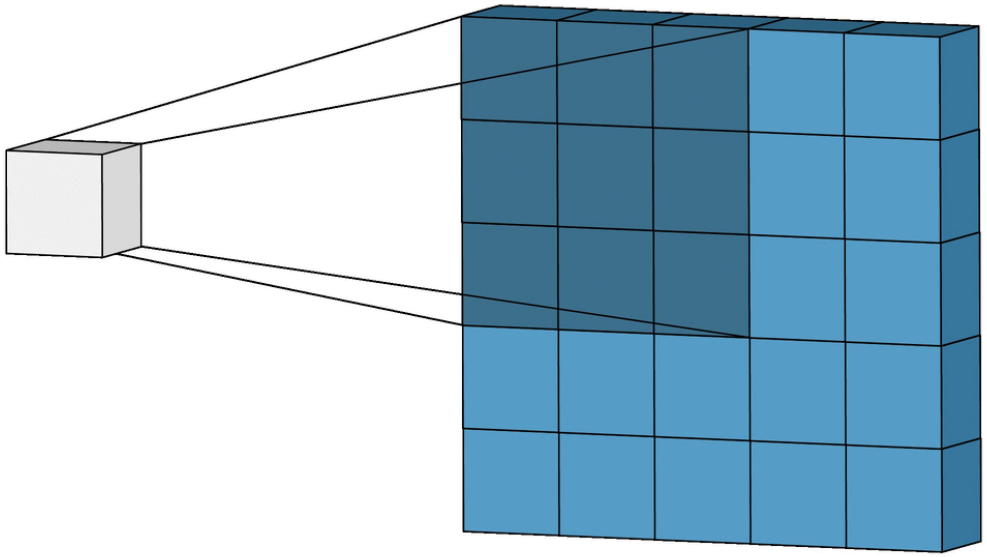
CBG
Max Planck Institute
of Molecular Cell Biology
and Genetics

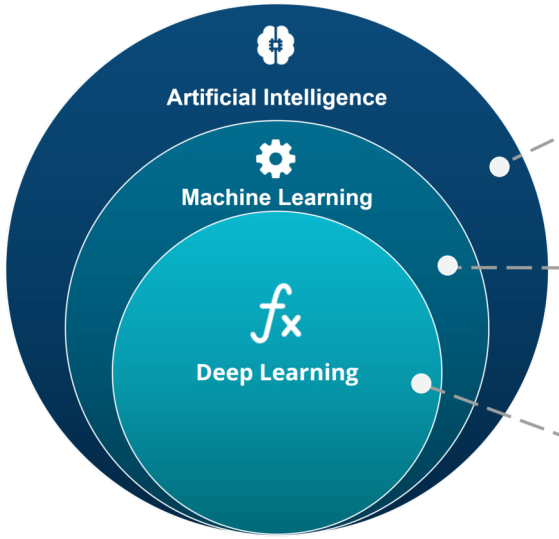
An Introduction to Deep Learning

Manan Lalit

July 5, 2022



Deep Learning Vs AI, Learning Vs Non-Learning

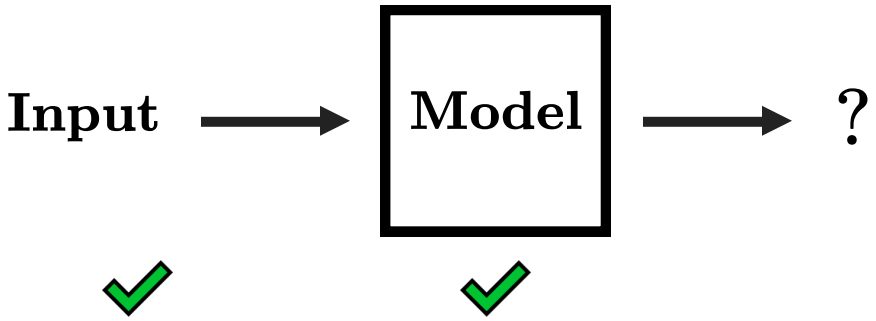


ARTIFICIAL INTELLIGENCE
A technique which enables machines to mimic human behaviour

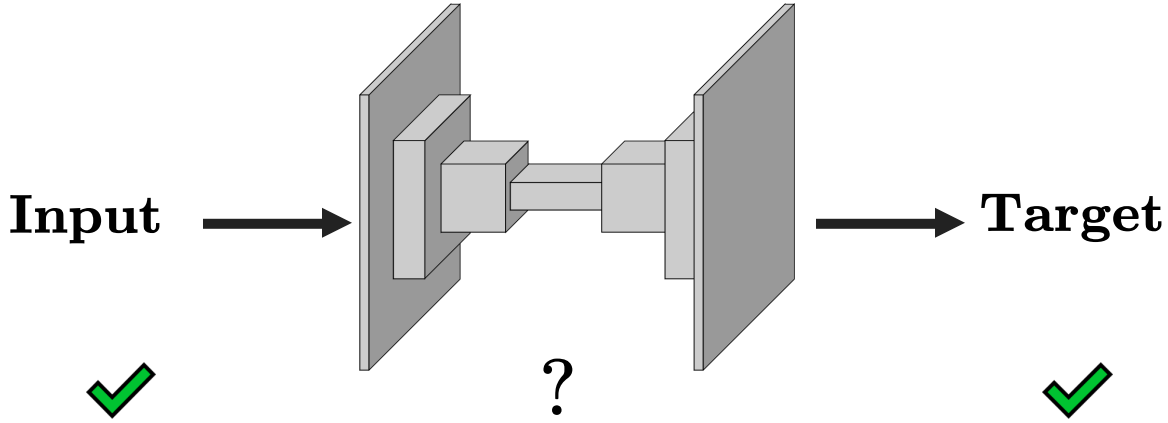
MACHINE LEARNING
Subset of AI technique which use statistical methods to enable machines to improve with experience

DEEP LEARNING
Subset of ML which make the computation of multi-layer neural network feasible

Non-Learning Approaches



(Deep) Learning-based Approaches



Training Phase: *Context*-aware learning of model weights or parameters

History of Deep Learning

Learning representations by back-propagating errors

David E. Rumelhart*, Geoffrey E. Hinton†
& Ronald J. Williams*

* Institute for Cognitive Science, C-015, University of Californ
San Diego, La Jolla, California 92093, USA
† Department of Computer Science, Carnegie-Mellon Universi
Pittsburgh, Philadelphia 15213, USA

(1986) Rumelhart ... Hinton showing back-propagation can be used to train networks

Multilayer Feedforward Networks are Universal Approximators

KURT HORNIK
Technische Universität Wien

MAXWELL STINCHCOMBE AND HALBERT WHITE
University of California, San Diego

(*Received* 16 *September* 1988; *revised and accepted* 9 *March* 1989)

(1989) Hornik et al showed that multi-layer networks are capable of approximating any function

(1993) Yann Le Cunn showing performance of convolutional nets on hand-written digits

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

(2012) AlexNet won the ImageNet challenge

First network to show that deep networks can be trained using GPUs

U-Net: Convolutional Networks for Biomedical Image Segmentation

Olaf Ronneberger, Philipp Fischer, and Thomas Brox

Computer Science Department and BIOSS Centre for Biological Signalling Studies,
University of Freiburg, Germany
ronneber@informatik.uni-freiburg.de,
WWW home page: http://lmb.informatik.uni-freiburg.de/

(2015) Ronneberger et al show skip connections architecture enables better training with fewer examples

Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

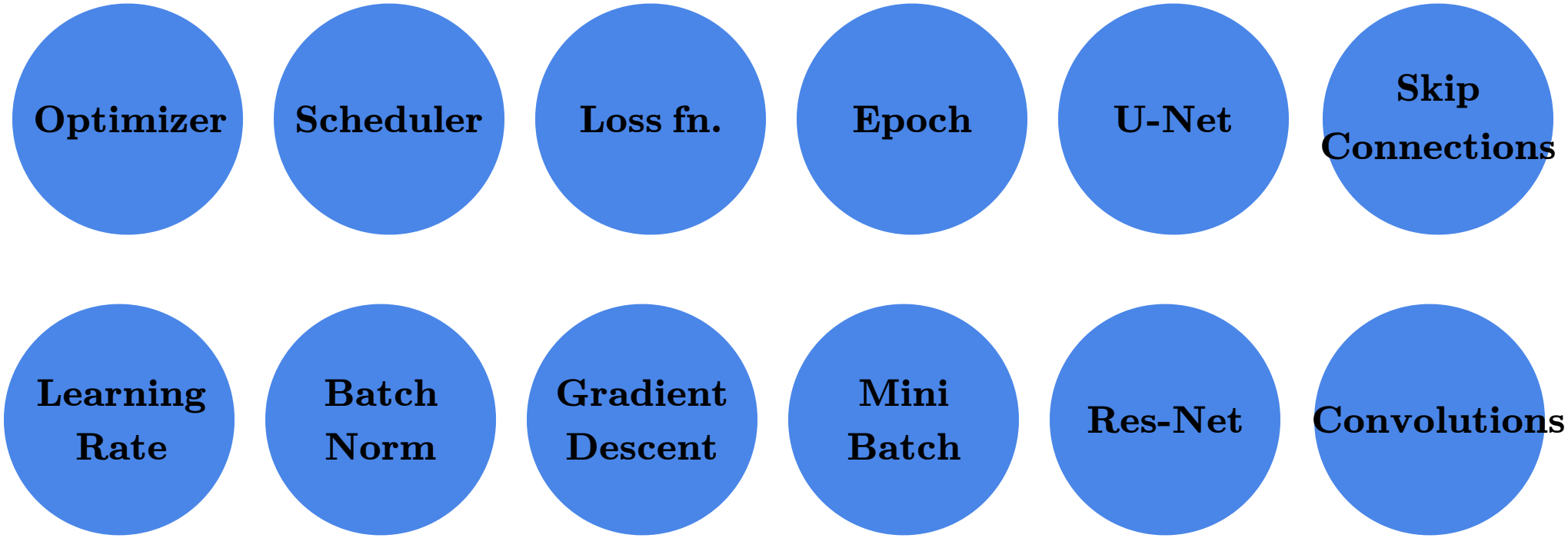
Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

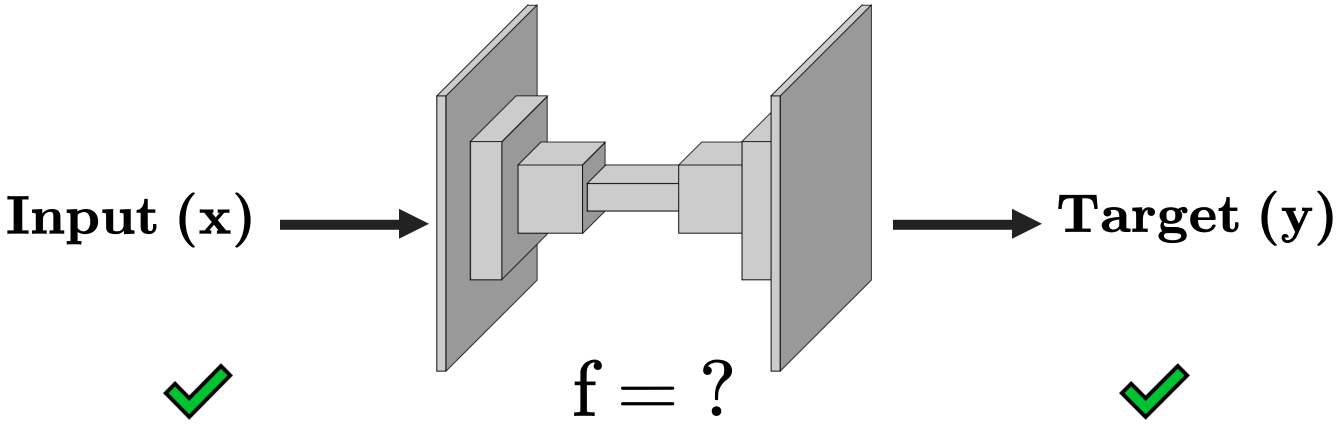
(2017) Vaswani et al propose a new non-convolutional architecture

Vocabulary of Deep Learning

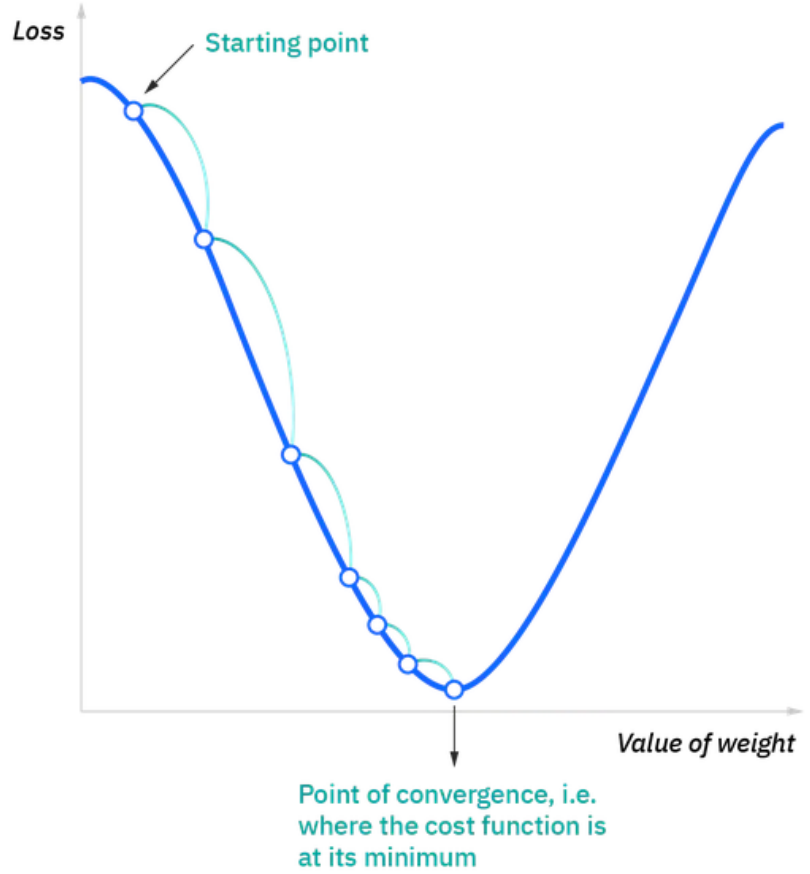


Loss Functions and Gradient Descent

(Deep) Learning-based Approaches



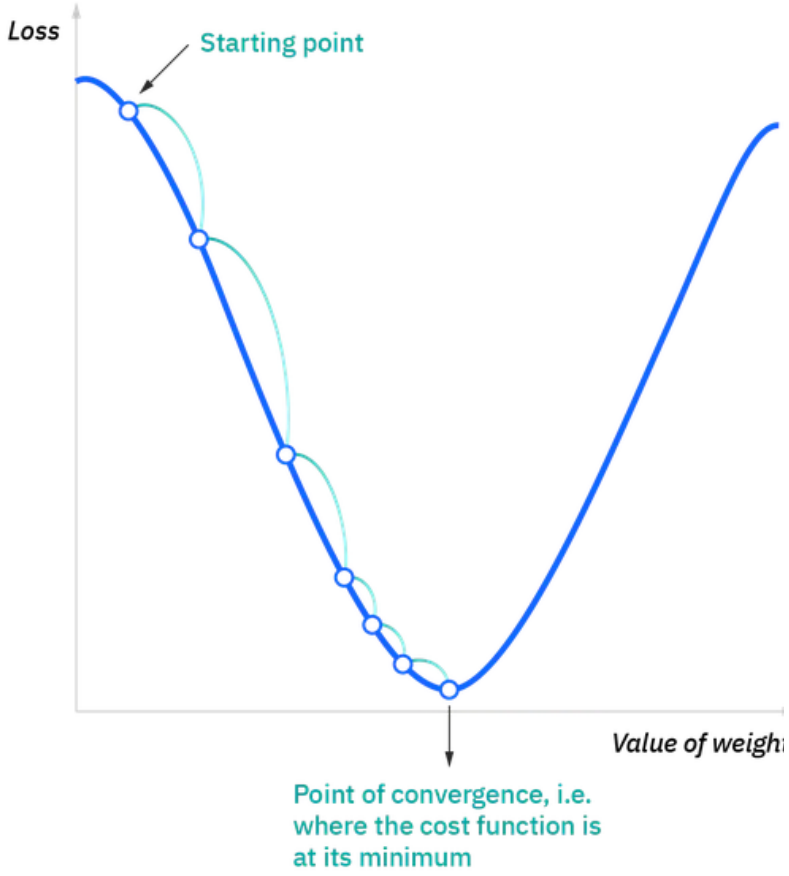
Training Phase: *Context*-aware learning of model weights or parameters

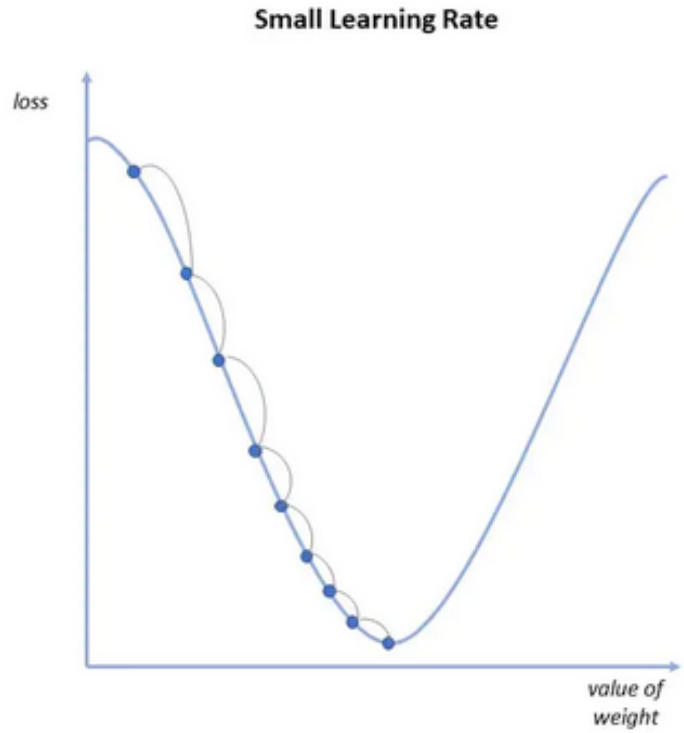


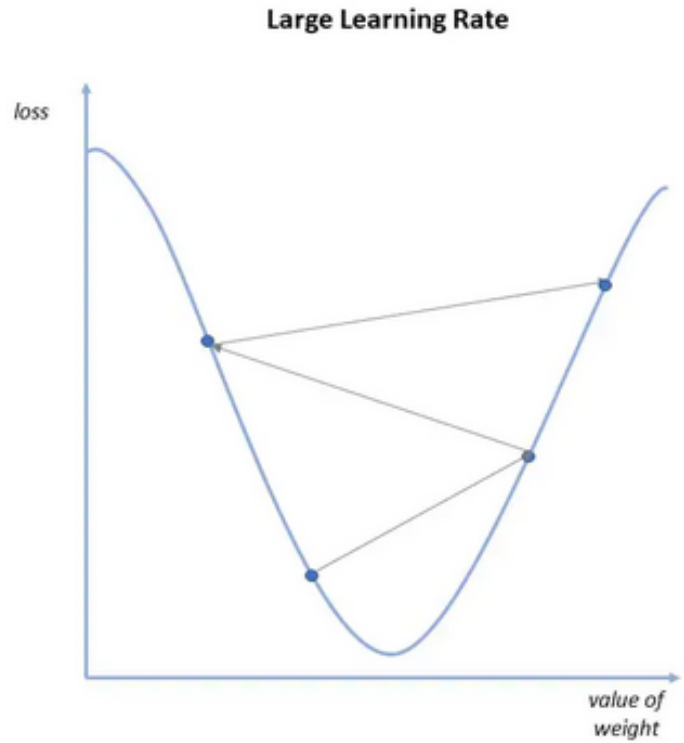
$$L = \frac{1}{M}\Sigma_i^M \| \hat{f}_\theta(x_i), y_i \|_2$$

$$\theta_{\text{next}} = \theta_{\text{prev}} - \alpha \frac{\partial L}{\partial \theta}|_{\theta_{\text{prev}}}$$

Gradient Descent, Learning Rate and Scheduling



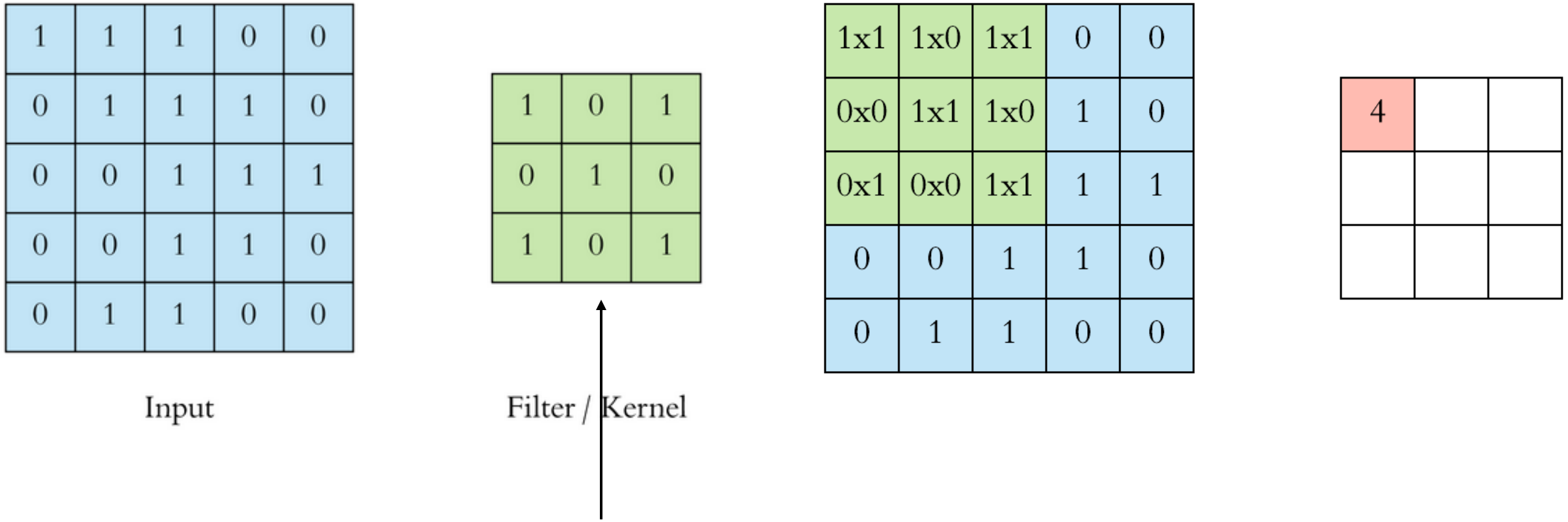




$$\theta_{\text{next}} = \theta_{\text{prev}} - \alpha \frac{\partial L}{\partial \theta}|_{\theta_{\text{prev}}}$$

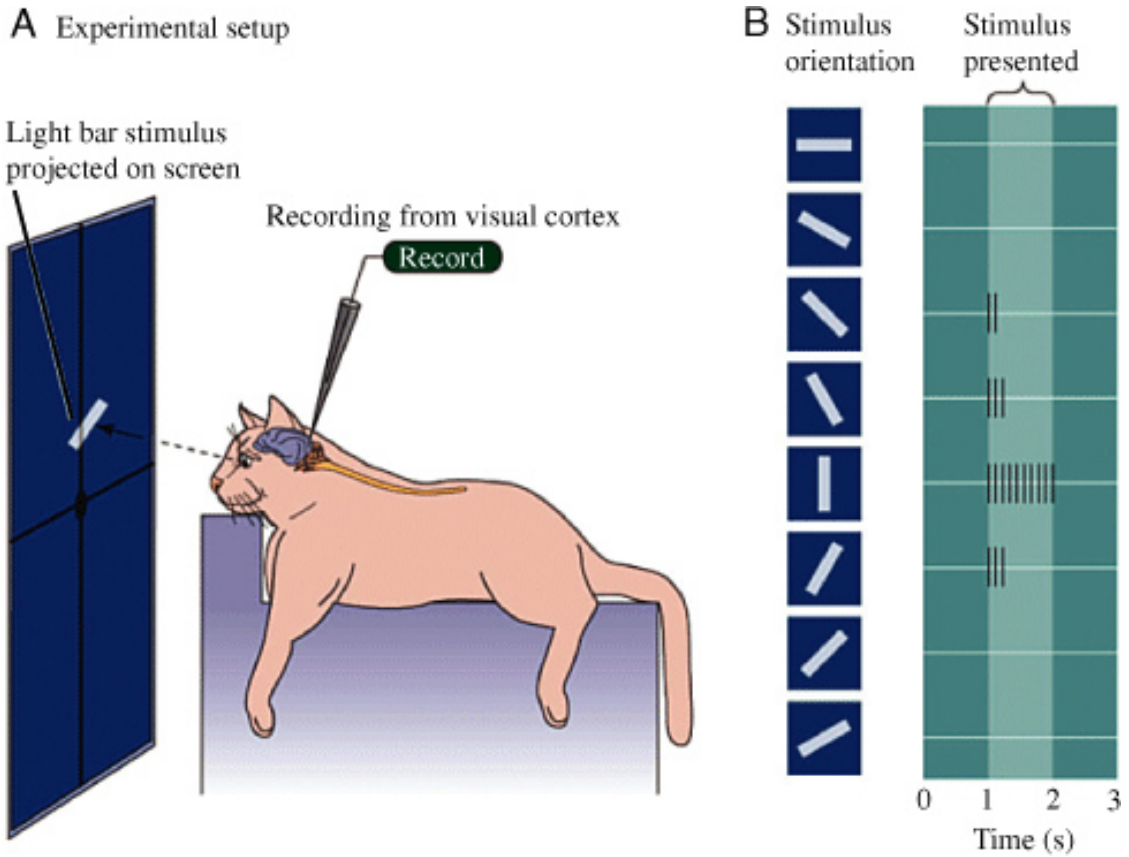
Reduce learning rate / take smaller steps as training progresses

Convolutional Layers enable fewer parameters



These are the weights which are learnt during the training procedure

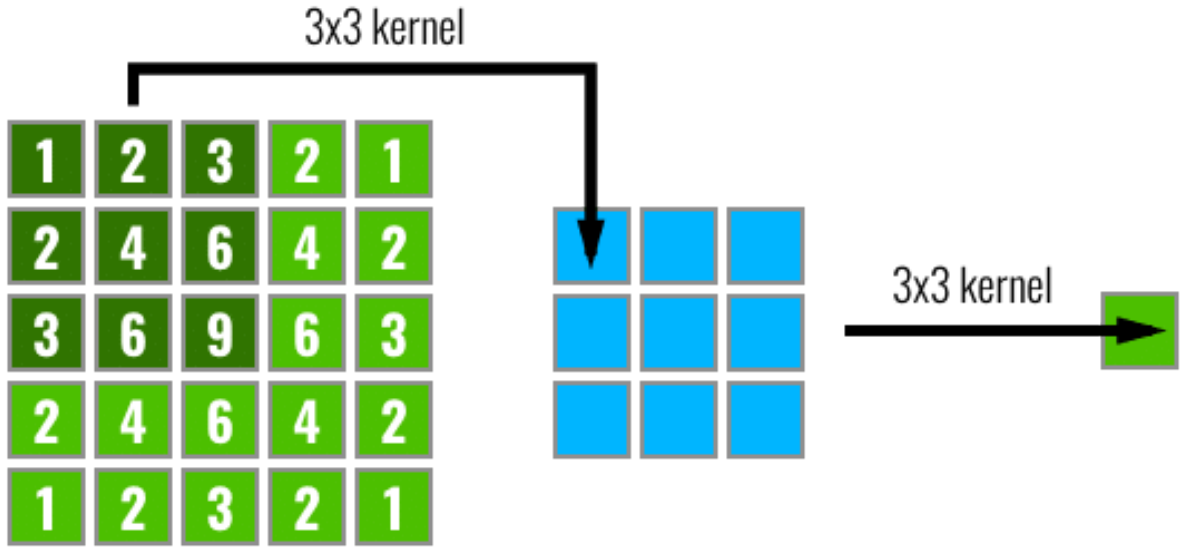
Convolutions and Effective Receptive Fields



Source: https://www.informit.com/articles/article.aspx?p=1431818

Hubel and Weisel's experiments showed that visual cortical neurons were encoding the the properties of objects in the world

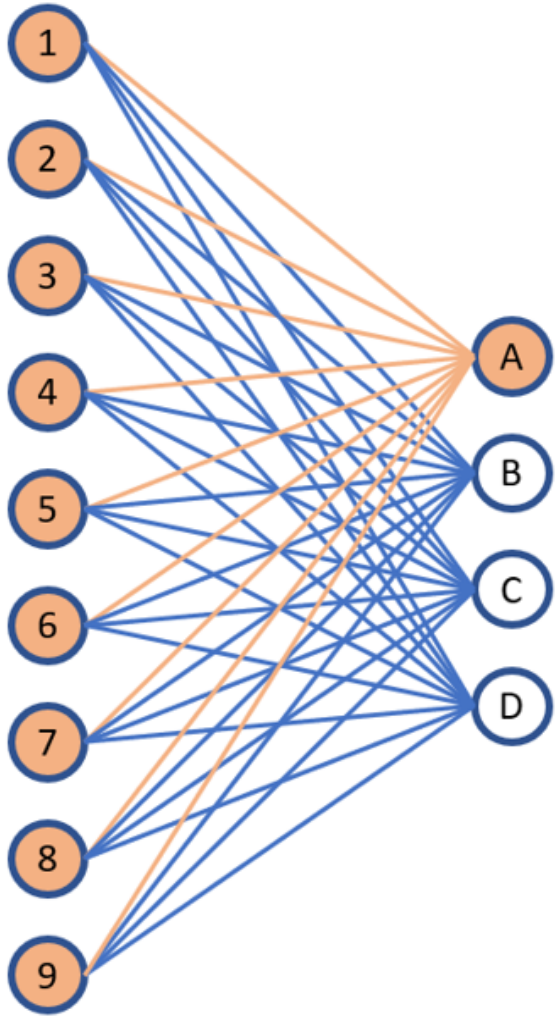
Also, a typical V1 neuron responded to stimulus at specific location (defines the neuron's receptive field)



Source: https://blog.christianperone.com/2017/11/the-effective-receptive-field-on-cnns

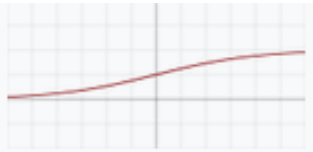
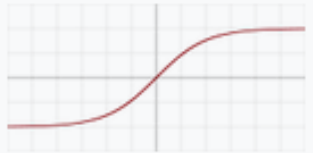
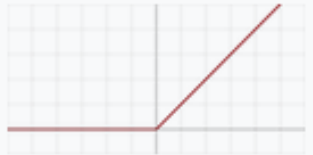
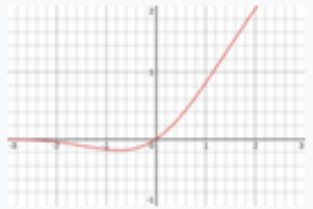
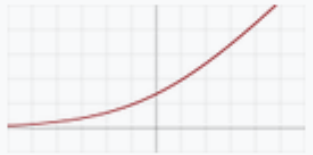
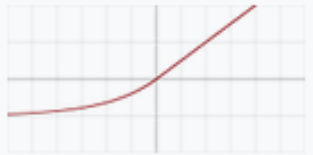
Effective Receptive Field of 5

Fully Connected Layers require more parameters



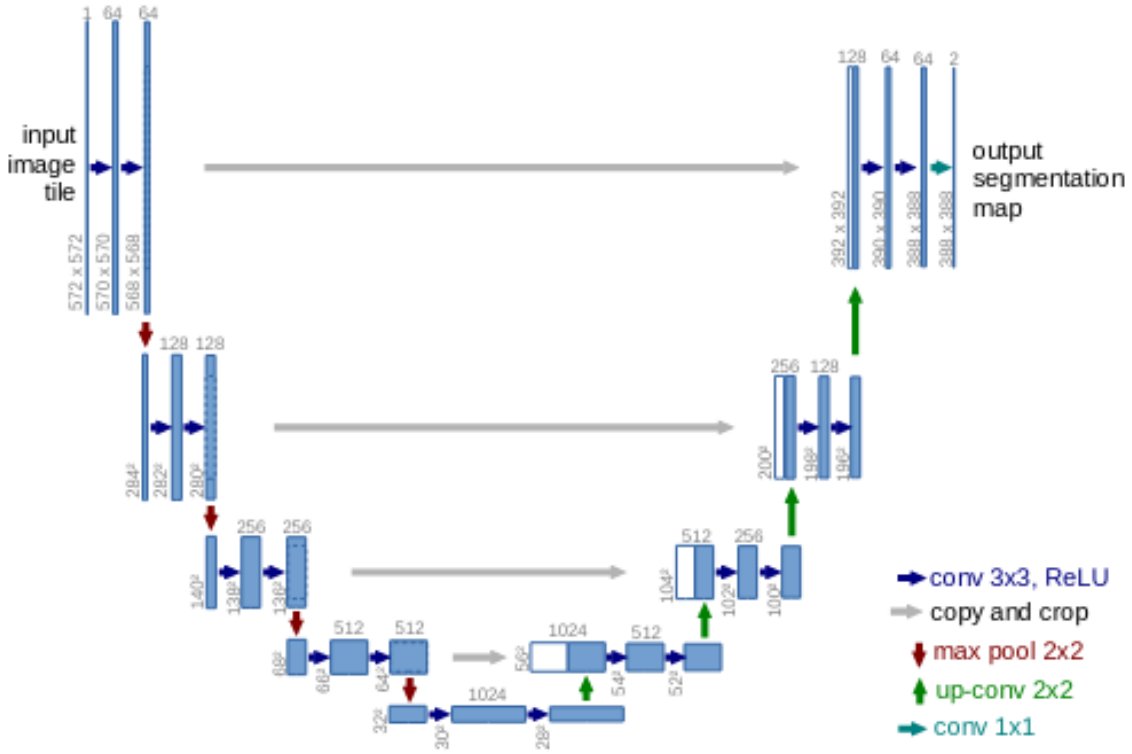
Source: https://towardsdatascience.com/convolutional-layers-vs-fully-connected-layers-364f05ab460b

Between each pair of consecutive layers, Non-Linear Activations are used

Logistic, sigmoid, or soft step		$\sigma(x) = \frac{1}{1+e^{-x}}$
Hyperbolic tangent (tanh)		$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$
Rectified linear unit (ReLU)[8]		$\begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ $= \max\{0, x\} = x\mathbf{1}_{x>0}$
Gaussian Error Linear Unit (GELU)[5]		$\frac{1}{2}x\left(1+\operatorname{erf}\left(\frac{x}{\sqrt{2}}\right)\right)$ $= x\Phi(x)$
Softplus[9]		$\ln(1+e^{x})$
Exponential linear unit (ELU)[10]		$\begin{cases} \alpha\,(e^{x}-1) & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases}$ with parameter α

Source: https://en.wikipedia.org/wiki/Activation_function

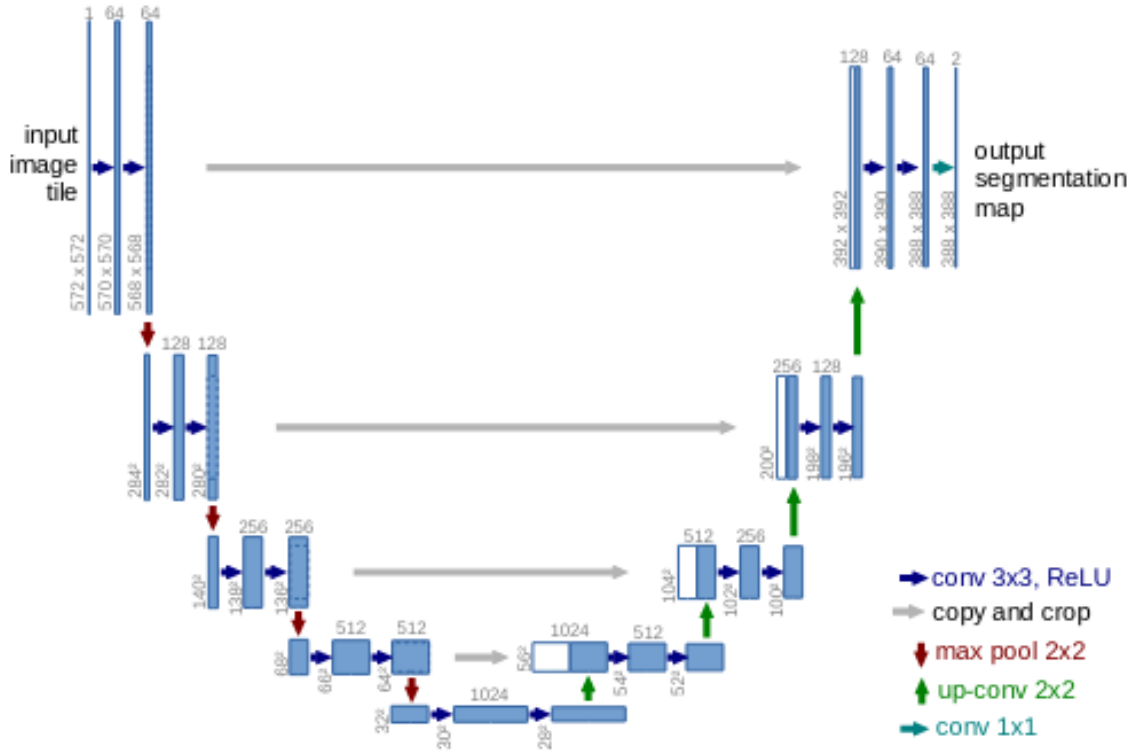
U-Net is a Fully Convolutional Network

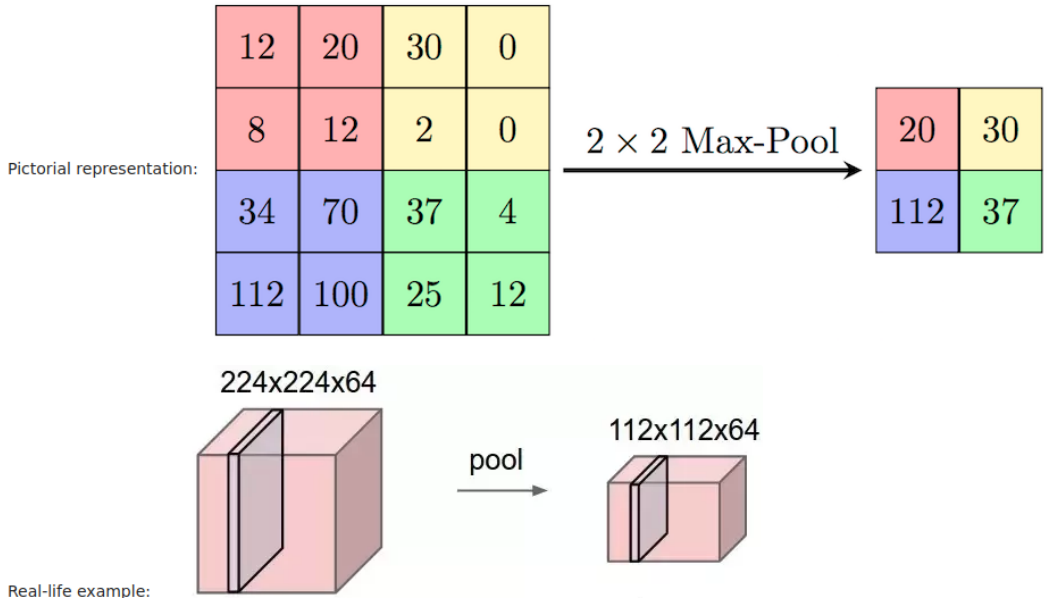


"U-Net: Convolutional Networks for Biomedical Image Segmentation.", Ronneberger, O. et al, 2015

U-Net is a Fully Convolutional Network

It achieves downsampling through Max Pool and upsampling through Up Convolutions

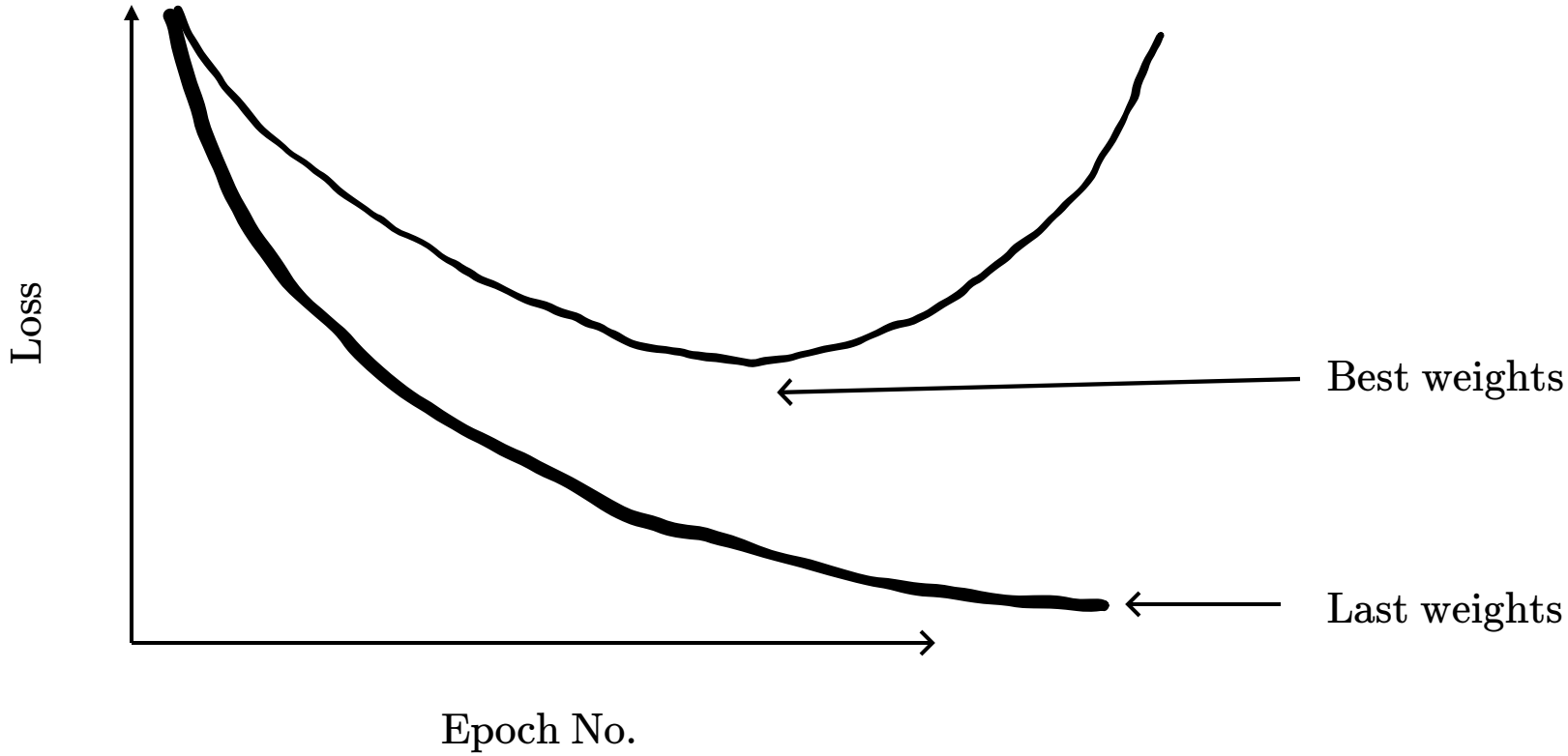




Source: https://computersciencewiki.org/index.php/Max-pooling_/_Pooling

"U-Net: Convolutional Networks for Biomedical Image Segmentation.", Ronneberger, O. et al, 2015

Overfitting and Validation



Training on small dataset can lead to overfitting!

One solution: Use the best performing model state on validation subset instead of last model state

Supervised vs Self-Supervised Training

Supervised Training

Both input (x) and targets (y) are available

Examples include CARE (denoising), StarDist (instance segmentation) etc

Self-Supervised Training

Only input (x) is available. Target (y) is constructed through a *proxy* task

Examples in NLP include given a sentence, predict the next word (y) in the sentence given all the preceding words (x)

Examples in CV include given an image, predict a masked pixel given all the neighboring pixels

Typical Lines of Code in DL training pipelines

```
# set device
device = torch.device("cuda:0" if args['cuda'] else "cpu")
```

```
# train dataloader
train_dataset = get_dataset(
    args['train_dataset']['name'], args['train_dataset']['kwargs'])

train_dataset_it = torch.utils.data.DataLoader(
    train_dataset, batch_size=args['train_dataset']['batch_size'], shuffle=True, drop_last=True,
    num_workers=args['train_dataset']['workers'], pin_memory=True if args['cuda'] else False)
```

```
# val dataloader
val_dataset = get_dataset(
    args['val_dataset']['name'], args['val_dataset']['kwargs'])

val_dataset_it = torch.utils.data.DataLoader(
    val_dataset, batch_size=args['val_dataset']['batch_size'], shuffle=True, drop_last=True,
    num_workers=args['train_dataset']['workers'], pin_memory=True if args['cuda'] else False)
```

```
# set model
model = get_model(args['model']['name'], args['model']['kwargs'])
model = torch.nn.DataParallel(model).to(device)
```

```
# set optimizer
optimizer = torch.optim.Adam(
    model.parameters(), lr=args['lr'], weight_decay=1e-4)
```

Typical Lines of Code in DL training pipelines

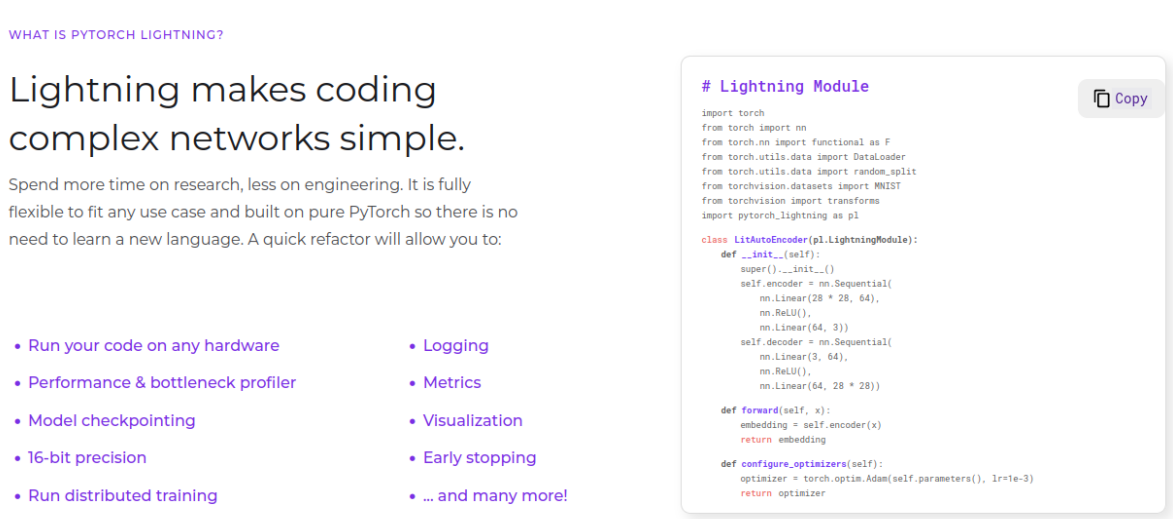
```
# set scheduler
scheduler = torch.optim.lr_scheduler.LambdaLR(
    optimizer, lr_lambda=lambda_, )
```

```
# create criterion
criterion = nn.CrossEntropyLoss(weight=torch.tensor([1.0, 1.0, 10.0]).cuda())
criterion = torch.nn.DataParallel(criterion).to(device)
```

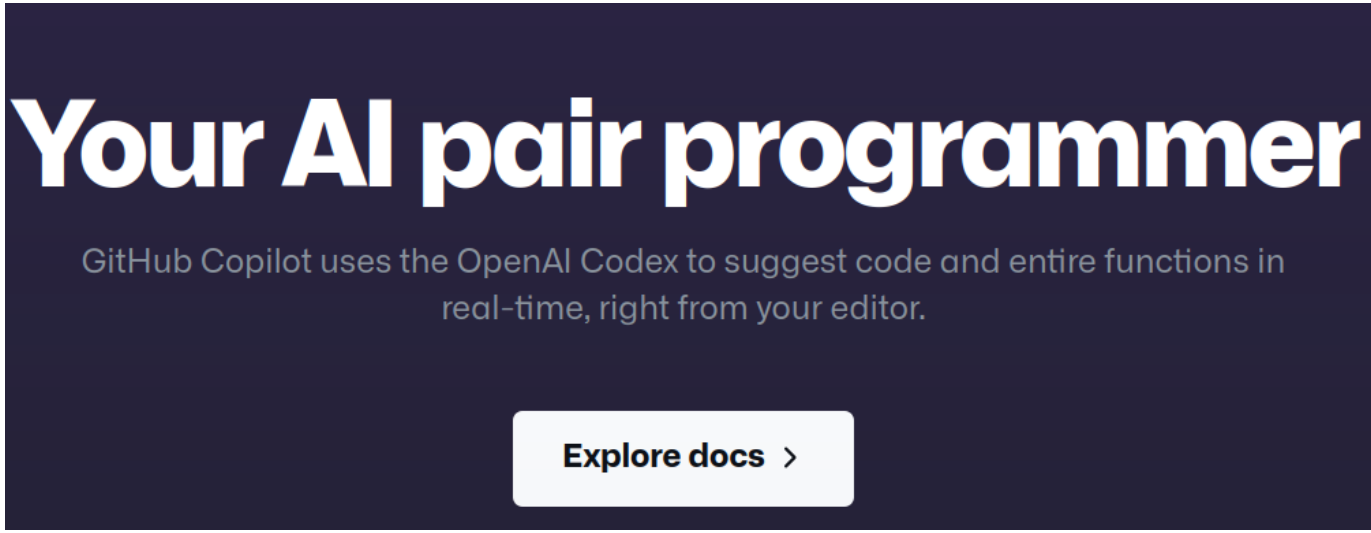
```
def train(epoch):
    model.train()
    for i, sample in enumerate(tqdm(train_dataset_it)):
        images = sample['image']
        semantic_masks = sample['semantic_mask']
        semantic_masks.squeeze_(1)
        instance = sample['instance_mask']
        output = model(images)
        loss = criterion(output, semantic_masks.long())
        loss = loss.mean()
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
        loss_meter.update(loss.item())

    return loss_meter.avg
```

Tools can hide away or suggest boilerplate code



Pytorch Lightning

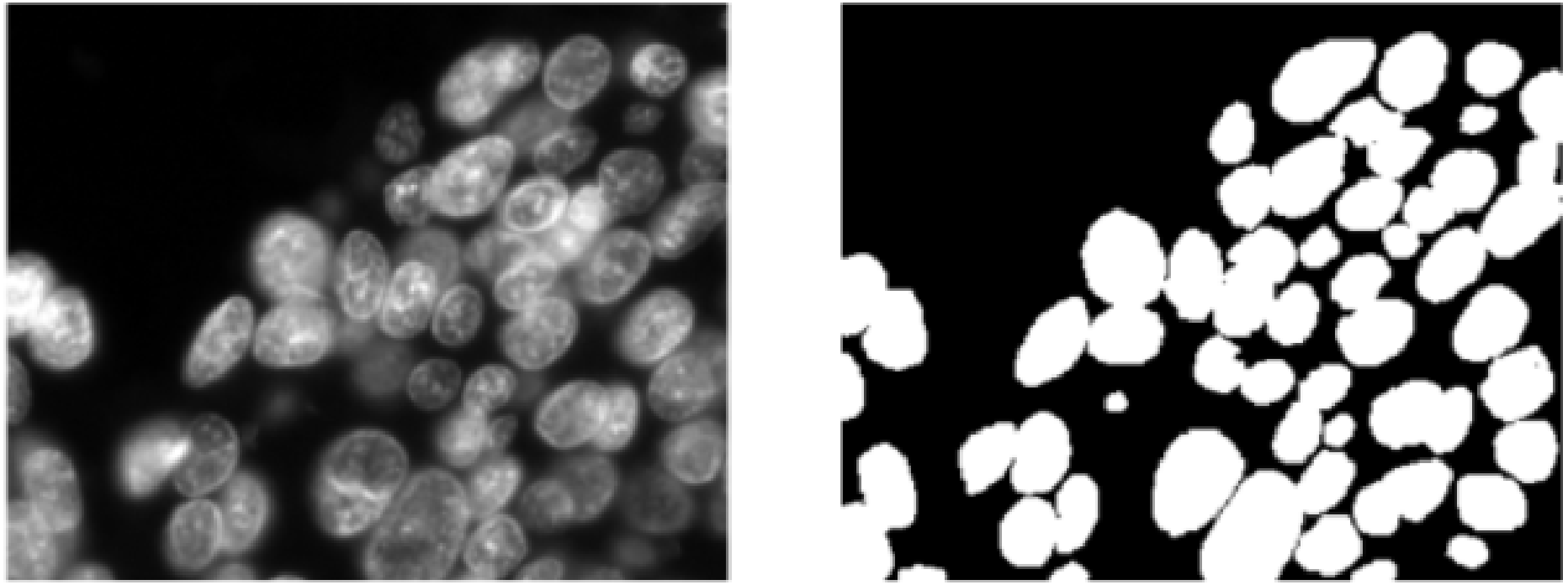


Github Copilot

Deep Learning in Bio-Image Analysis

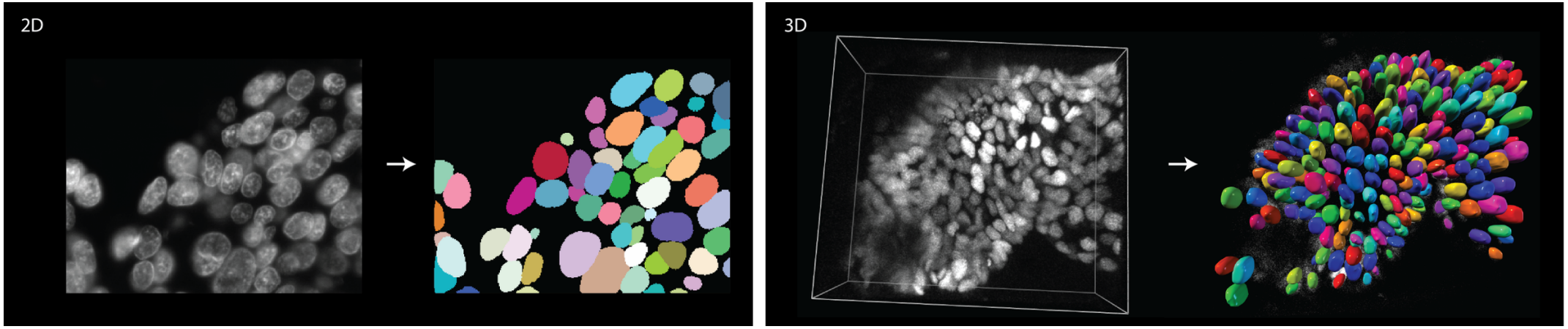
Semantic Segmentation

Input Image → Semantic Segmentation (foreground/background)



Deep Learning in Bio-Image Analysis

Instance Segmentation



"Cell Detection with Star-convex Polygons.", Schmidt, U. and Weigert, M. et al, 2018

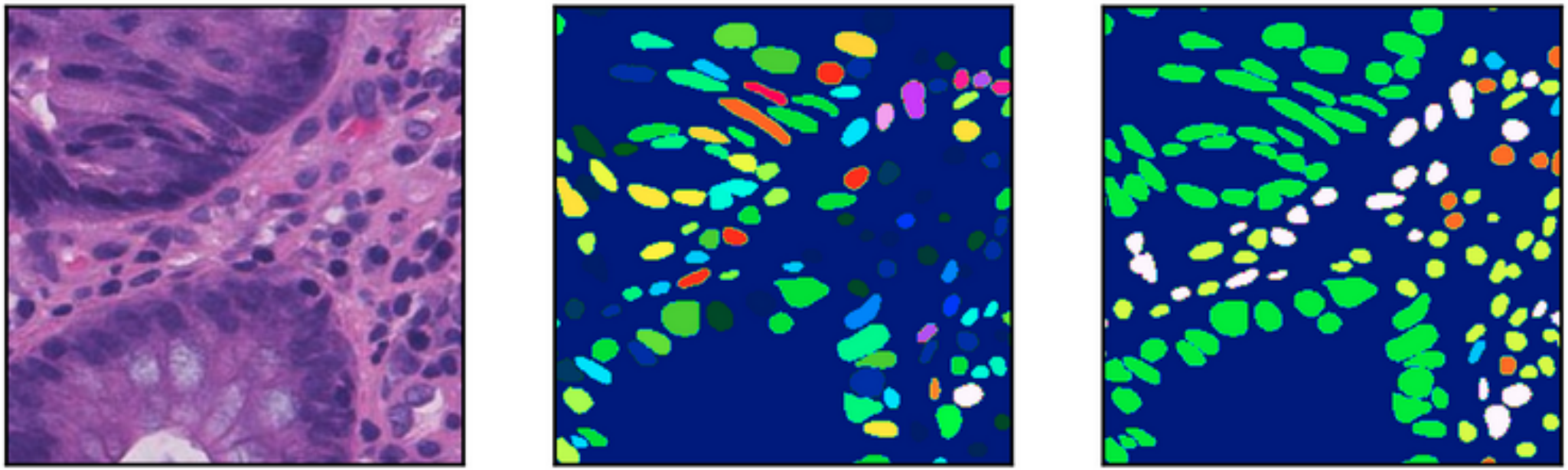
"Star-convex Polyhedra for 3D Object Detection and Segmentation in Microscopy.", Weigert, M. and Schmidt, U. et al, 2020

"Cellpose: a generalist algorithm for cellular segmentation.", Stringer, C. et al, 2021

"Accurate and versatile 3D segmentation of plant tissues at cellular resolution.", Wolny, A. et al, 2020

Deep Learning in Bio-Image Analysis

Joint Instance Segmentation & Classification

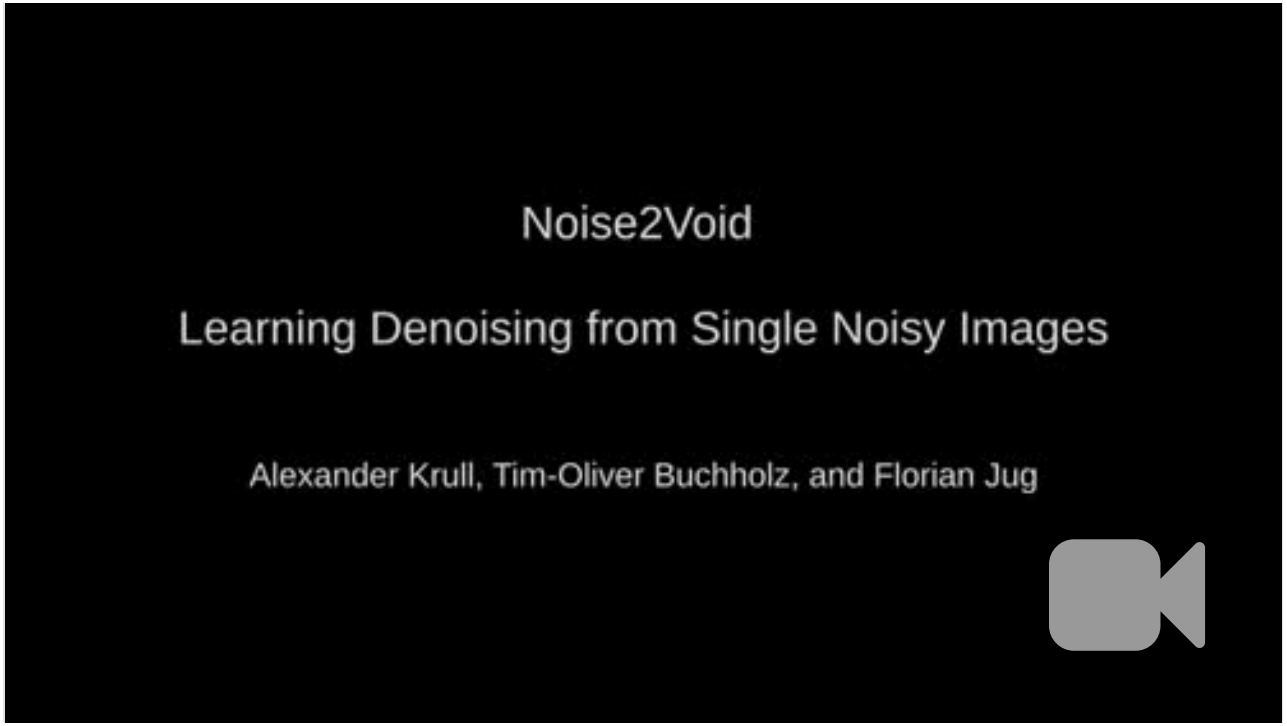


"Nuclei Instance Segmentation & Classification in Histopathology Images with StarDist", Weigert, M. and Schmidt, U., 2022

"CoNIC: Colon Nuclei Identification and Counting Challenge 2022.", Graham, S. et al.

Deep Learning in Bio-Image Analysis

Image Denoising and Restoration

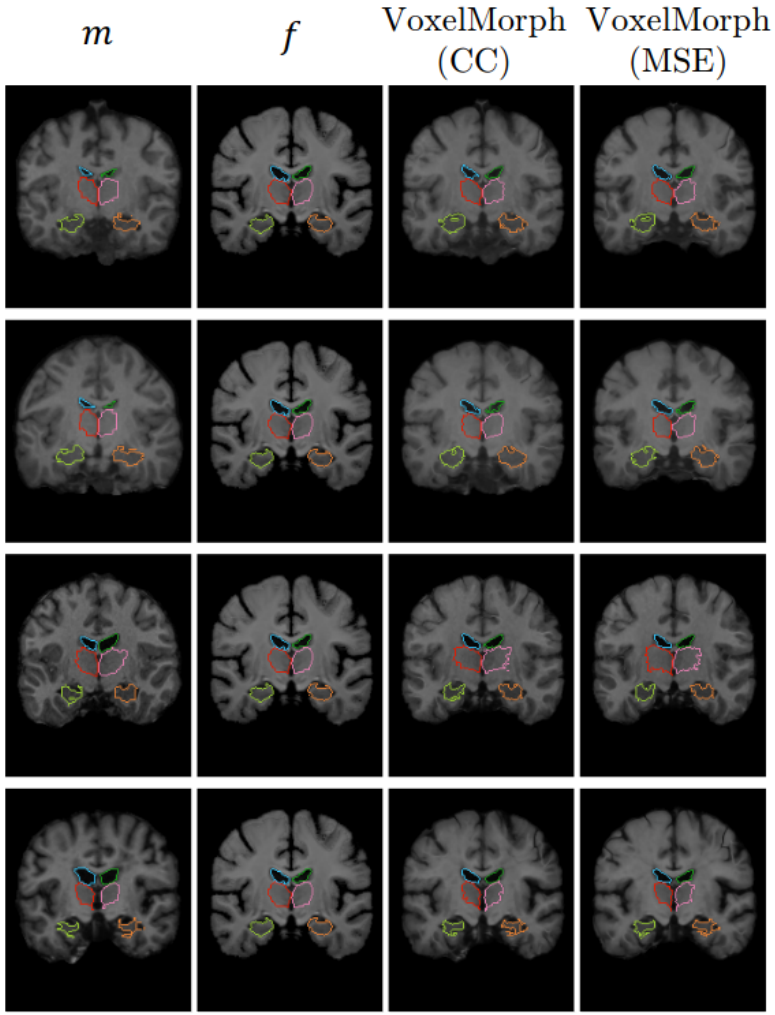


“Noise2Void - Learning Denoising from Single Noisy Images”, Krull, A. and Buchholz, T-O et al, 2018

“Content-aware image restoration: pushing the limits of fluorescence microscopy”, Weigert, M. et al, 2018

Deep Learning in Bio-Image Analysis

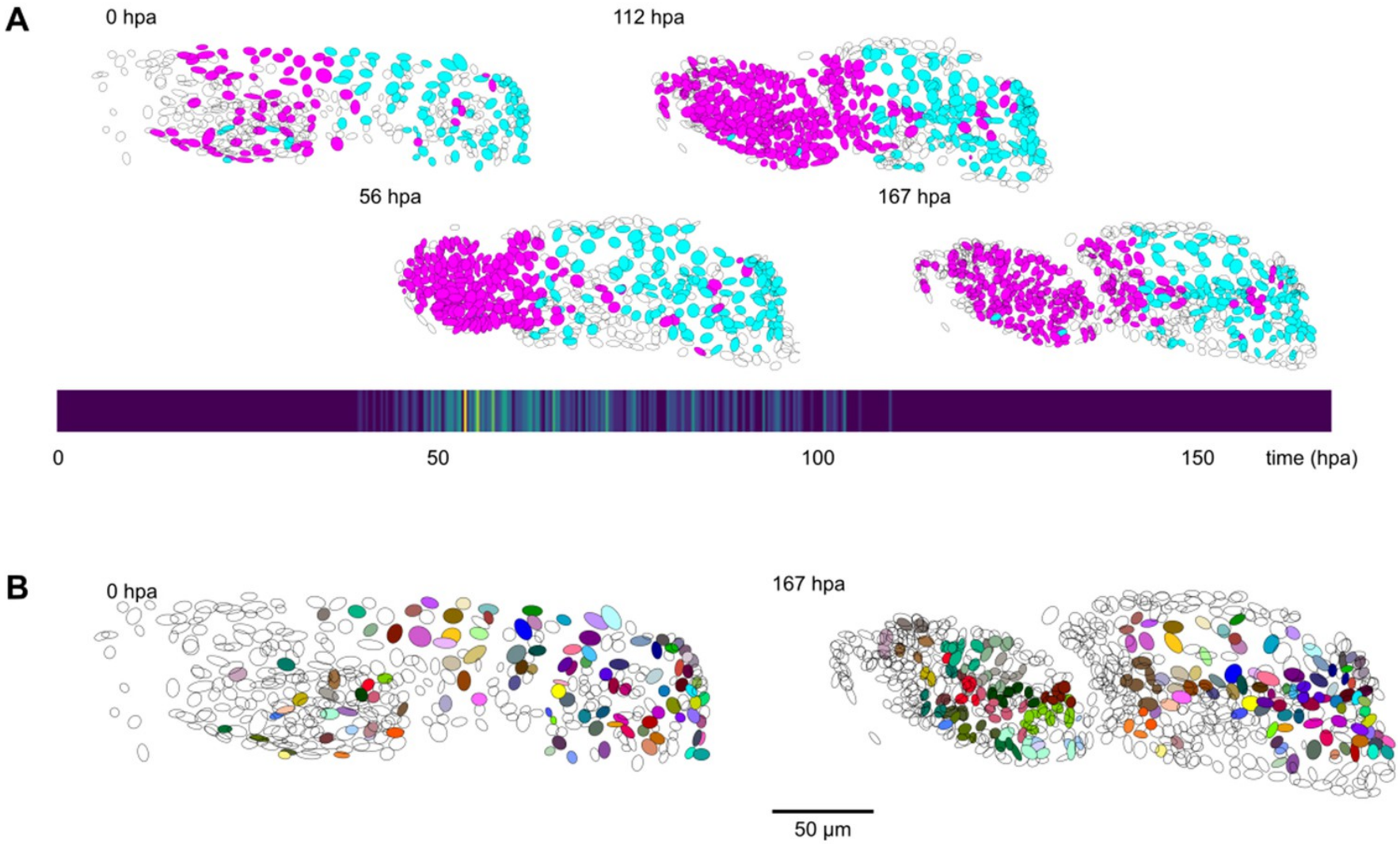
Image Registration



"VoxelMorph: A Learning Framework for Deformable Medical Image Registration", Balakrishnan, G. et al, 2018

Deep Learning in Bio-Image Analysis

Tracking in Time Lapse Movies



"Tracking cell lineages in 3D by incremental deep learning", Sugawara, K. et al, 2022

Exercises for Today

1 - Image Denoising using Noise2Void

2 - Image Semantic Segmentation using Standard U-Nets