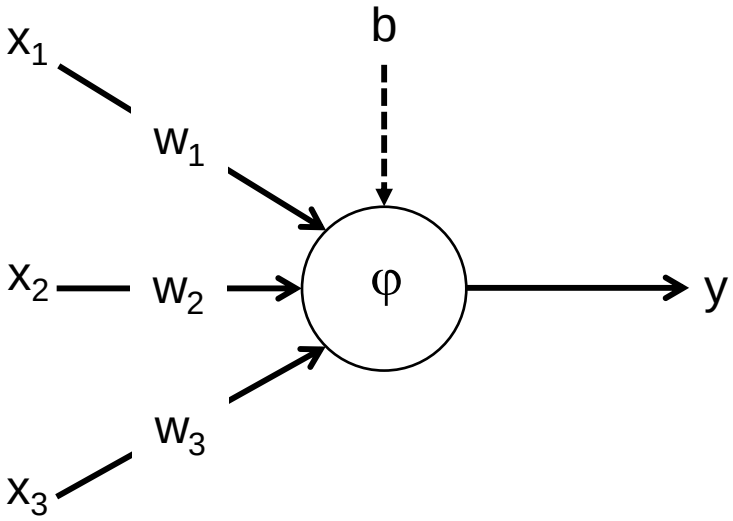
IV. Convolutional neural network (CNN) for image classification

V. Van-Valen network for cell segmentation

VI. Results

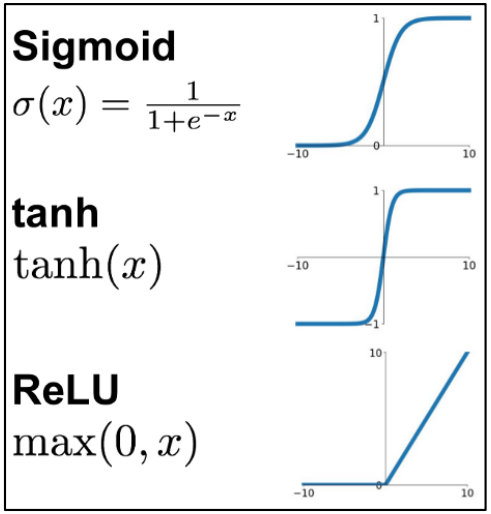
VII. Conclusion & perspectives

Definition of a neuron

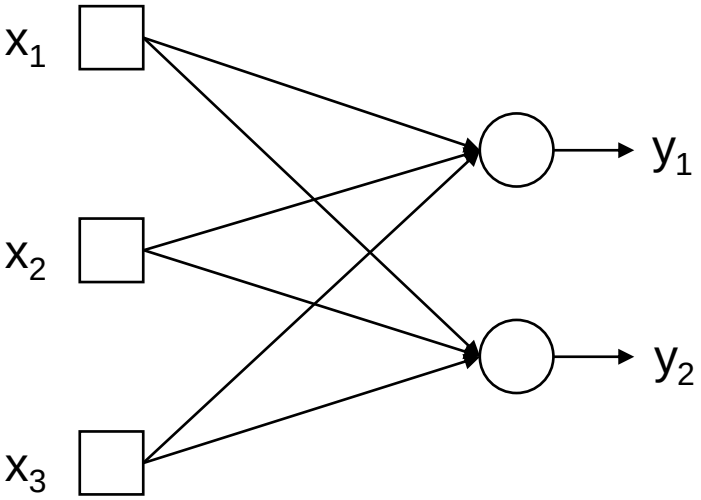


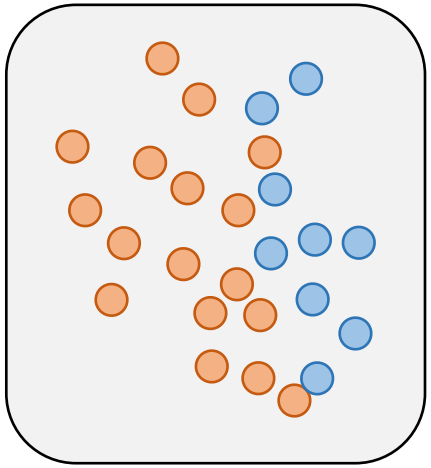
$$v = w_1 * x_1 + w_2 * x_2 + w_3 * x_3$$

$$y = \varphi(v)$$

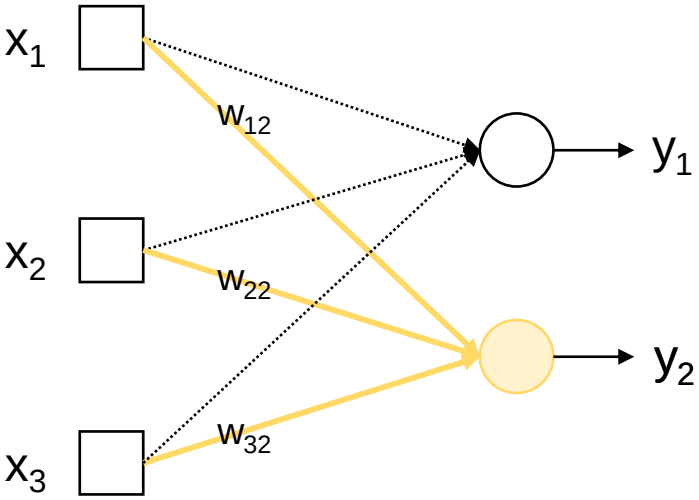


The simplest neural network : 1 layer

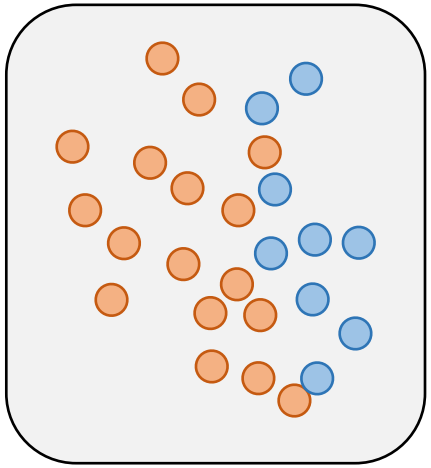




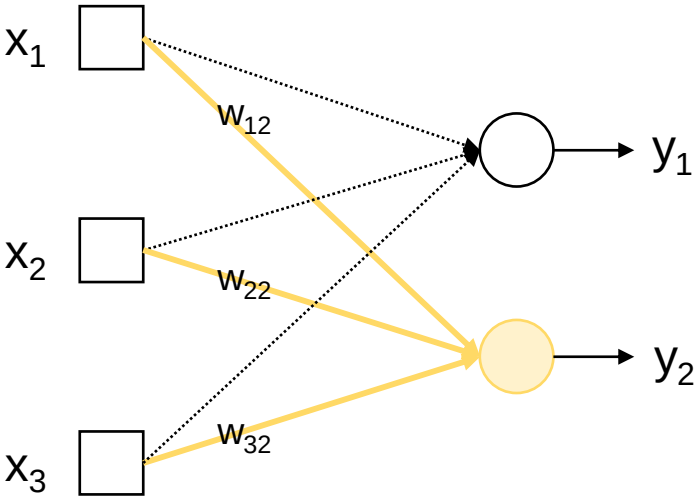
Training of the shallow neural network



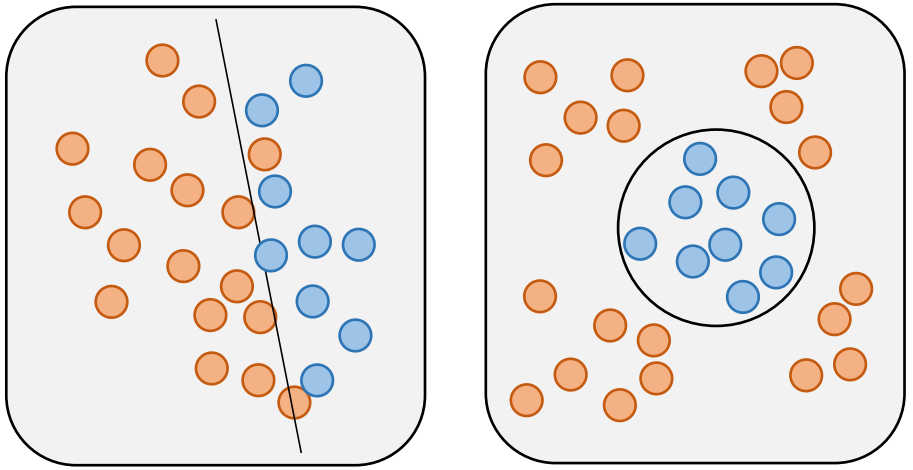
$$\Rightarrow \quad e_i = t_i - y_i \qquad \Rightarrow \begin{cases} w_{ij} \leftarrow w_{ij} + \alpha \delta_i x_j \\ \delta_i = \varphi'(v_i) e_i \end{cases}$$



Training of the shallow neural network

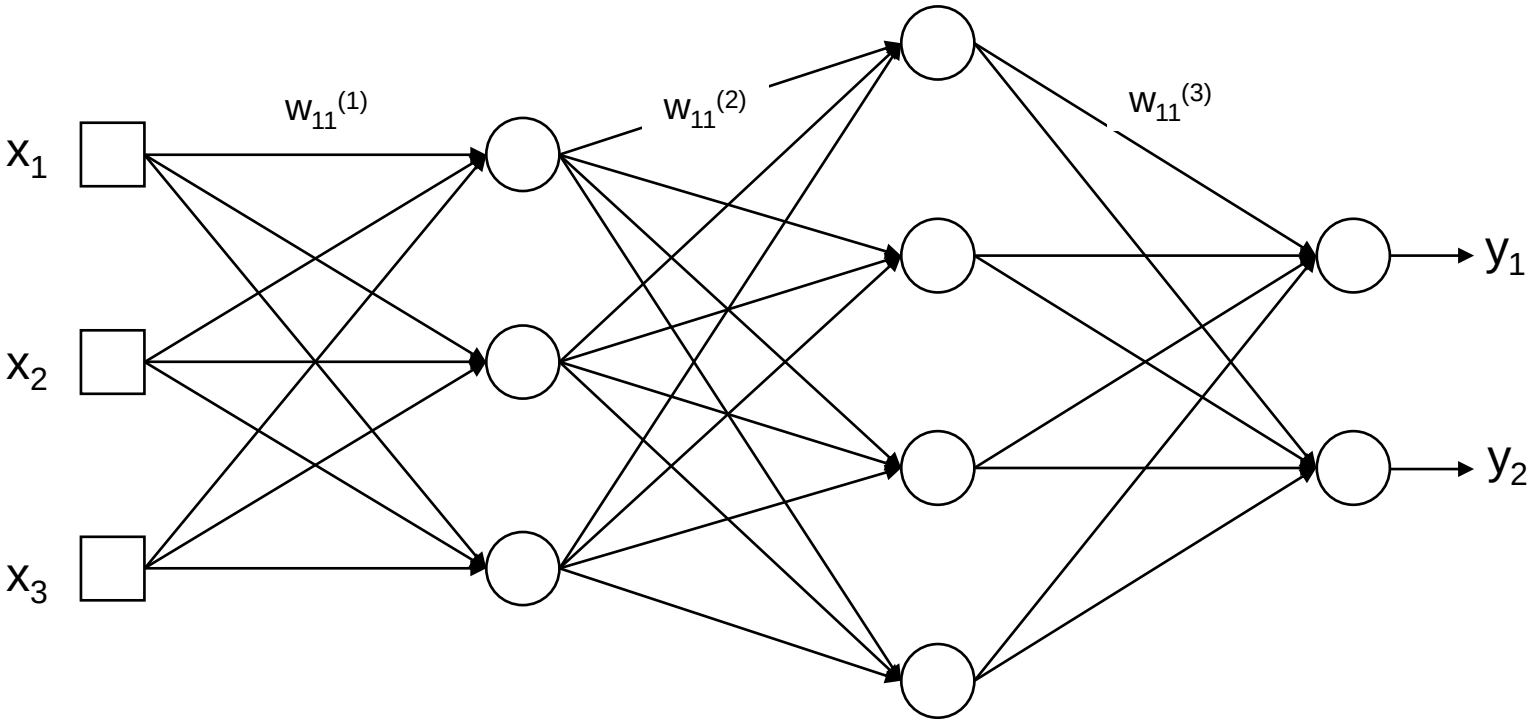


$$\Rightarrow \quad e_i = t_i - y_i \qquad \Rightarrow \begin{cases} w_{ij} \leftarrow w_{ij} + \alpha \delta_i x_j \\ \delta_i = \varphi'(v_i) e_i \end{cases}$$



Linearly separable problem

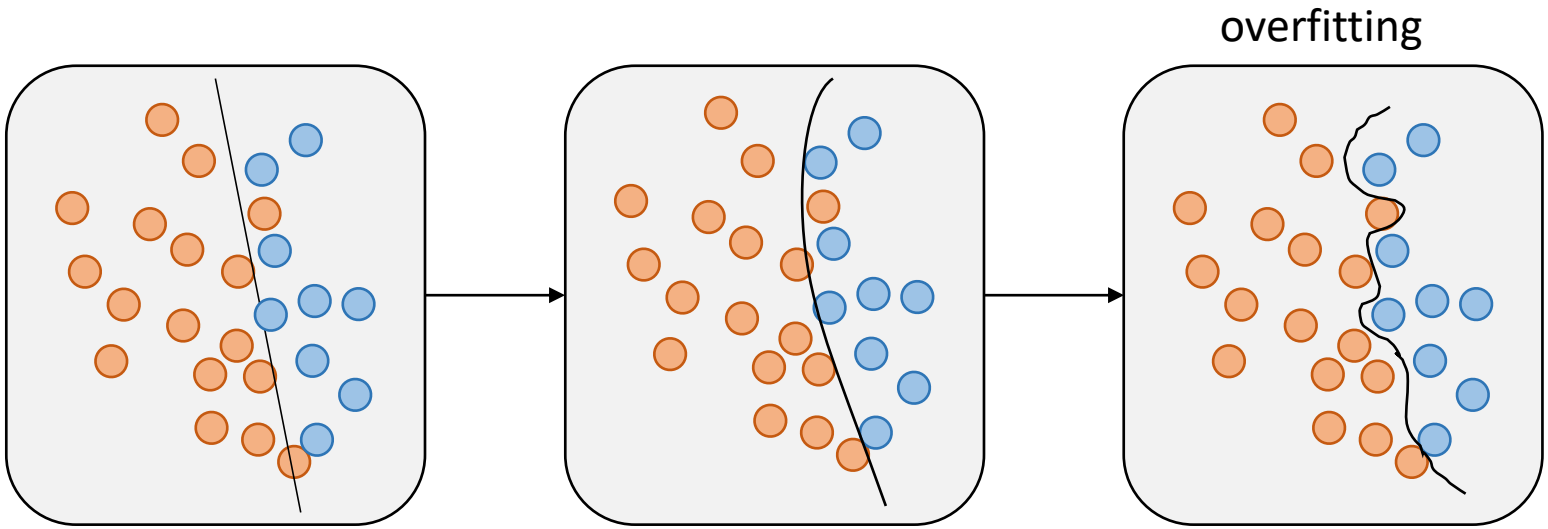
Deep neural network



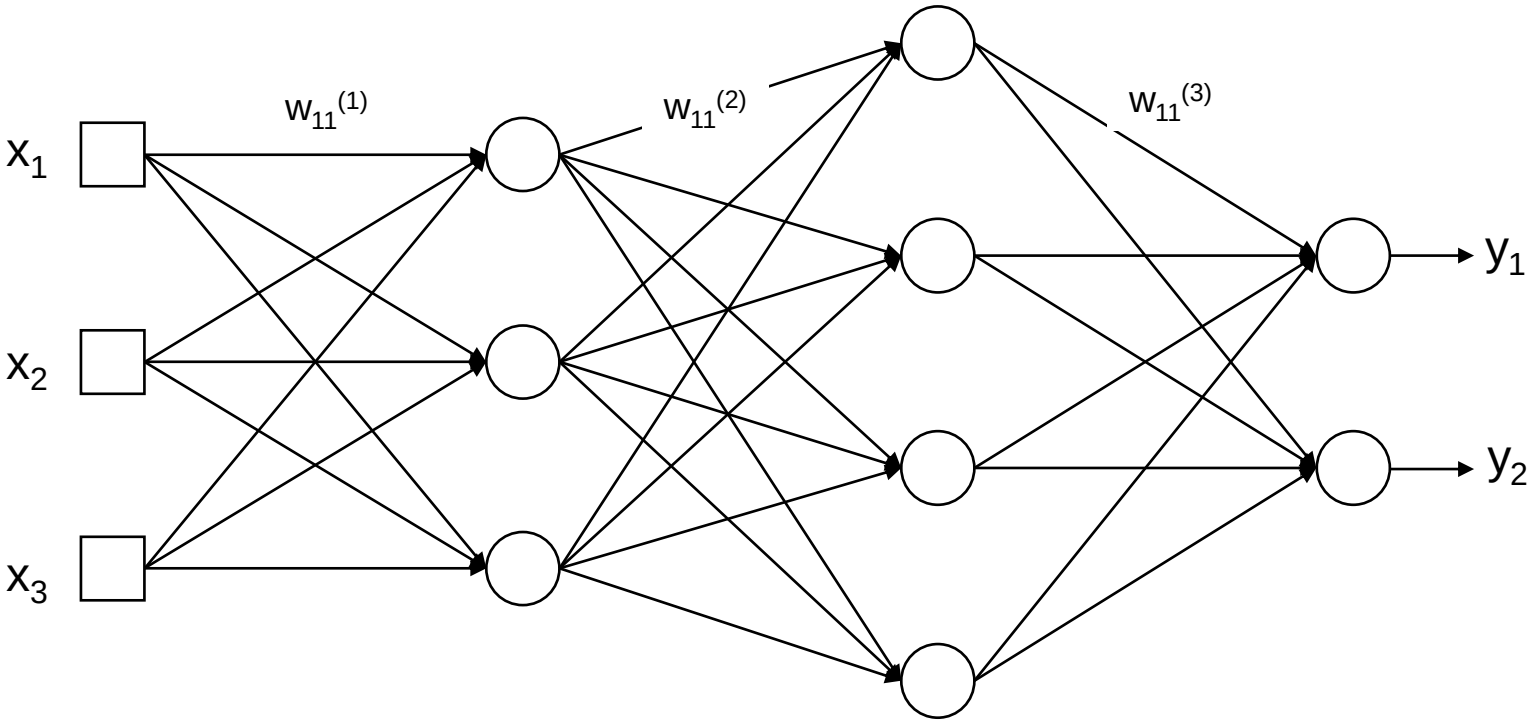
$$e^{(k)} = W^T \delta^{(k+1)}$$

$$\delta^{(k)} = \varphi'\left(v^{(k)}\right) e^{(k)}$$

Error back-propagation



Quick introduction on how neural network works?



$$e^{(k)} = W^T \delta^{(k+1)}$$

$$\delta^{(k)} = \varphi'\left(v^{(k)}\right) e^{(k)}$$

Error back-propagation

Sigmoid

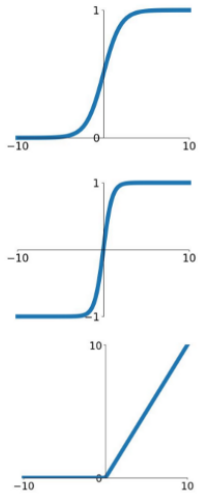
$\sigma(x) = \frac{1}{1+e^{-x}}$

tanh

$\tanh(x)$

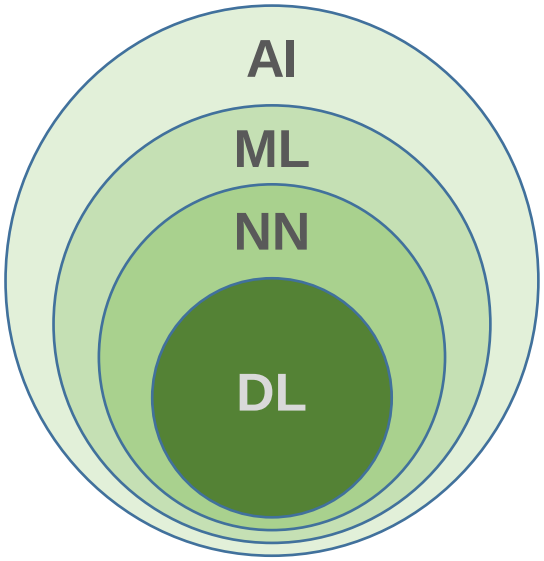
ReLU

$\max(0, x)$

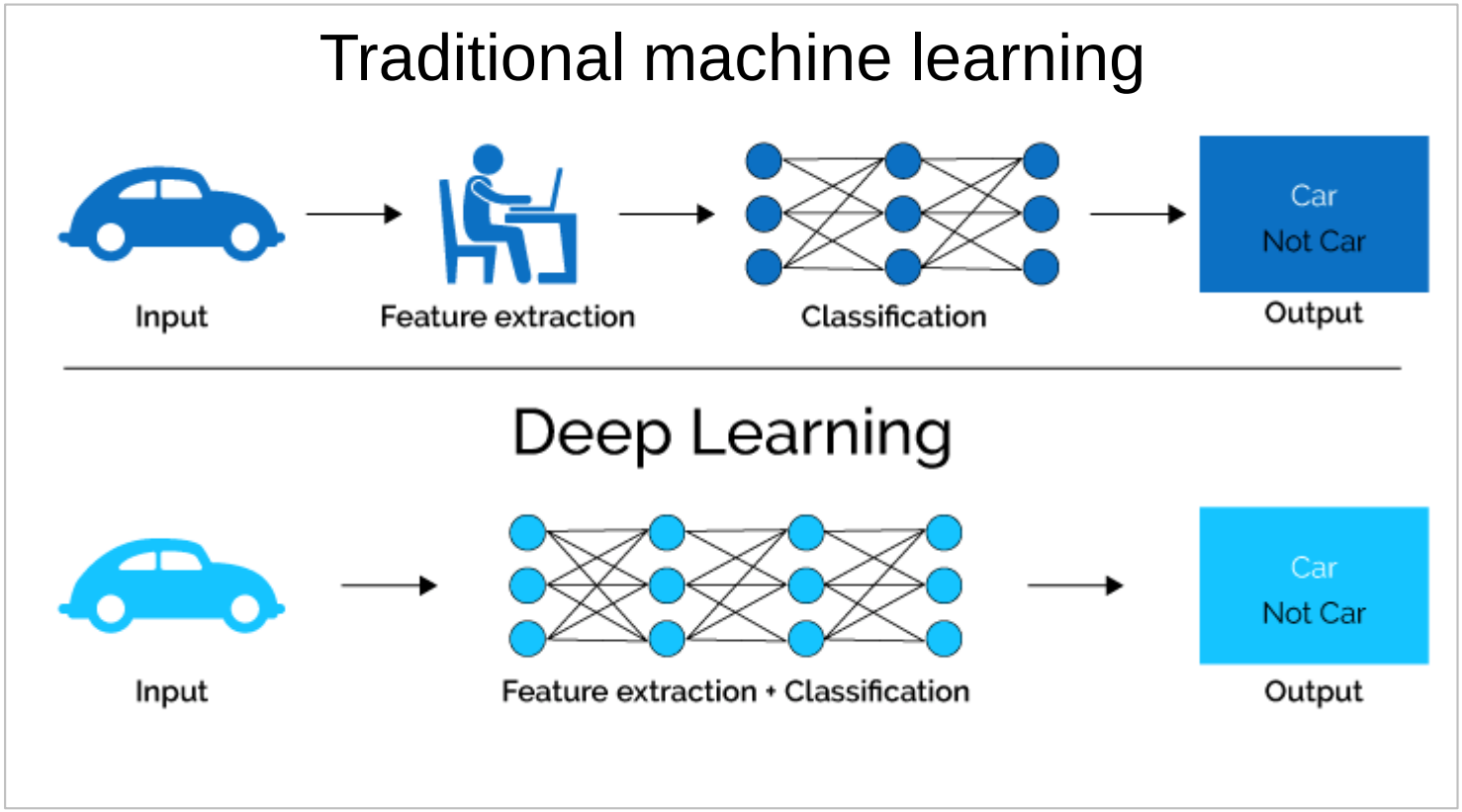


The ReLU activation function is used in most of the recent NN as it helps prevent the **vanishing gradient** issue.

Deep Learning vs. Machine Learning



AI = artificial intelligence
ML = machine learning
NN = neural network
DL = deep learning



For image classification, objects recognition and localization, most algorithms use a Convolutional Neural Network (CNN).

Outline

I. Why do we want to used deep learning?

II. Brief history of deep learning

III. Short introduction on neural network (NN)

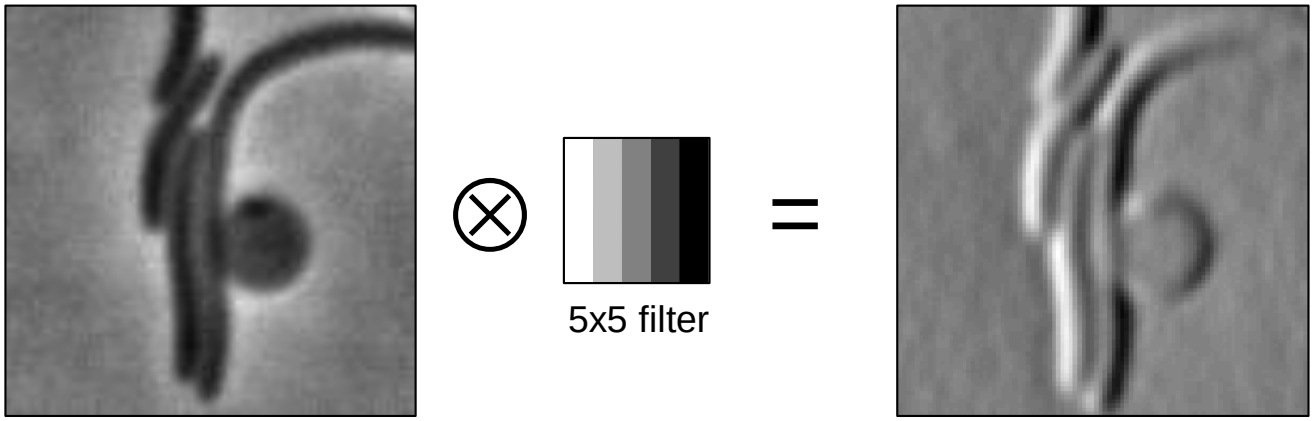
IV. Convolutional neural network (CNN) for image classification

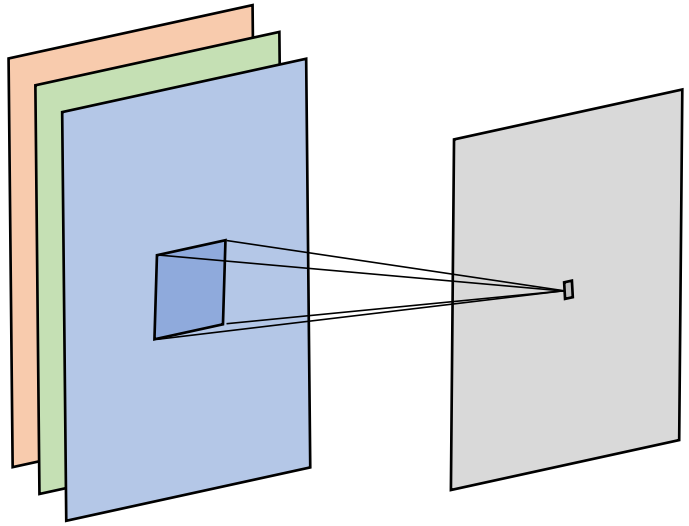
V. Van-Valen network for cell segmentation

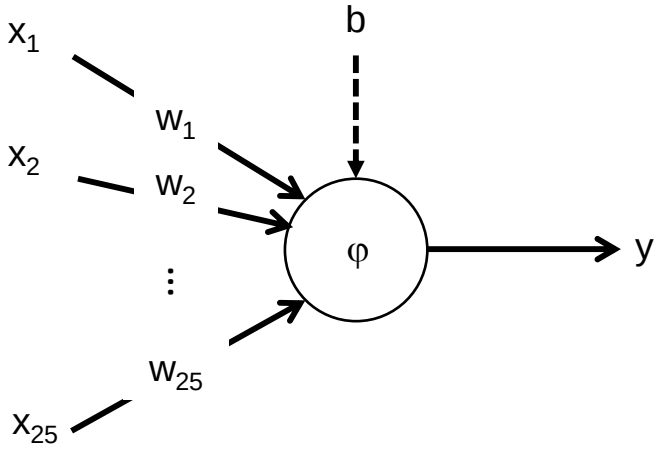
VI. Results

VII. Conclusion & perspectives

Convolutional neural network

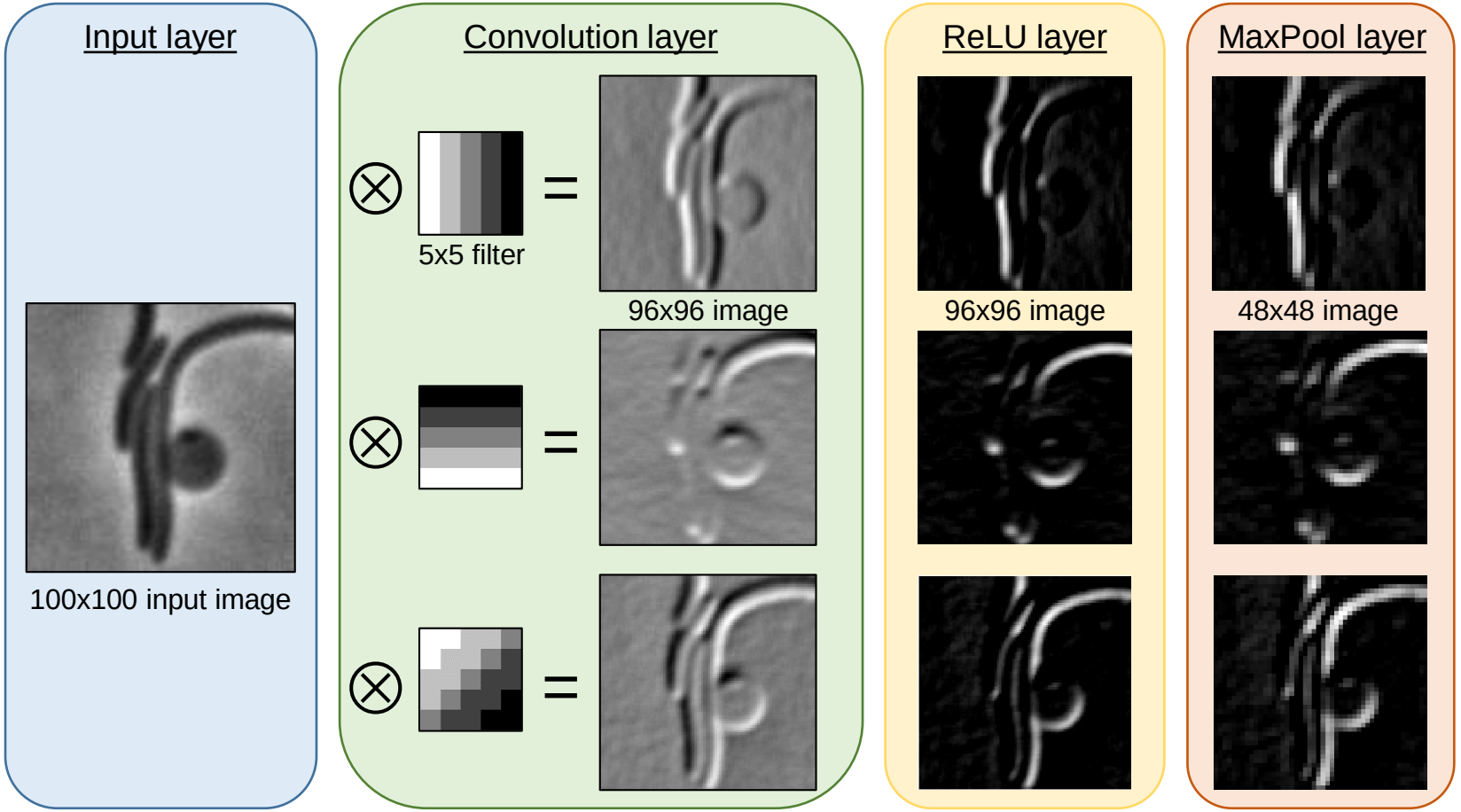


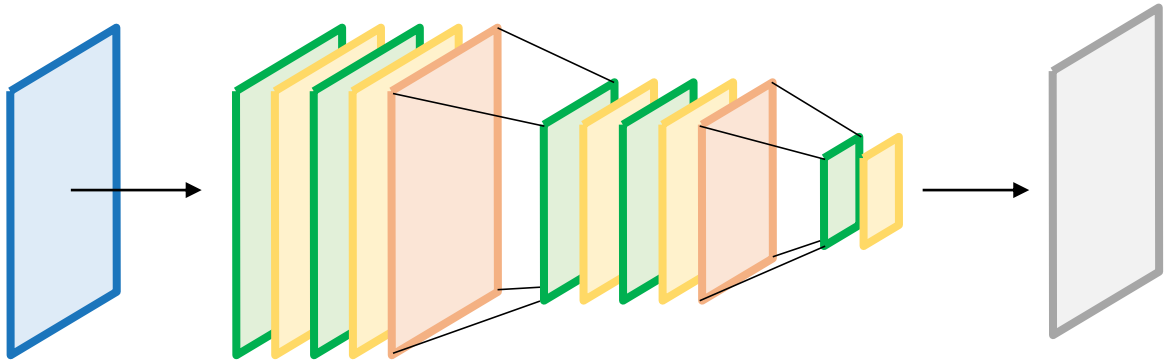




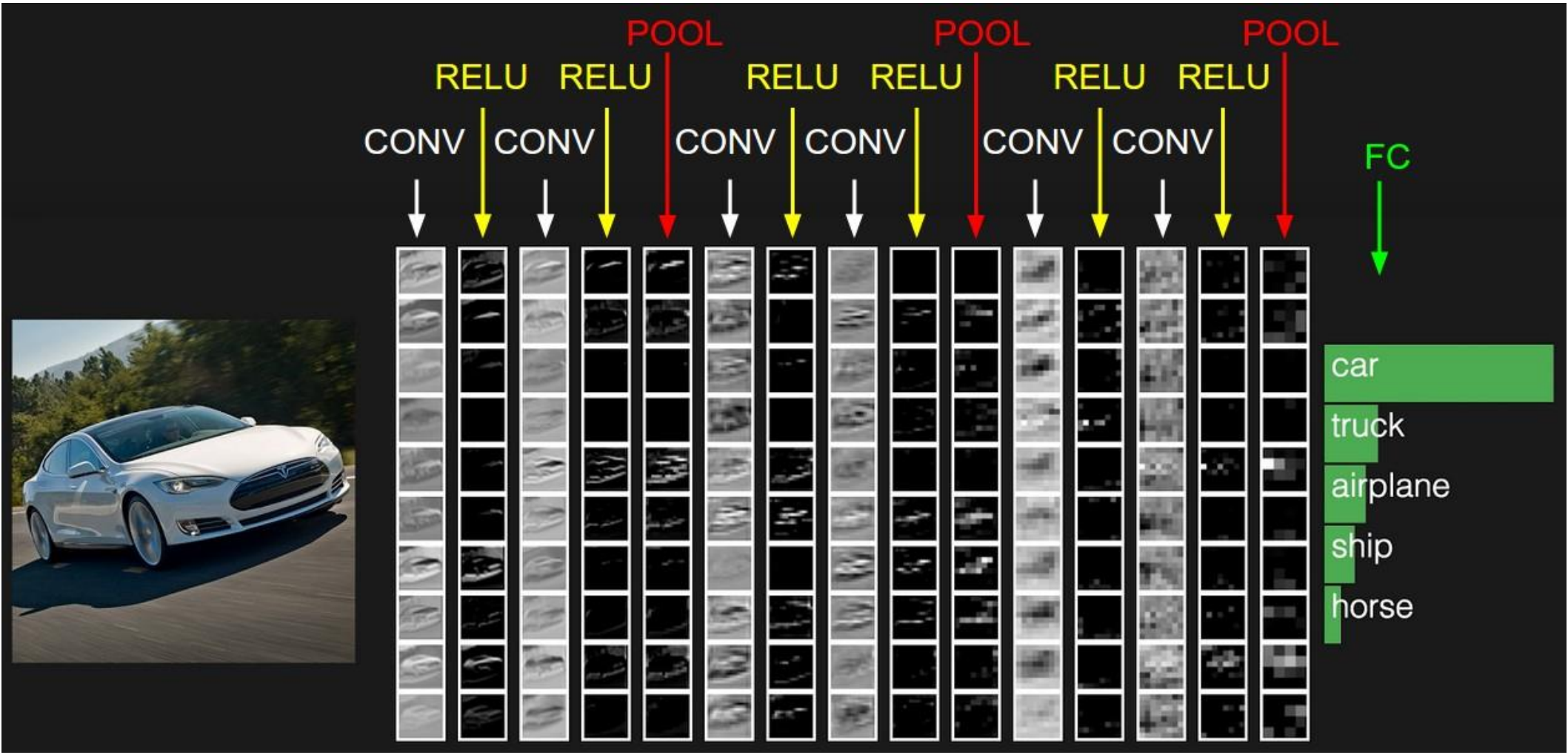
$$v = w_1 * x_1 + w_2 * x_2 + \ ... + \ w_{25} * x_{25}$$
$$y = \ \varphi(v)$$

Convolutional neural network



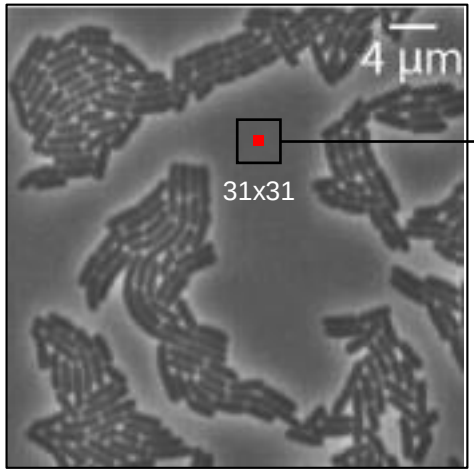


Convolutional neural network



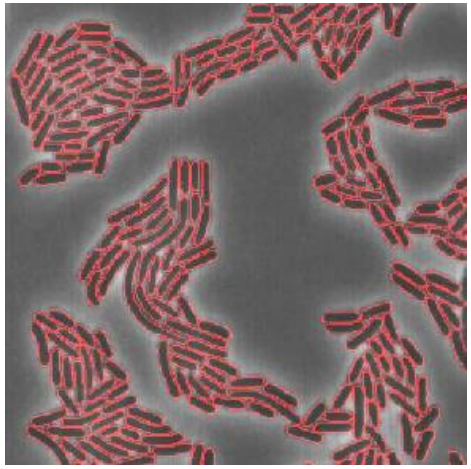
http://cs231n.github.io/convolutional-networks/

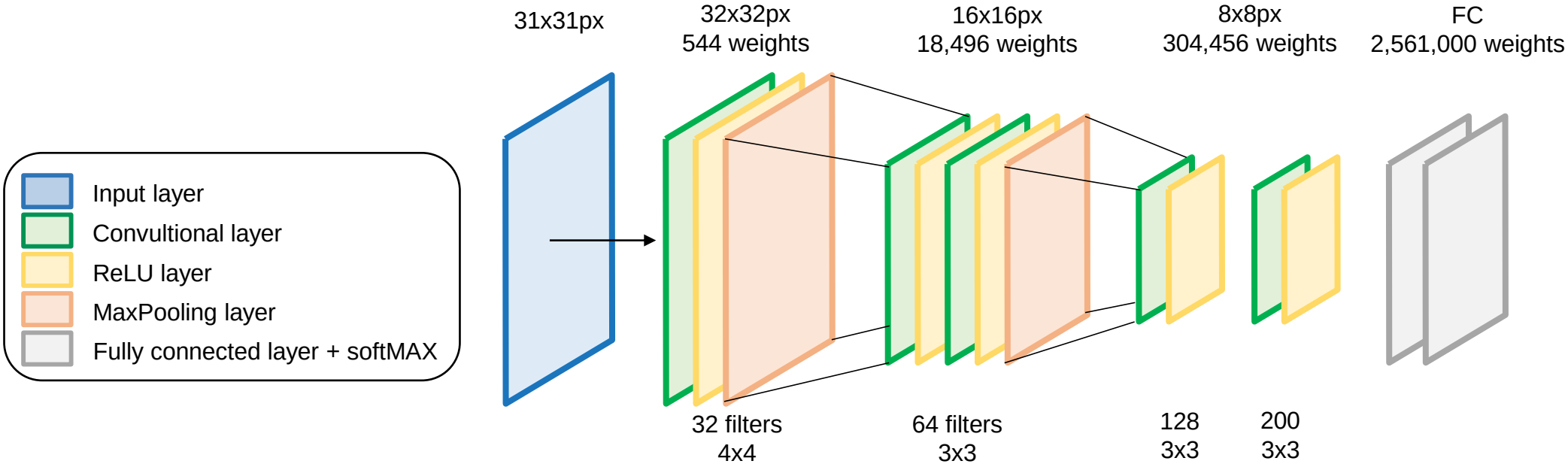
Convolutional neural network applied to cell segmentation



CNN

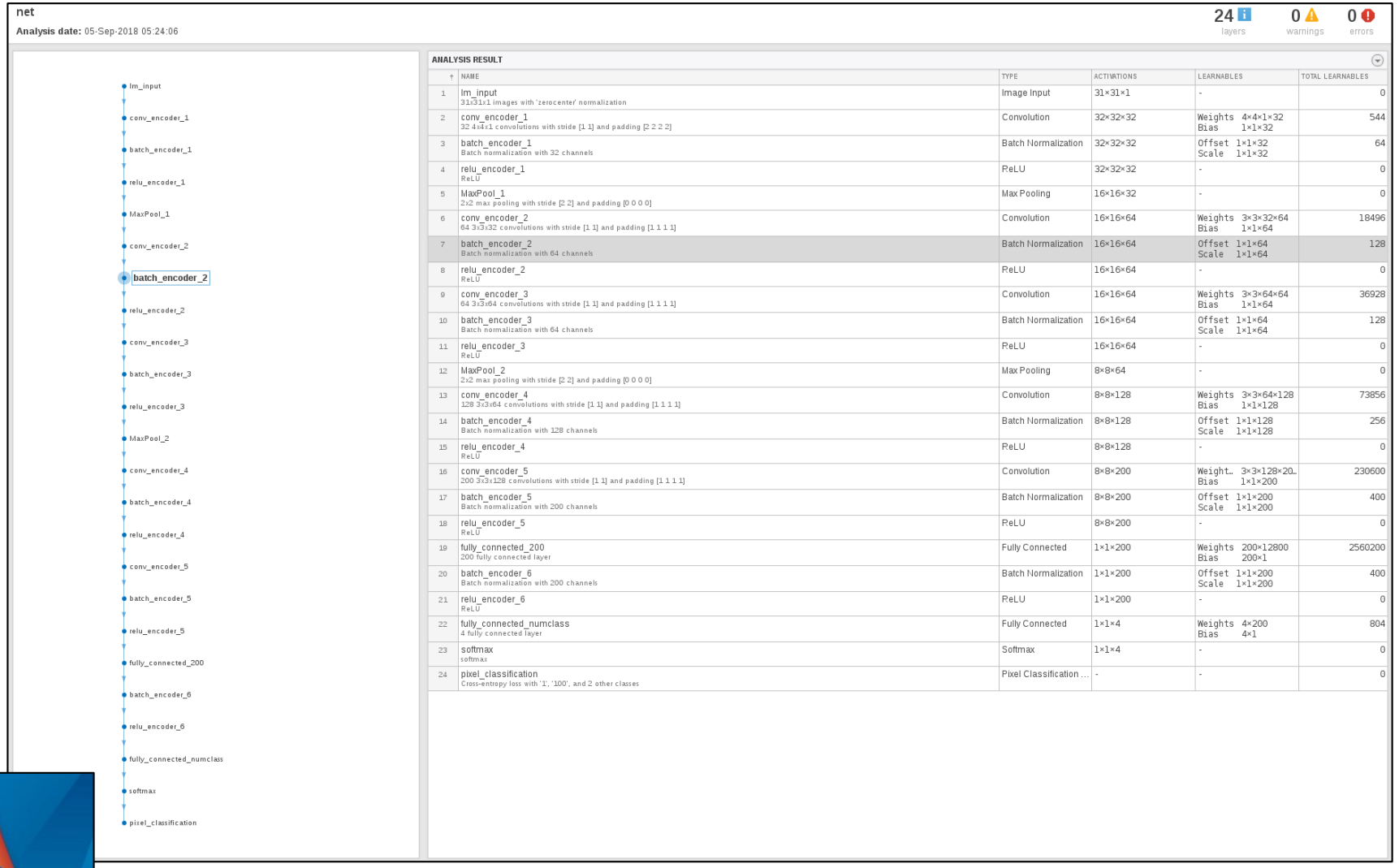
- Cell boundary
- Cell interior
- Background





Van Valen et al. 2016. Plos Computational Biology

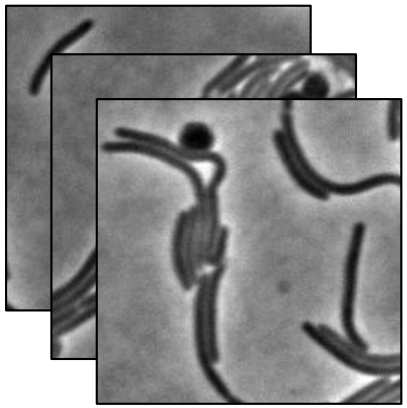
Van-Valen semantic CNN with Matlab



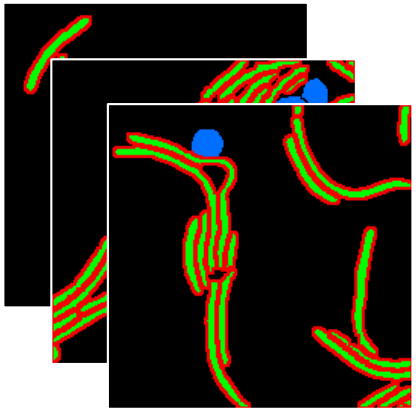
	NAME	TYPE	ACTIVATIONS	LEARNABLES	TOTAL LEARNABLES
1	Im_input 31×31×1 images with 'zerocenter' normalization	Image Input	31×31×1	-	0
2	conv_encoder_1 32 4×4×1 convolutions with stride [1 1] and padding [2 2 2 2]	Convolution	32×32×32	Weights 4×4×1×32 Bias 1×1×32	544
3	batch_encoder_1 Batch normalization with 32 channels	Batch Normalization	32×32×32	Offset 1×1×32 Scale 1×1×32	64
4	relu_encoder_1 ReLU	ReLU	32×32×32	-	0
5	MaxPool_1 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	16×16×32	-	0
6	conv_encoder_2 64 3×3×32 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	16×16×64	Weights 3×3×32×64 Bias 1×1×64	18496
7	batch_encoder_2 Batch normalization with 64 channels	Batch Normalization	16×16×64	Offset 1×1×64 Scale 1×1×64	128
8	relu_encoder_2 ReLU	ReLU	16×16×64	-	0
9	conv_encoder_3 64 3×3×64 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	16×16×64	Weights 3×3×64×64 Bias 1×1×64	36928
10	batch_encoder_3 Batch normalization with 64 channels	Batch Normalization	16×16×64	Offset 1×1×64 Scale 1×1×64	128
11	relu_encoder_3 ReLU	ReLU	16×16×64	-	0
12	MaxPool_2 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	8×8×64	-	0
13	conv_encoder_4 128 3×3×64 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	8×8×128	Weights 3×3×64×128 Bias 1×1×128	73856
14	batch_encoder_4 Batch normalization with 128 channels	Batch Normalization	8×8×128	Offset 1×1×128 Scale 1×1×128	256
15	relu_encoder_4 ReLU	ReLU	8×8×128	-	0
16	conv_encoder_5 200 3×3×128 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	8×8×200	Weight… 3×3×128×20… Bias 1×1×200	230600
17	batch_encoder_5 Batch normalization with 200 channels	Batch Normalization	8×8×200	Offset 1×1×200 Scale 1×1×200	400
18	relu_encoder_5 ReLU	ReLU	8×8×200	-	0
19	fully_connected_200 200 fully connected layer	Fully Connected	1×1×200	Weights 200×12800 Bias 200×1	2560200
20	batch_encoder_6 Batch normalization with 200 channels	Batch Normalization	1×1×200	Offset 1×1×200 Scale 1×1×200	400
21	relu_encoder_6 ReLU	ReLU	1×1×200	-	0
22	fully_connected_numclass 4 fully connected layer	Fully Connected	1×1×4	Weights 4×200 Bias 4×1	804
23	softmax softmax	Softmax	1×1×4	-	0
24	pixel_classification Cross-entropy loss with '1', '100', and 2 other classes	Pixel Classification …	-	-	0

How do we train this supervised deep learning network?

Test images



200x200px

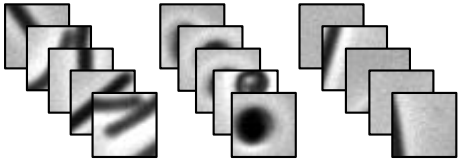


Ground truth

Step 1 : Manual labelling

100 images from 3 different experiments
On each image the four classes are observed

Create the training/testing set



31x31

Step 2 : Training & Testing sets

Training/testing set of 31x31 images, with the same number of images for each class.

- Training set : ~50,000 images/class
- Testing set : ~10,000

Training rules

- Background
- Cell outline
- Cell body
- Dirt / Dead cells

Step 3 : Training (2,5h)

After defining the training rules, the training start with the following parameters :

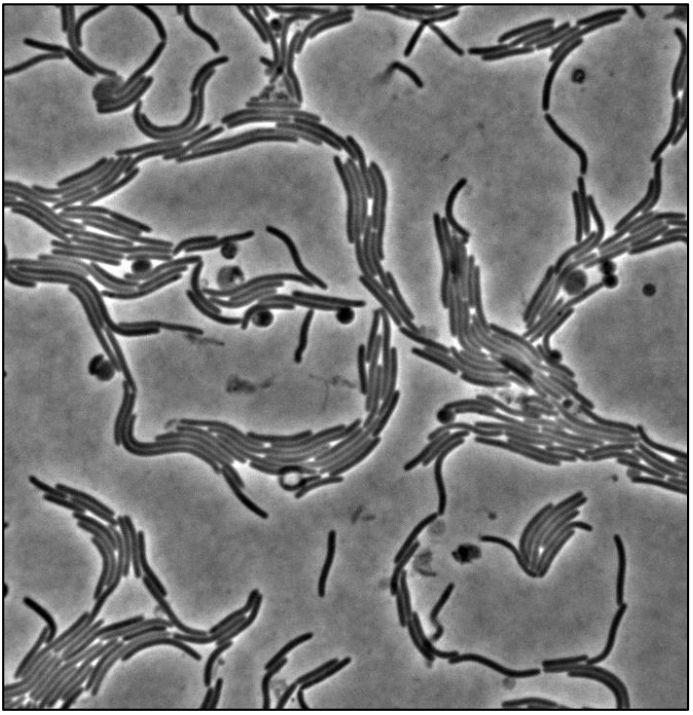
- Stochastic gradient descent method
- Minibatch = 2500
- Epoc = 21

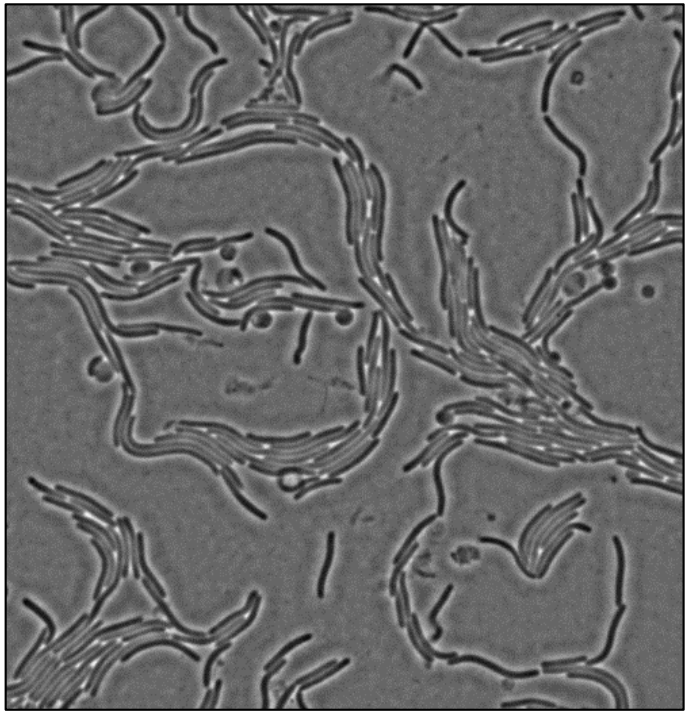
Deep Neural Network

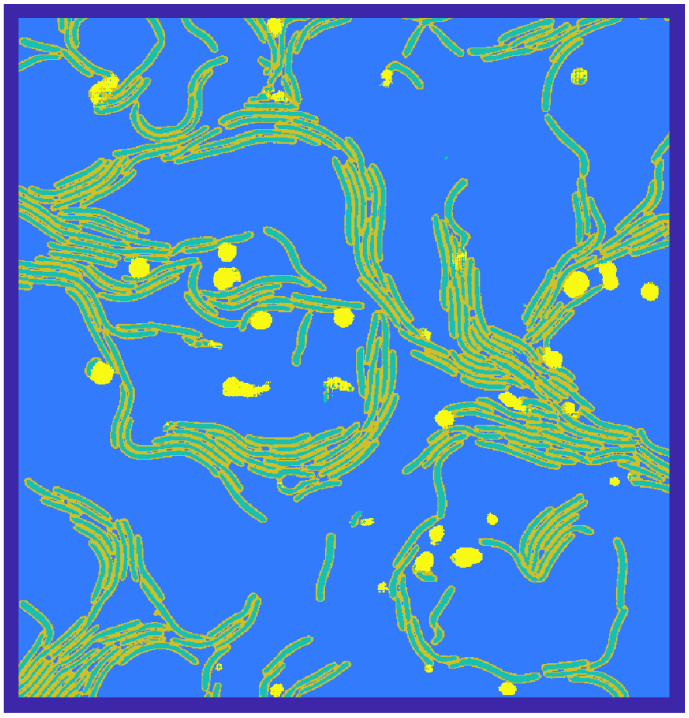
Segmentation of *Myxococcus xanthus* cells

Image normalization

Deep learning network







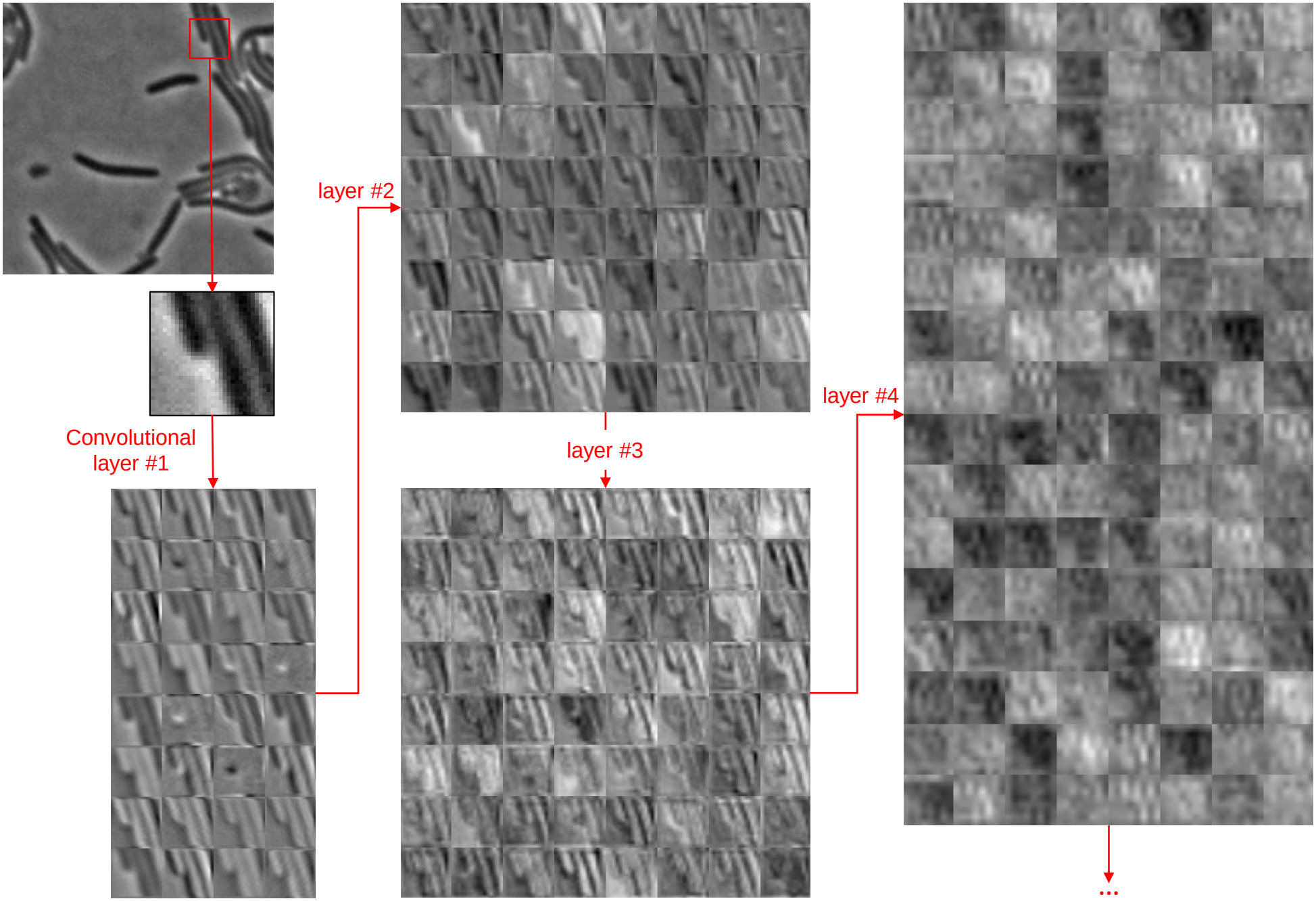
Time-lapse - ΔPilA AglZ-YFP
1000x1000 Phase contrast image

Computation time with 15 cores parallel computing : 4,5min

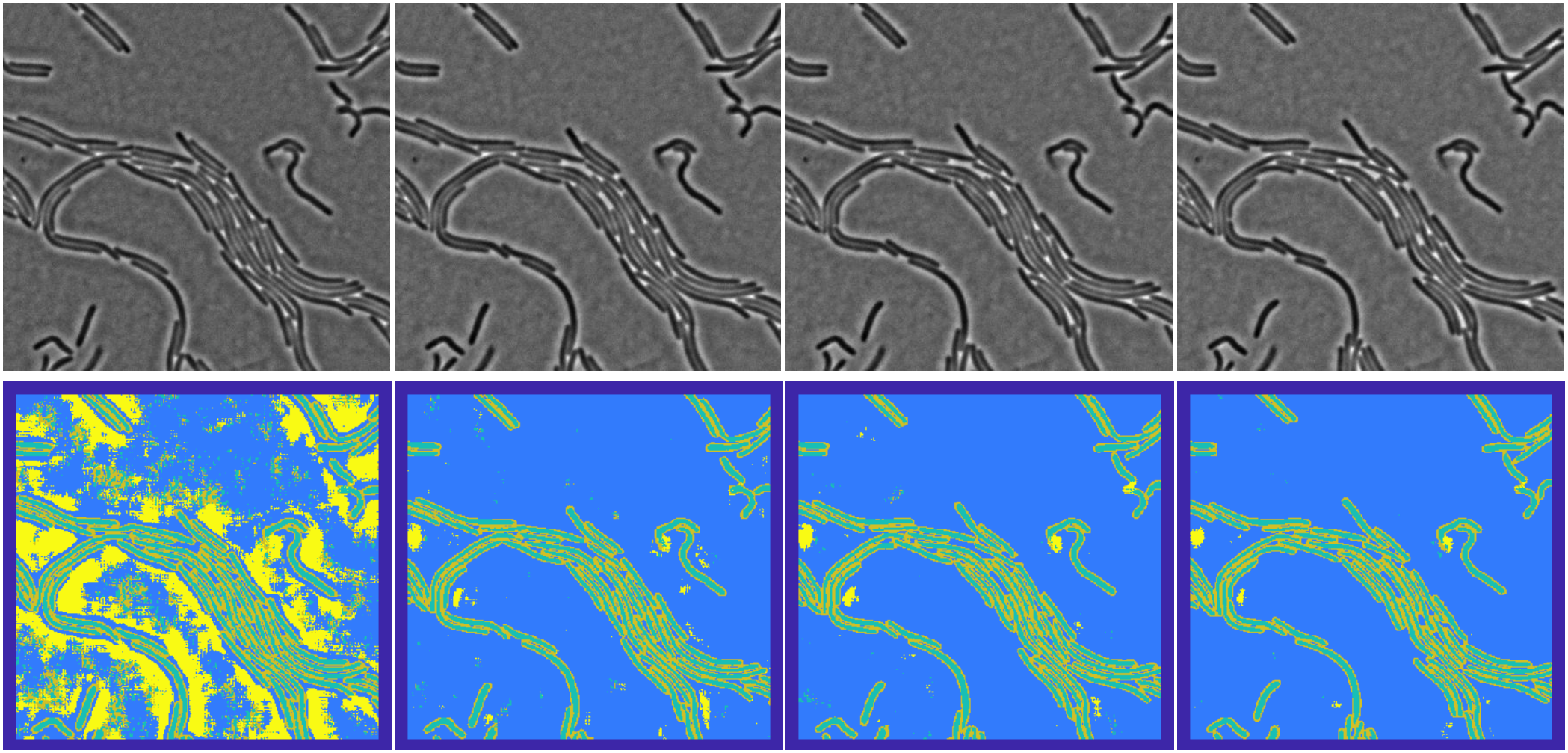
Accuracy for the testing set :

- global accuracy : 93.8 %
- accuracy for « background » : 94.1 %
- accuracy for « dirt » : 99.1 %
- accuracy for « cell inside » : 91 %
- accuracy for « cell contour »:91 %

A glimpse of what is going on « inside »



Limitations of the method

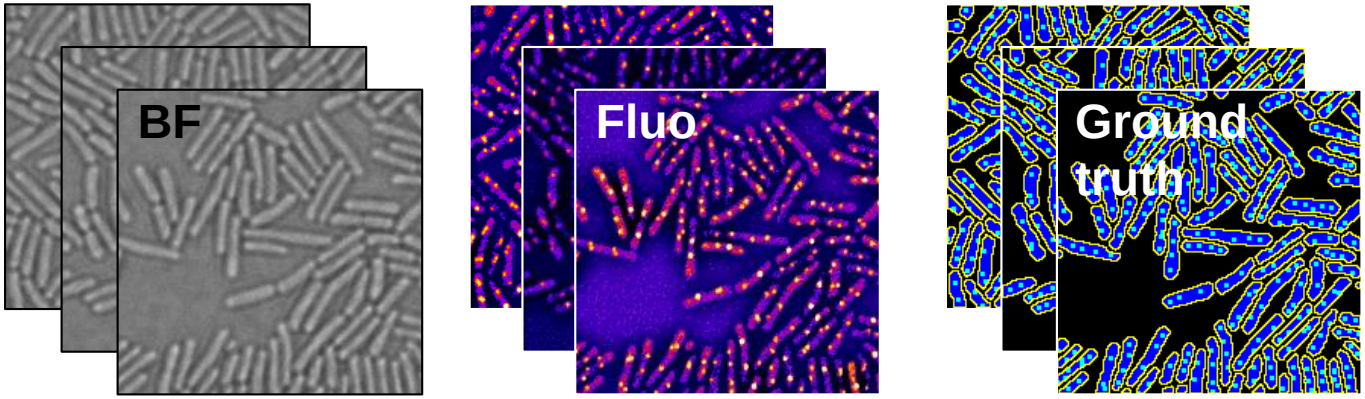


- Class 1 : Background
- Class 2 : Cell outline
- Class 3 : Cell body
- Class 4 : Dirt / Dead cells

For certain images, the classes got mixed up (particularly between classes 1 & 4). The classification might be sensitive to variation in the experimental conditions (intensity, …) → **the experimental conditions needs to be well controlled.**

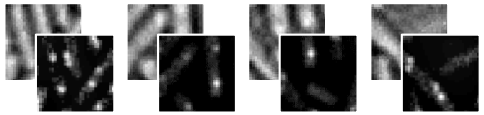
Segmentation of *B.subtilis* : Brightfield + Fluorescence

B.Subtilis - Channel 1 : BF - Channel 2 : Fros Ori-CFP



200x200px

Create the training/testing set



31x31

Training rules

- Background
- Cell outline
- Cell body
- Fluorescent spot

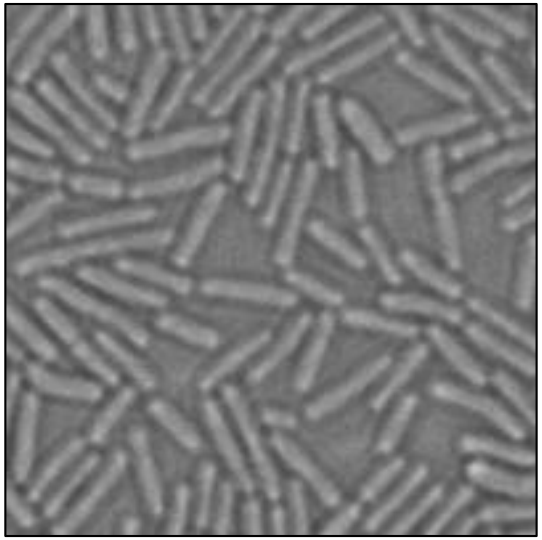
Deep Neural Network

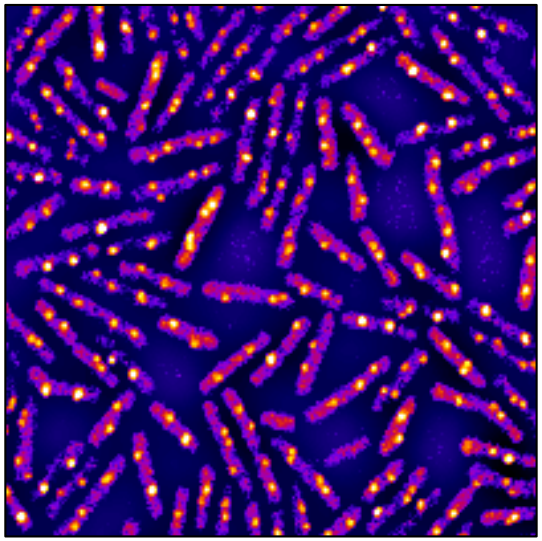
Training & Testing sets

Training/testing set of 31x31 images, with the same number of images for each class.

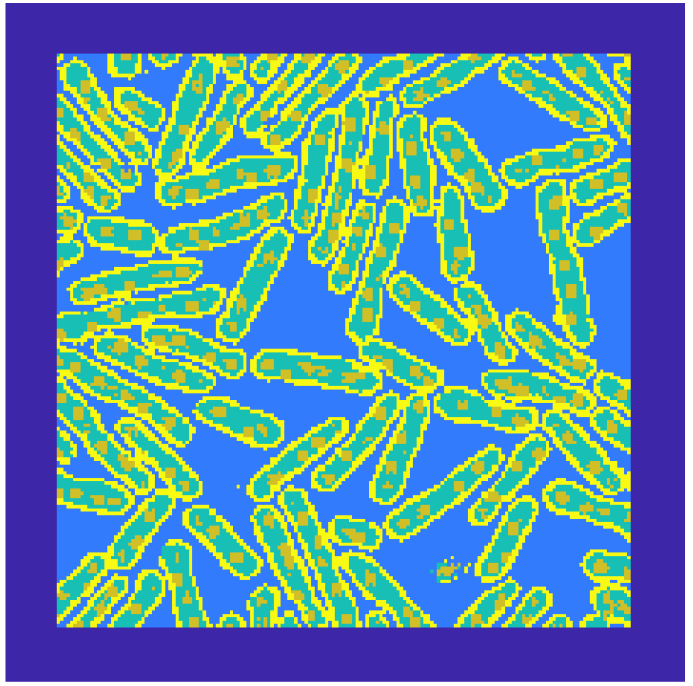
- Training set : ~9,500 images/class
- Testing set : ~2,500 images/class

Segmentation of *B.subtilis* : Brightfield + Fluorescence





CNN



Accuracy for the testing set :

- **Global accuracy : 87.6 %**
- Global accuracy for class1 : 91.6 %
- Global accuracy for class2 : 82.6 %
- Global accuracy for class3 : 91.6 %
- Global accuracy for class4 : 84.5 %

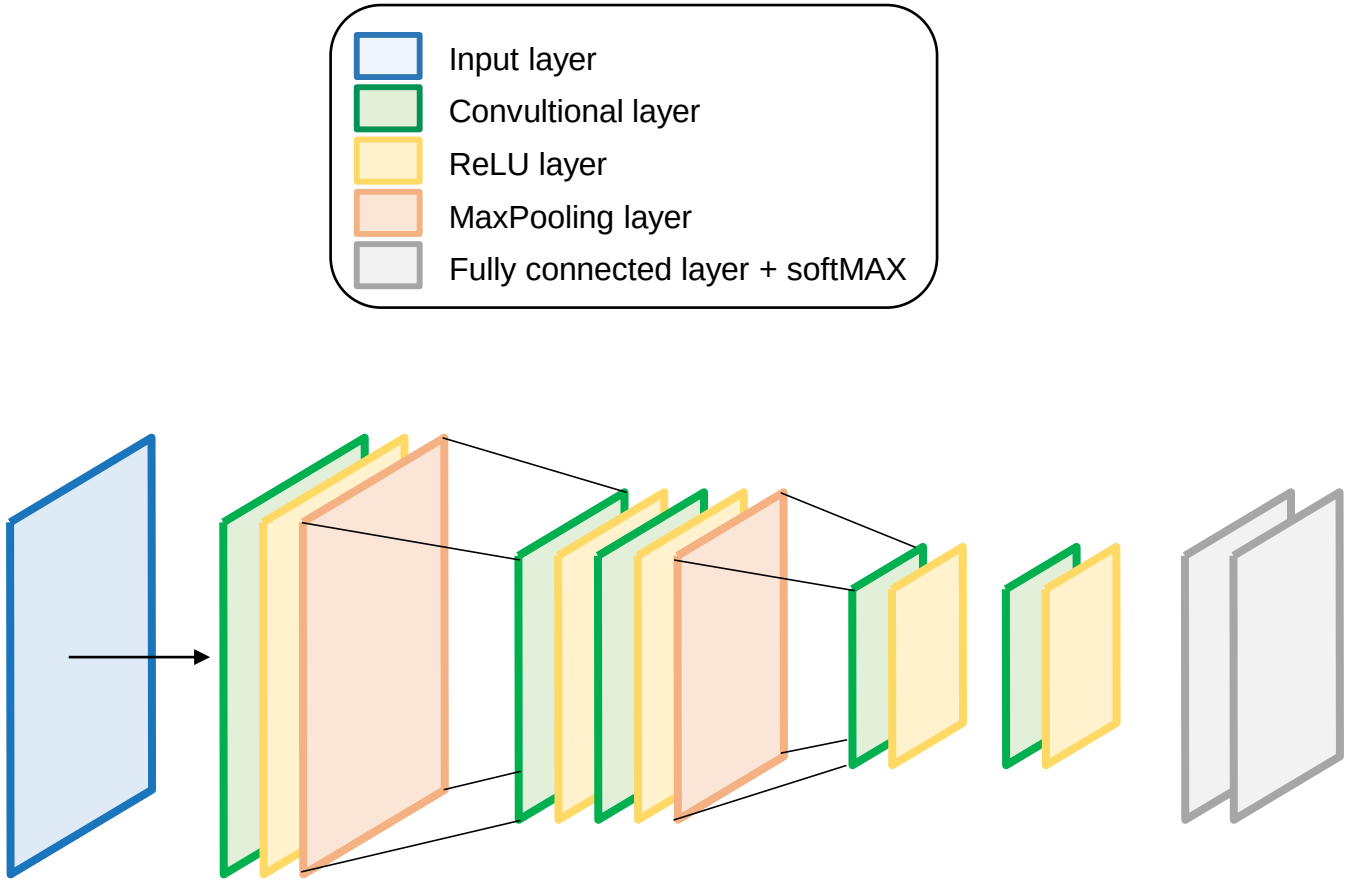
Conclusion and perspectives

- So far, deep learning look like a promising tool for segmentation :
  - ✓ *Myxococcus xanthus* cells on phase contrast images
  - ✓ Bacillus subtilis cells taking into account brightfield and fluorescence images
- The accuracy of segmentation is substancially better as compared to previous results obtained with SuperSegger

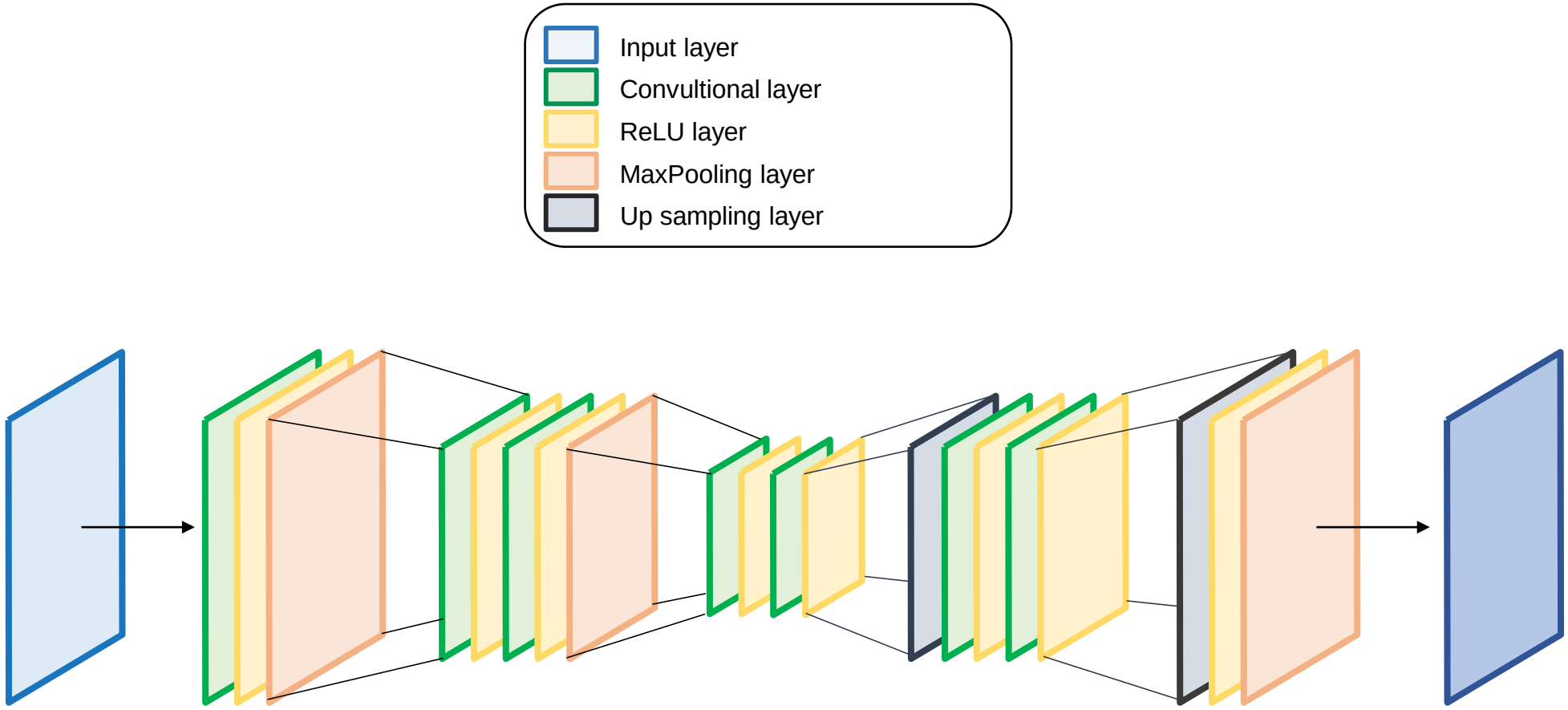
- Improve the computation time – for a 2000x2000 images it takes >15min

  → Fully convolutional network
- How to make the network less susceptible to experimental conditions?

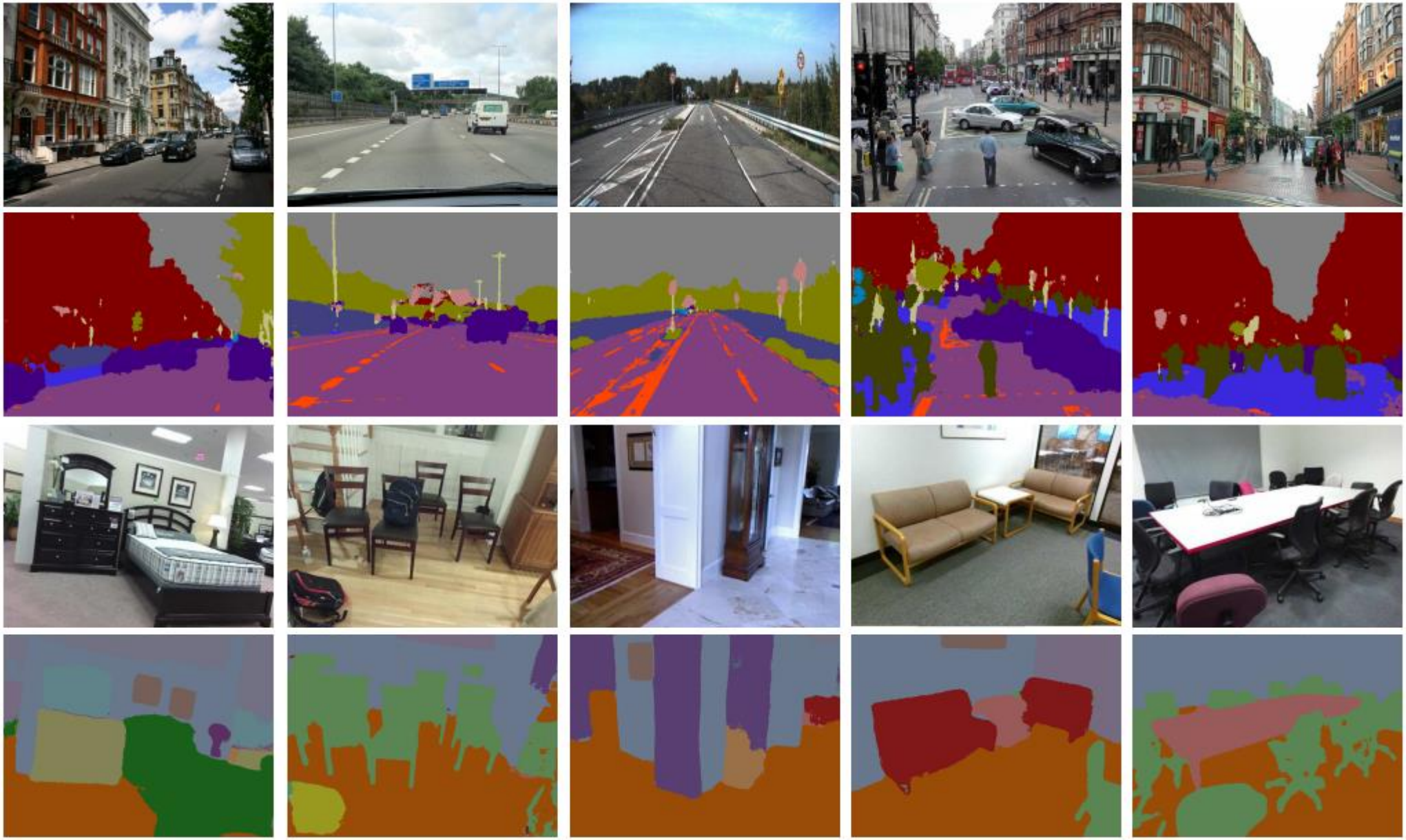
Fully convolutional network



Fully convolutional network

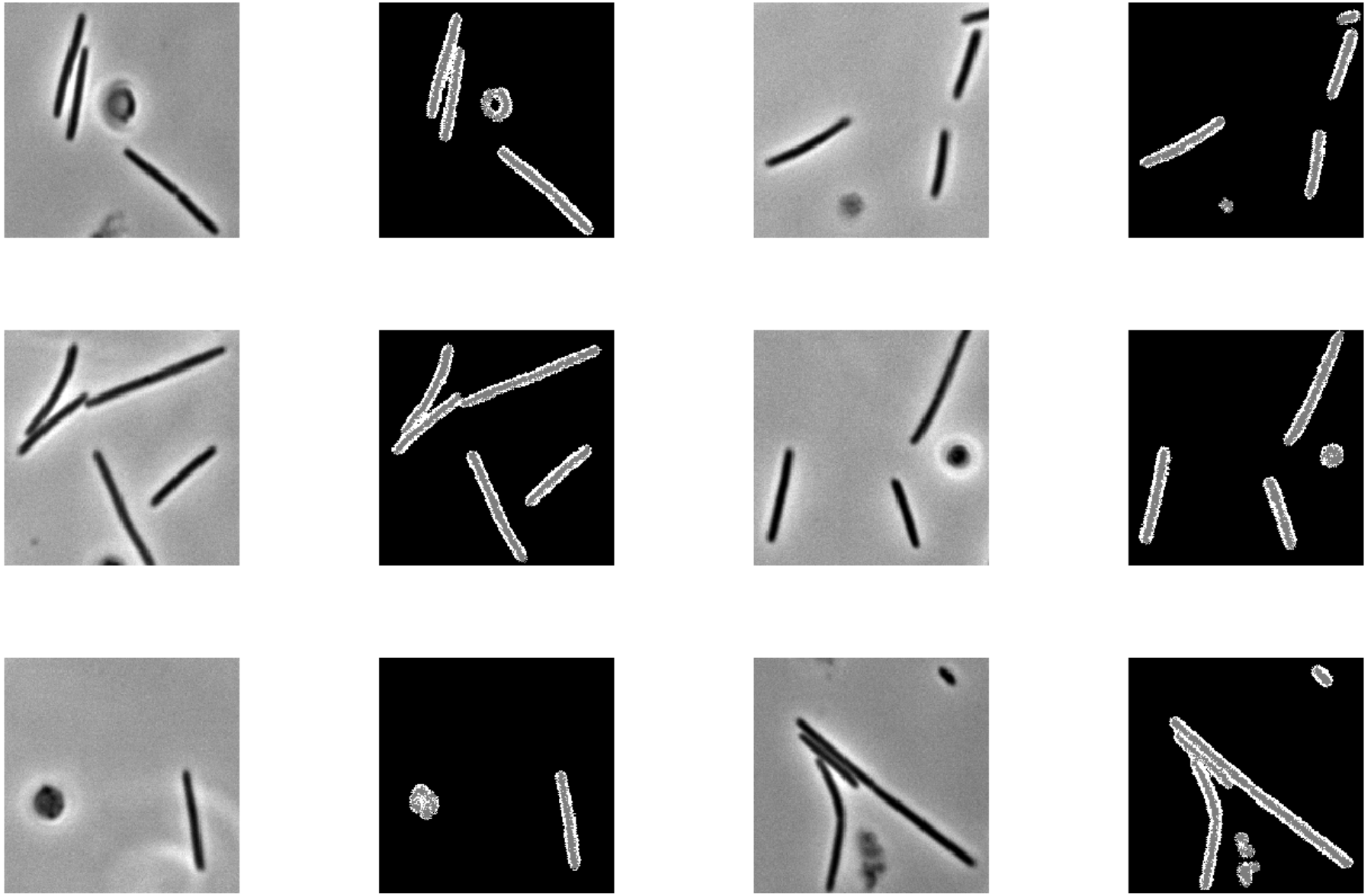


Fully convolutional network



Badrinarayanan et al. 2016. ArXiv

First test with the built-in network on Matlab (SegNet)



Acknowledgement

Thanks!

Sara Rombouts
Hernan Bonomi
Marcelo Nöllmann
Antoine Le Gall
Baptiste Guilhas
Julian Gurgo
Sergio Espinola
Markus Götz
Andrès Cardozzo
Christophe Houbron
Diego Cattoni
Fanny Berard

Standford Lectures on neural network and deep Learning

https://www.youtube.com/playlist?list=PLC1qU-LWwrF64f4QKQT-Vg5Wr4qEE1Zxk