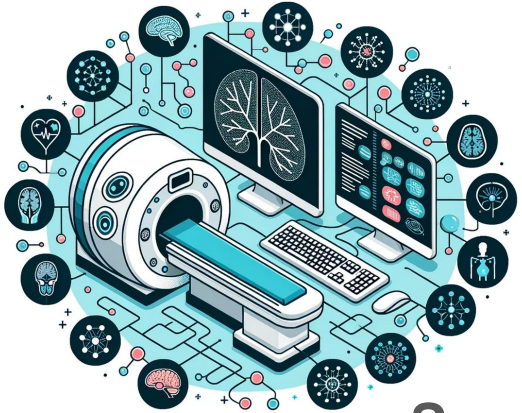
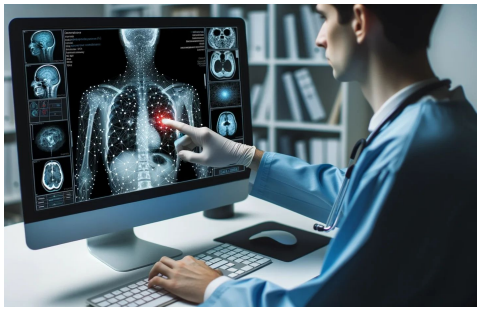
Applied Deep Learning and Generative Models in Healthcare





Session 2: Medical Image Segmentation
Date: Jan 18 2025

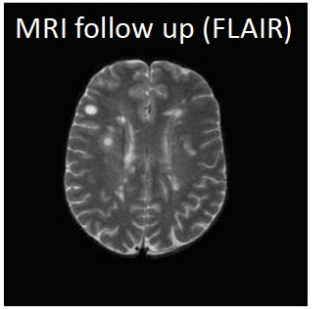
Instructor: Mahmoud E. Khani, Ph.D.

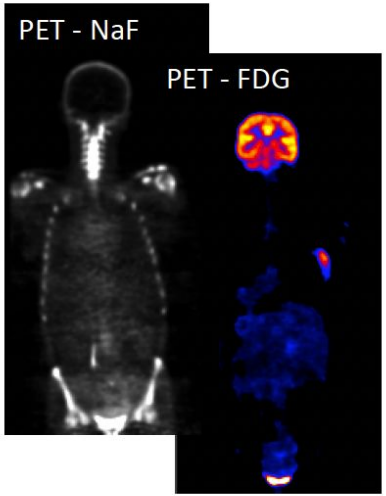
Motivation

- Images represent 90% of medical data

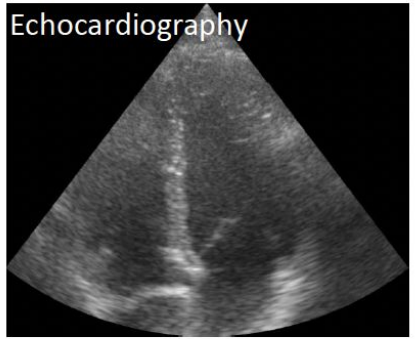
Medical images, many modalities

- 3D/2D (+t), different physics

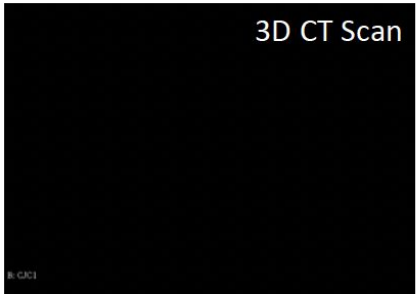




Positron Emission Tomography

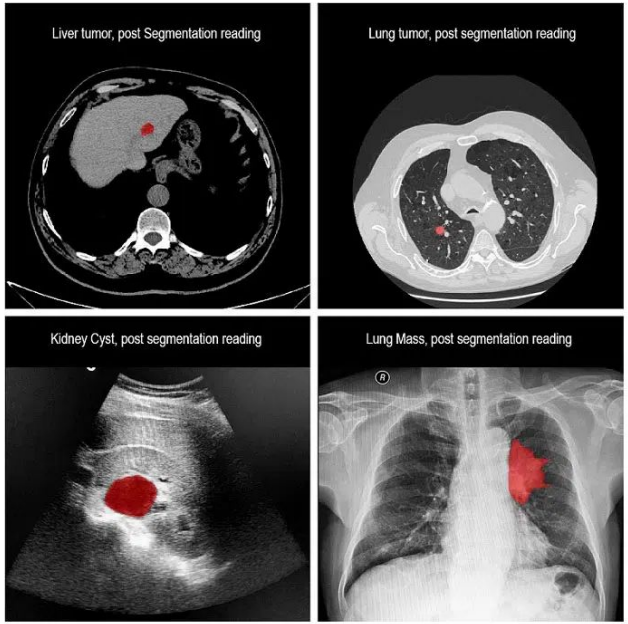


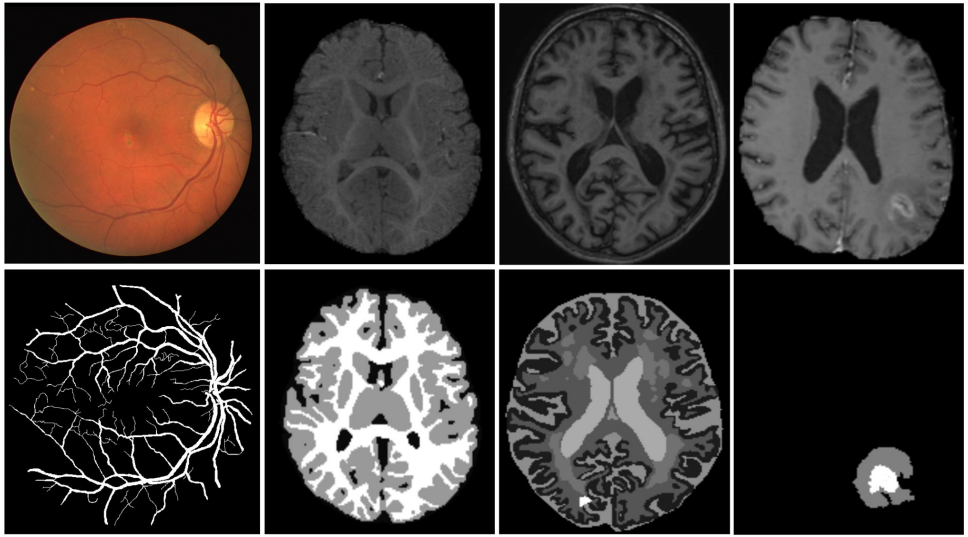
Fundus examination



Medical Image Segmentation - Significance

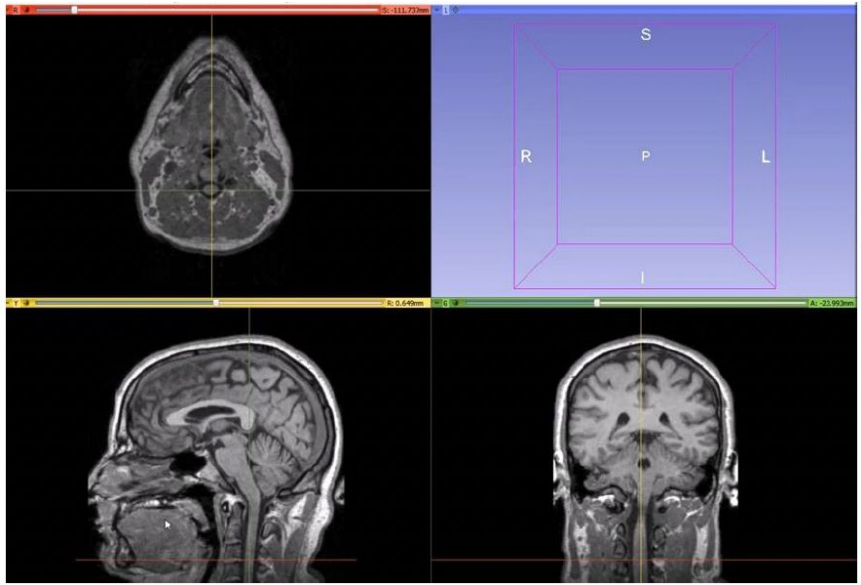
- Most publications related to applications of deep learning and computer vision in medical imaging





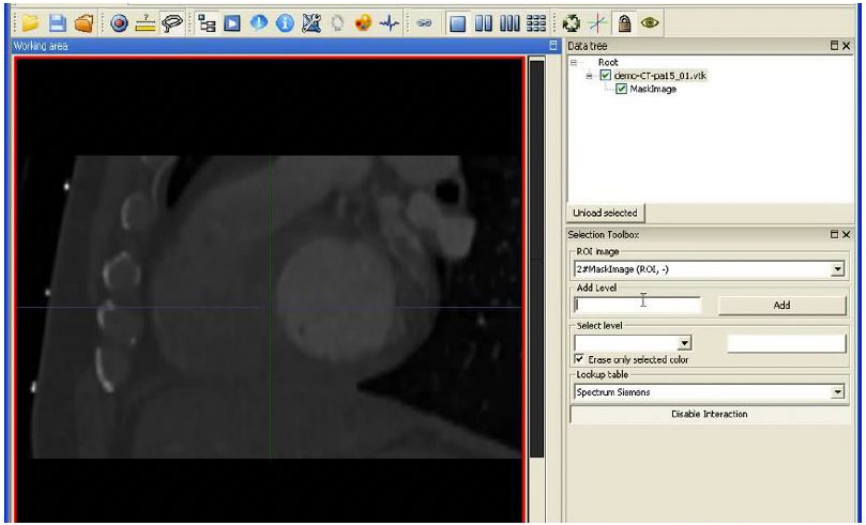
Manual image segmentation is time consuming!

3D Slicer - Quick Manual Segmentation with Interpolation



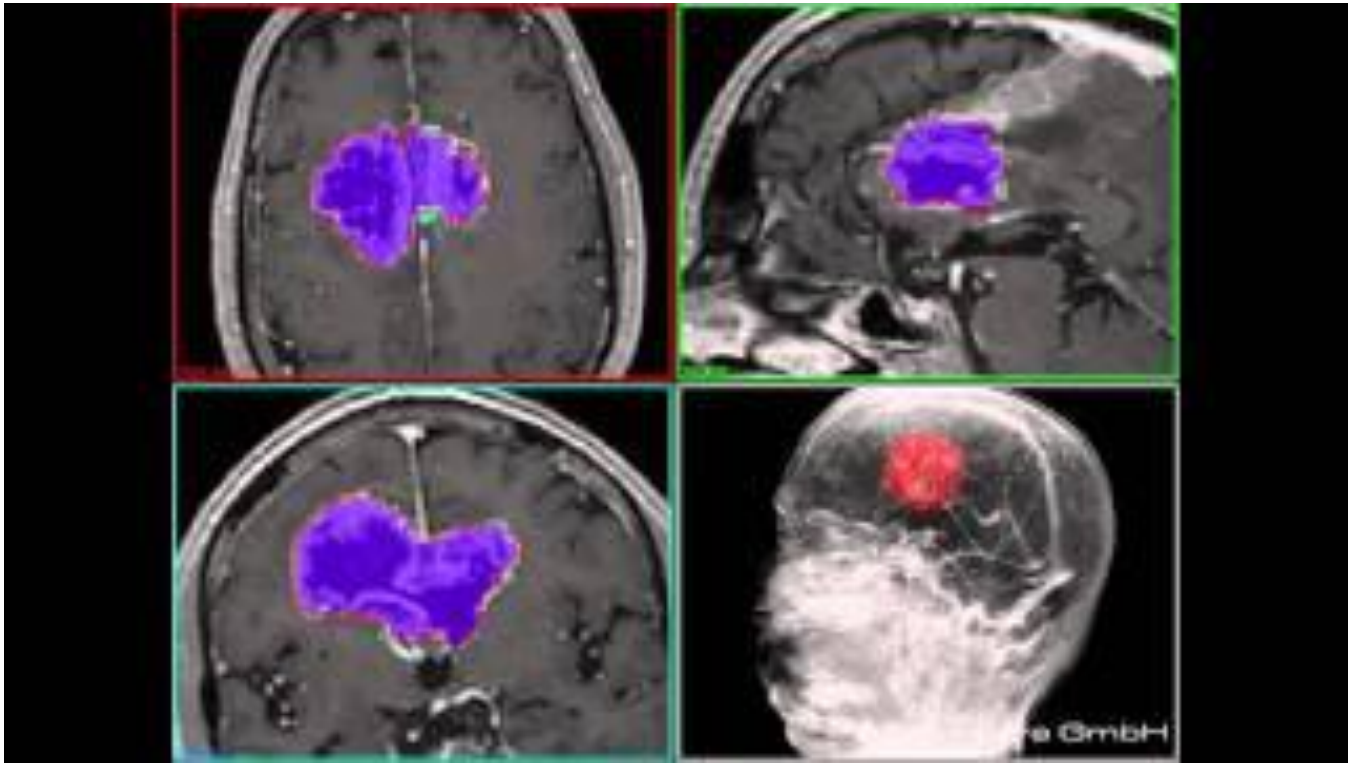
https://www.youtube.com/watch?v=u93kI1MG6Ic&ab_channel=PerkLabResearch

GIMIAS Manual Segmentation



https://www.youtube.com/watch?v=rAHA1OZC8h8&ab_channel=GIMIAS

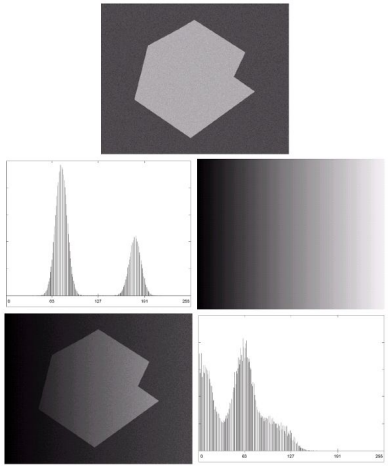
Semi-automated medical image segmentation

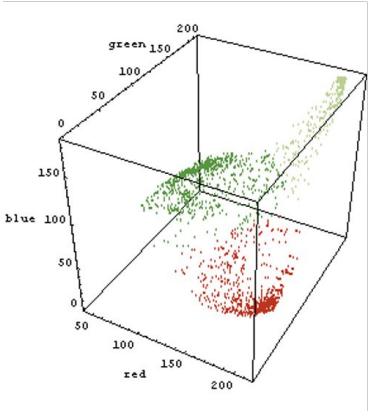


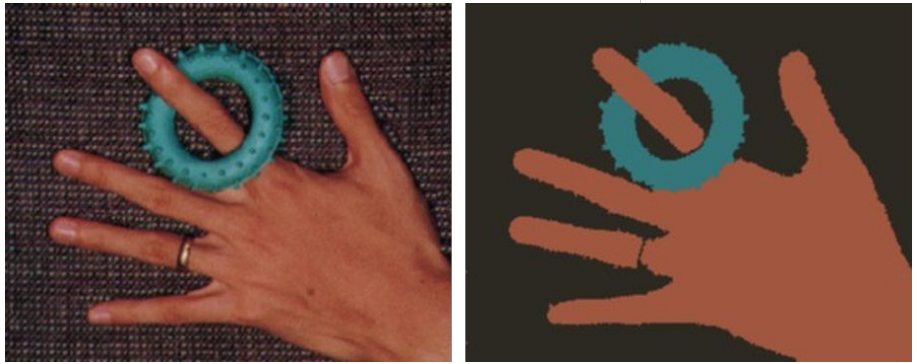
https://www.youtube.com/watch?v=7wCC2NaVLjs&ab_channel=ChimaeraGmbH

Classical image segmentation algorithms

- Thresholding
- Region-oriented segmentation
- Watershed segmentation
- K-means clustering







U-Net: Convolutional Networks for Biomedical Image Segmentation

Olaf Ronneberger, Philipp Fischer, and Thomas Brox

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

Abstract. There is large consent that successful training of deep networks requires many thousand annotated training samples. In this paper, we present a network and training strategy that relies on the strong use of data augmentation to use the available annotated samples more efficiently. The architecture consists of a contracting path to capture context and a symmetric expanding path that enables precise localization. We show that such a network can be trained end-to-end from very few images and outperforms the prior best method (a sliding-window convolutional network) on the ISBI challenge for segmentation of neuronal structures in electron microscopic stacks. Using the same network trained on transmitted light microscopy images (phase contrast and DIC) we won the ISBI cell tracking challenge 2015 in these categories by a large margin. Moreover, the network is fast. Segmentation of a 512x512 image takes less than a second on a recent GPU. The full implementation (based on Caffe) and the trained networks are available at http://lmb.informatik.uni-freiburg.de/people/ronneber/u-net.

IEEE International Symposium on Biomedical Imaging (ISBI). It focuses on benchmarking image segmentation methods applied to biomedical and medical imaging tasks. The challenge typically involves segmenting anatomical structures, pathological regions, or cellular/molecular components in medical images, such as MRI, CT, microscopy, or histopathology scans.

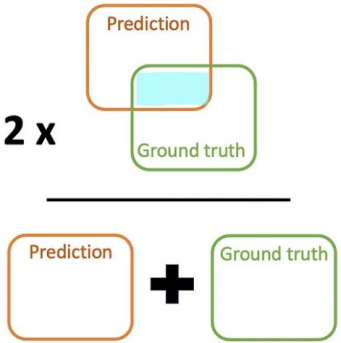
Medical image segmentation challenges

- Tumor segmentation
- Retinal vessel segmentation
- Organ delineation
- Cell tracking
- Disease detection.

Evaluation metrics:

- Dice similarity coefficient (DSC)
- intersection-over-union (IoU)

$$\text{Dice} = \frac{2\ X\ Area\ of\ overlap}{Total\ area} = \frac{2 \times}{}$$




```
def dice_coef(groundtruth_mask, pred_mask):
    intersect = np.sum(pred_mask*groundtruth_mask)
    total_sum = np.sum(pred_mask) + np.sum(groundtruth_mask)
    dice = np.mean(2*intersect/total_sum)
    return round(dice, 3) #round up to 3 decimal places
```

https://medium.com/@nghihuynh_37300/understanding-evaluation-metrics-in-medical-image-segmentation-d289a373a3f

Segmentation success

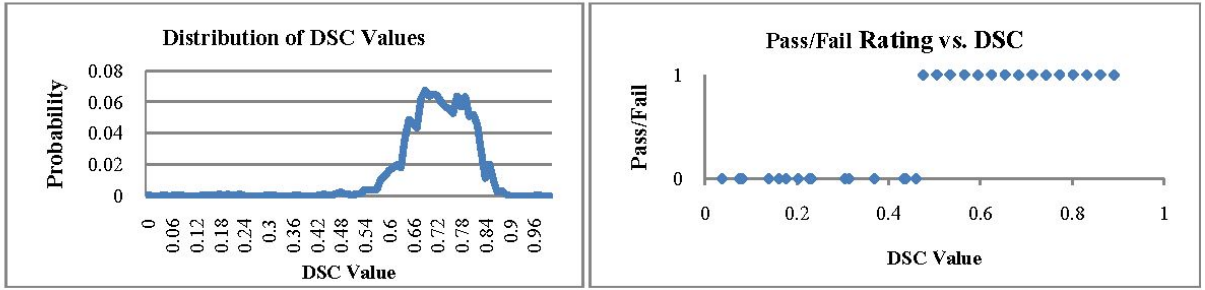


Fig. 1. (a) Distribution of DSC values from the 2,500 segmentations. (b) "Pass/fail" ratings assigned by the radiologists for the 30 randomly selected images used in the study. A value of one indicates a success and a value of zero indicates a failure.

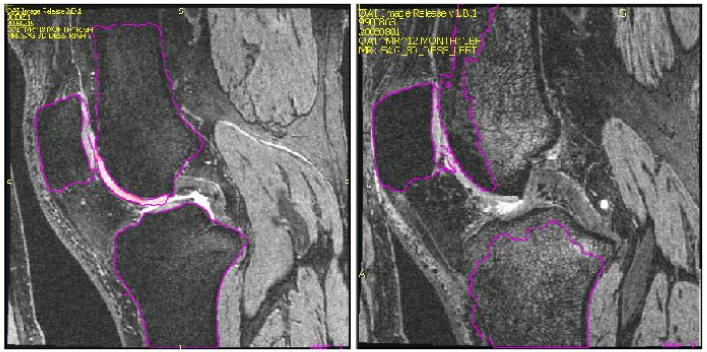
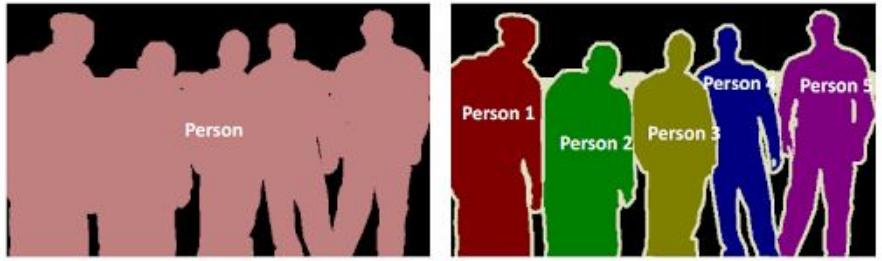


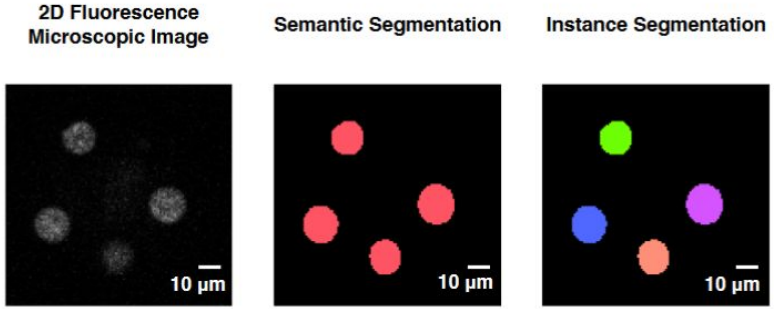
Fig. 2. (left) A typical segmentation labeled as passing by the expert radiologists. (right) A typical segmentation labeled as failing by the expert radiologists.

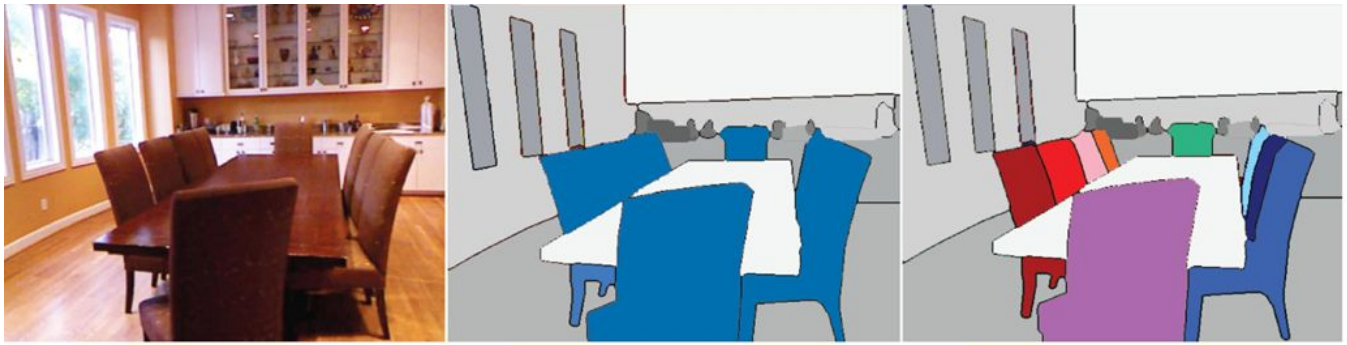
https://api.semanticscholar.org/CorpusID:33232467

Semantic vs Instance Segmentation



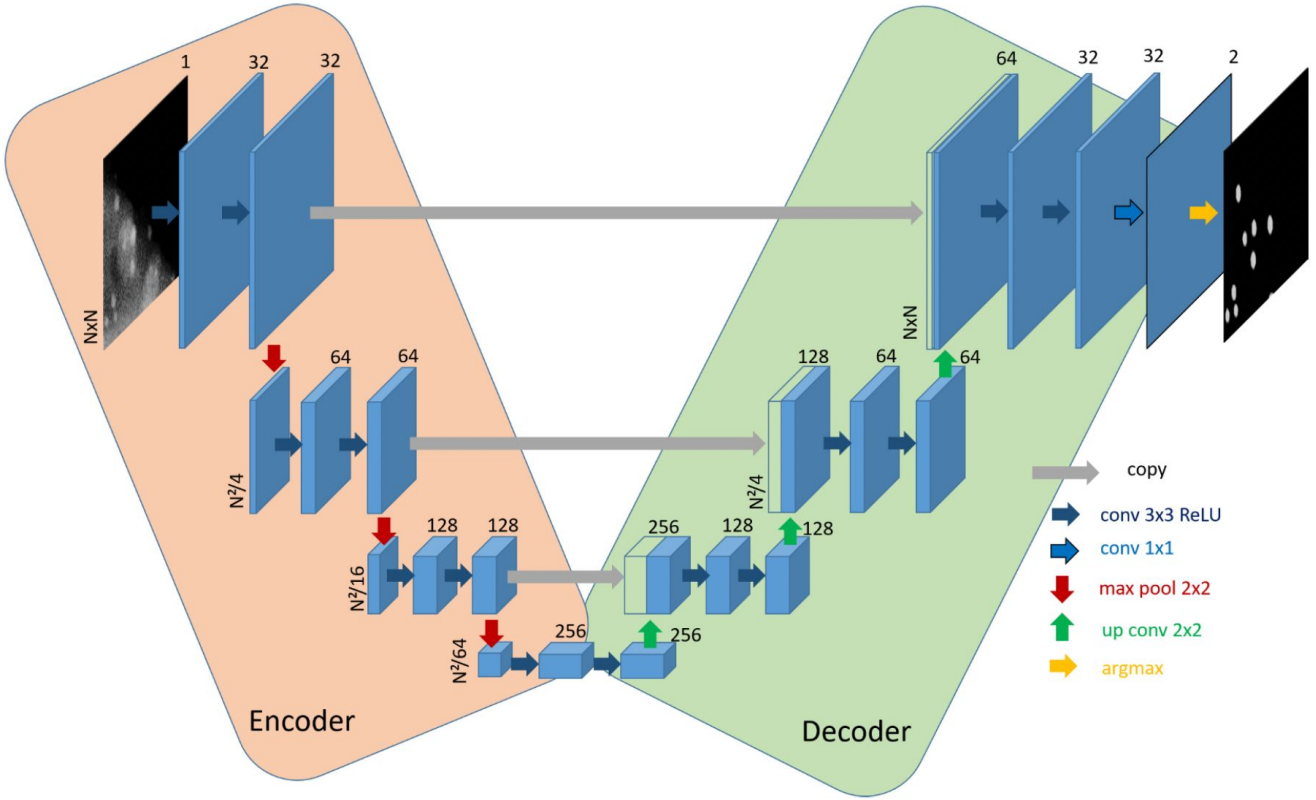
Semantic Segmentation **Instance Segmentation**



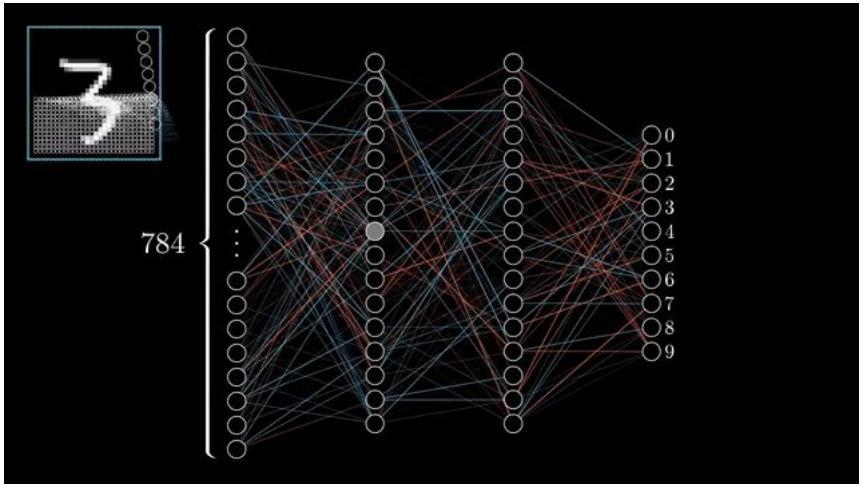


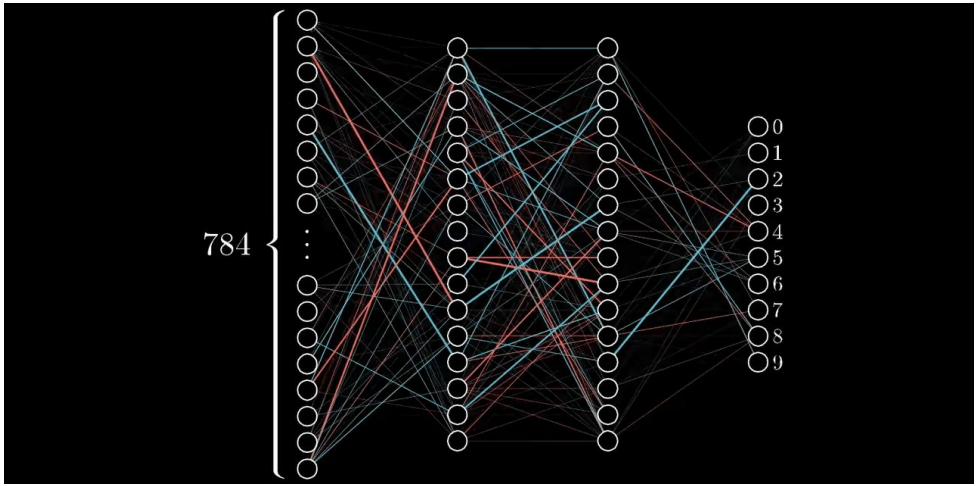
Input Image Semantic Segmentation Instance Segmentation

U-net: Automatic image semantic segmentation



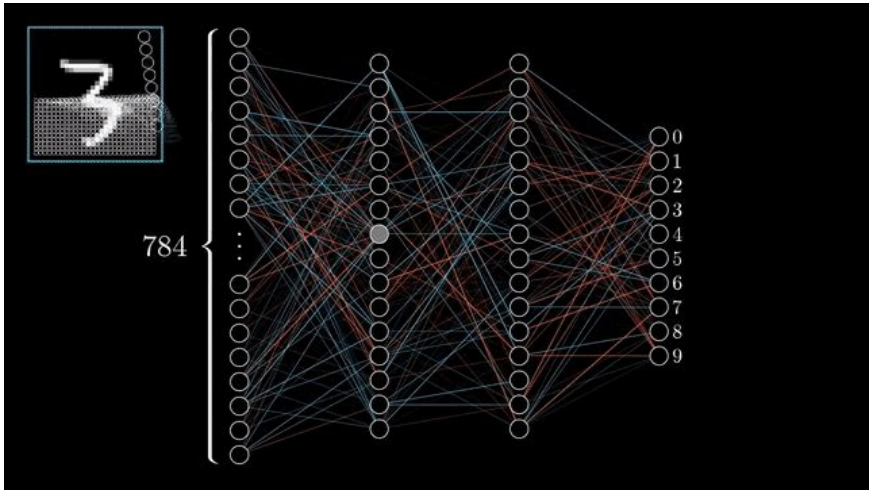
Multi-layer perceptrons (MLP) - Backpropagation

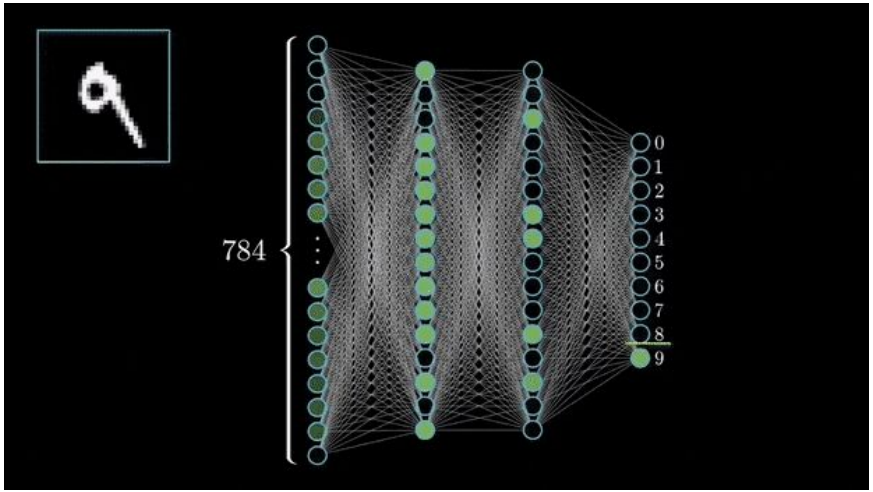




https://pub.towardsai.net/backpropagation-2eeb25201095

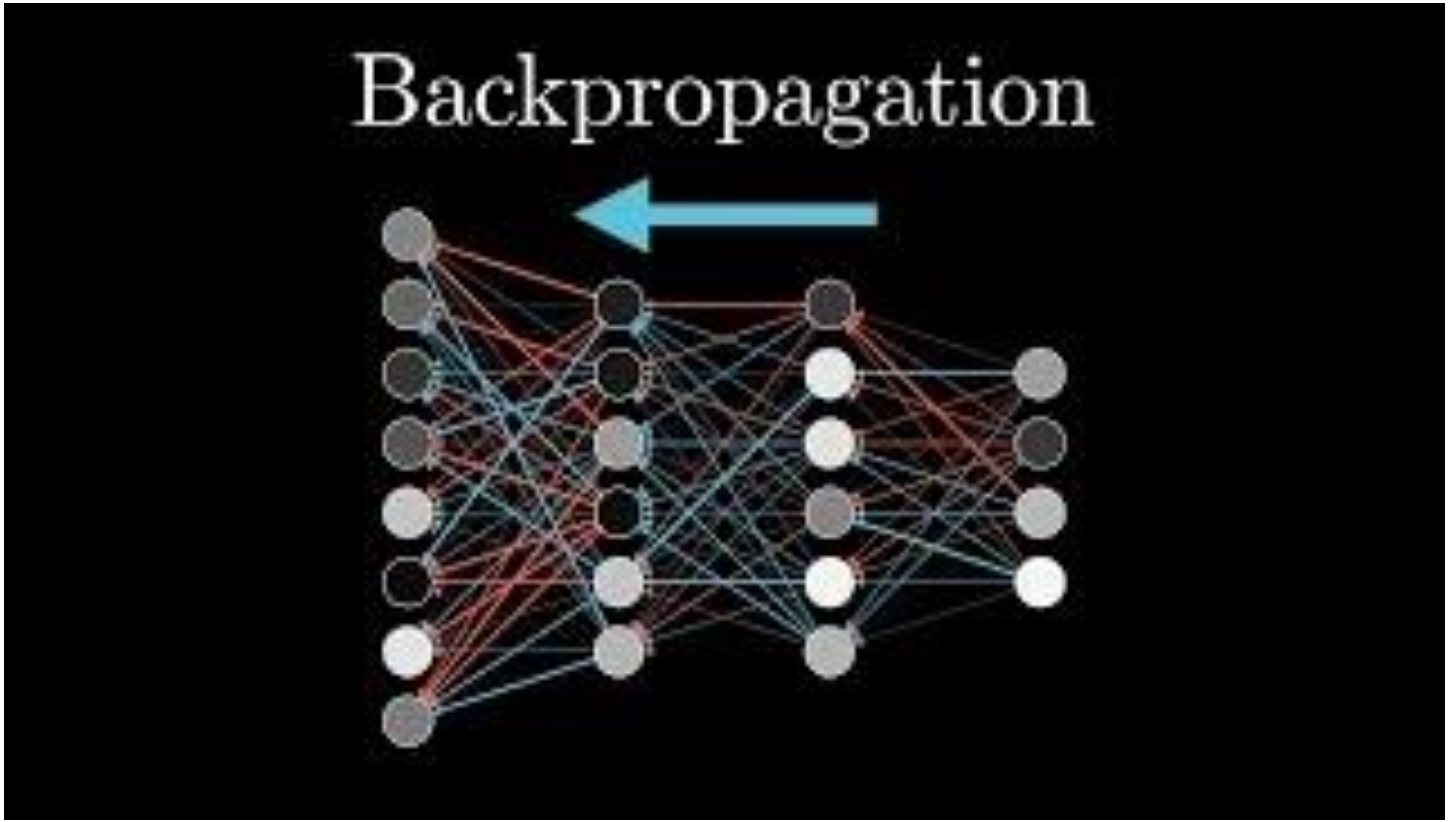
Multi-layer perceptrons - Backpropagation



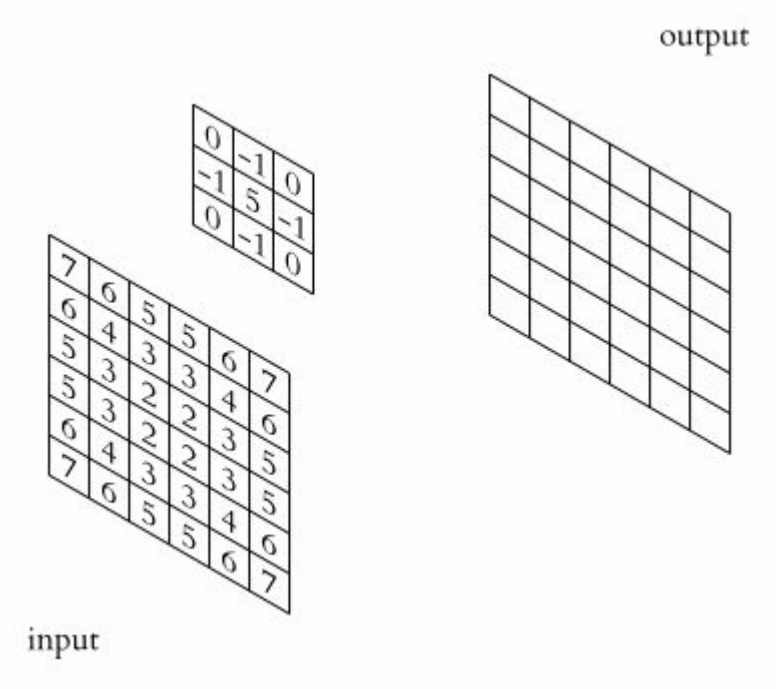


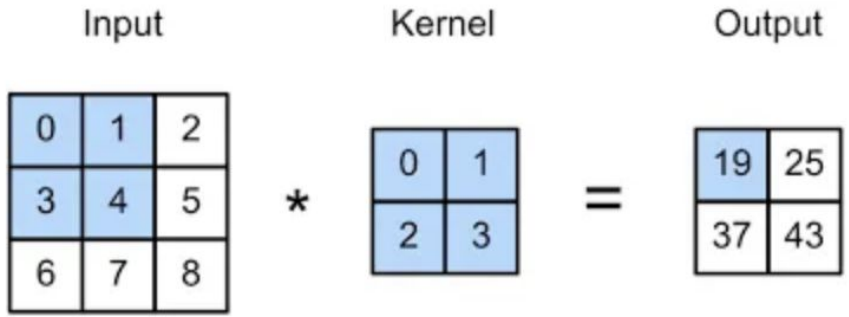
https://medium.com/thedeephub/convolutional-neural-networks-a-comprehensive-guide-5cc0b5eae175

Backpropagation explained intuitively



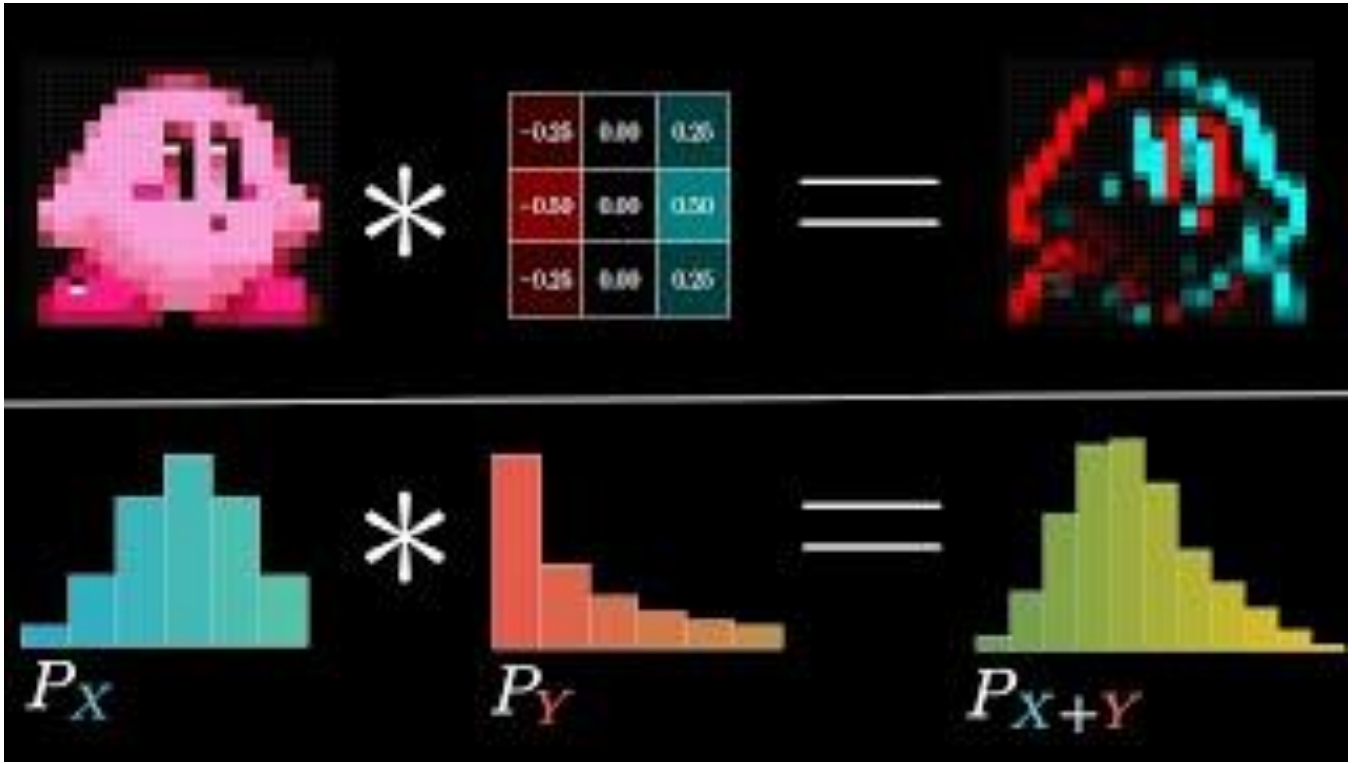
Convolutional Layers





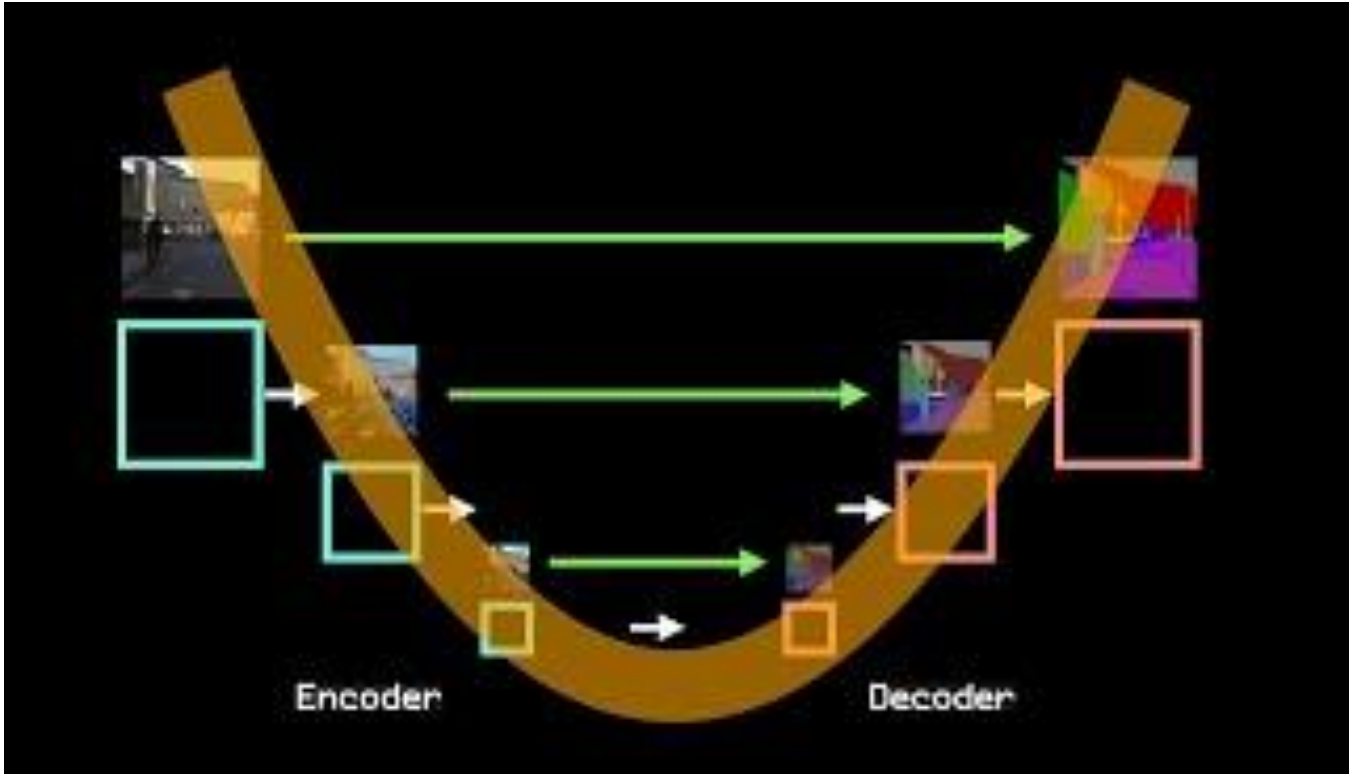
- $(0 \times 0) + (1 \times 1) + (3 \times 2) + (4 \times 3) = \mathbf{19}$
- $(1 \times 0) + (2 \times 1) + (4 \times 2) + (5 \times 3) = \mathbf{25}$
- $(3 \times 0) + (4 \times 1) + (6 \times 2) + (7 \times 3) = \mathbf{37}$
- $(4 \times 0) + (5 \times 1) + (7 \times 2) + (8 \times 3) = \mathbf{43}$

What is a convolution in image processing?



https://youtu.be/KuXjwB4LzSA?si=h6NaxGXPJ4JXFjnV

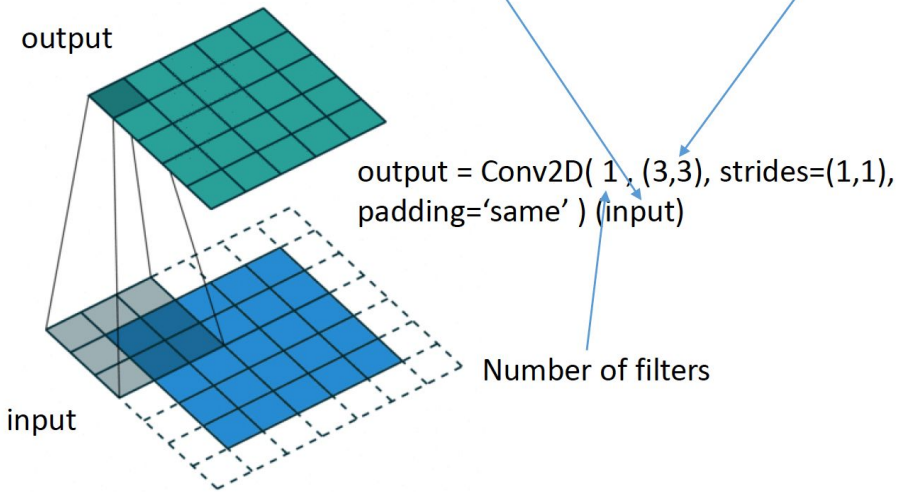
U-Nets explained



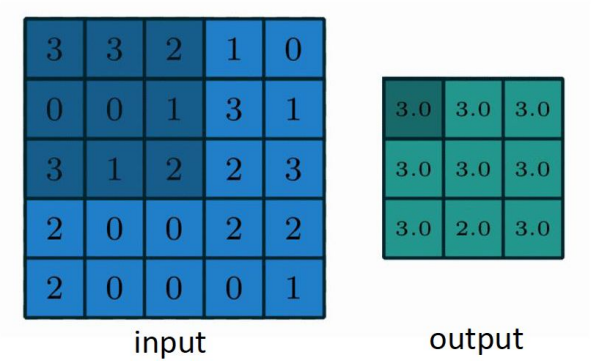
https://www.youtube.com/watch?v=NhdzGfB1q74&ab_channel=rupertai

Convolutions + Max Pooling

2D convolution using a kernel size of 3, stride of 1 and padding



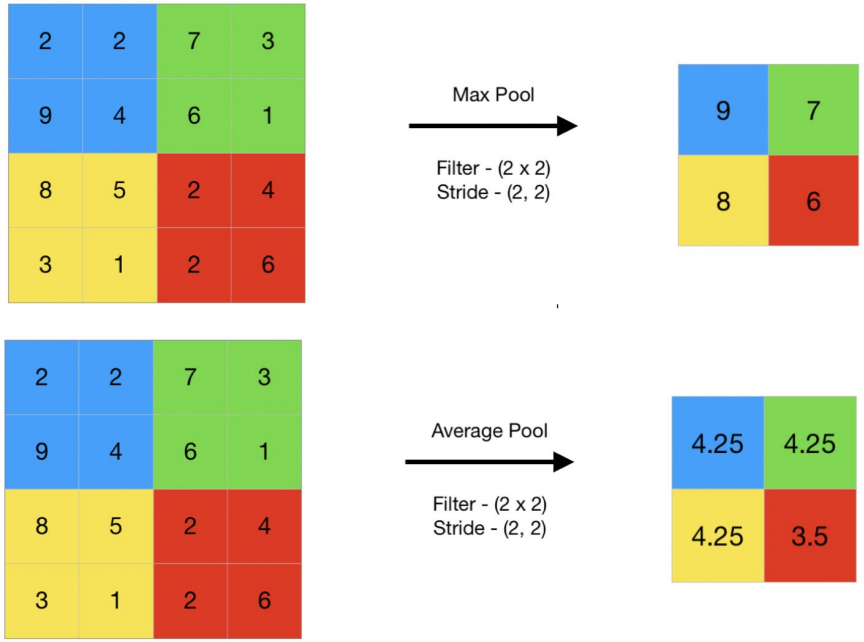
Max pooling kernel size of 3, stride of 1, no padding



output = MaxPooling2D((3, 3), strides=(1,1), padding='valid') (input)

https://medium.com/hitchhikers-guide-to-deep-learning/10-introduction-to-deep-learning-with-computer-vision-types-of-convolutions-atrous-convolutions-3cf142f77bc0

Different pooling layers

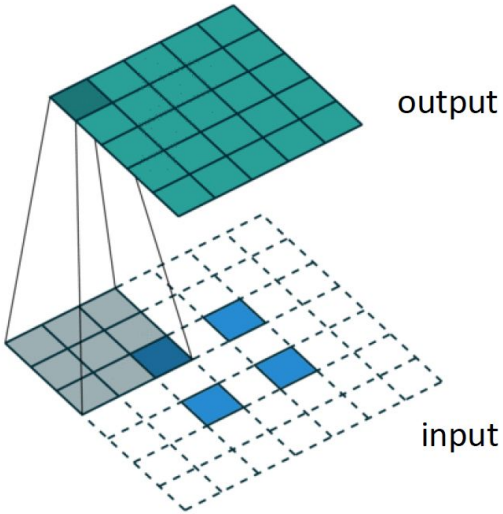


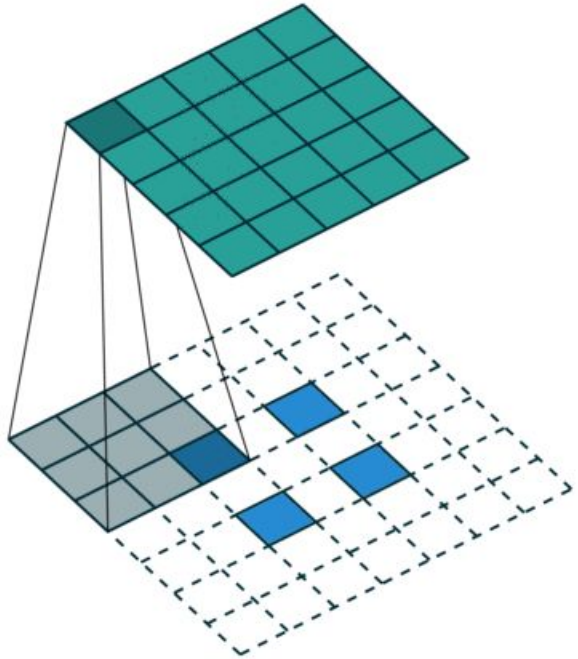

```python
import numpy as np
from keras.models import Sequential
from keras.layers import MaxPooling2D

# define input image
image = np.array([[2.0, 2.0, 7.0, 3.0],
                  [9.0, 4.0, 6.0, 1.0],
                  [8.0, 5.0, 2.0, 4.0],
                  [3.0, 1.0, 2.0, 6.0]])
image = image.reshape(1.0, 4.0, 4.0, 1.0)

# define model containing just a single max pooling layer
model = Sequential(
    [MaxPooling2D(pool_size = 2, strides = 2)])

# generate pooled output
output = model.predict(image)

# print output image
output = np.squeeze(output)
print(output)
```

https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/

Transpose convolution - Upsampling

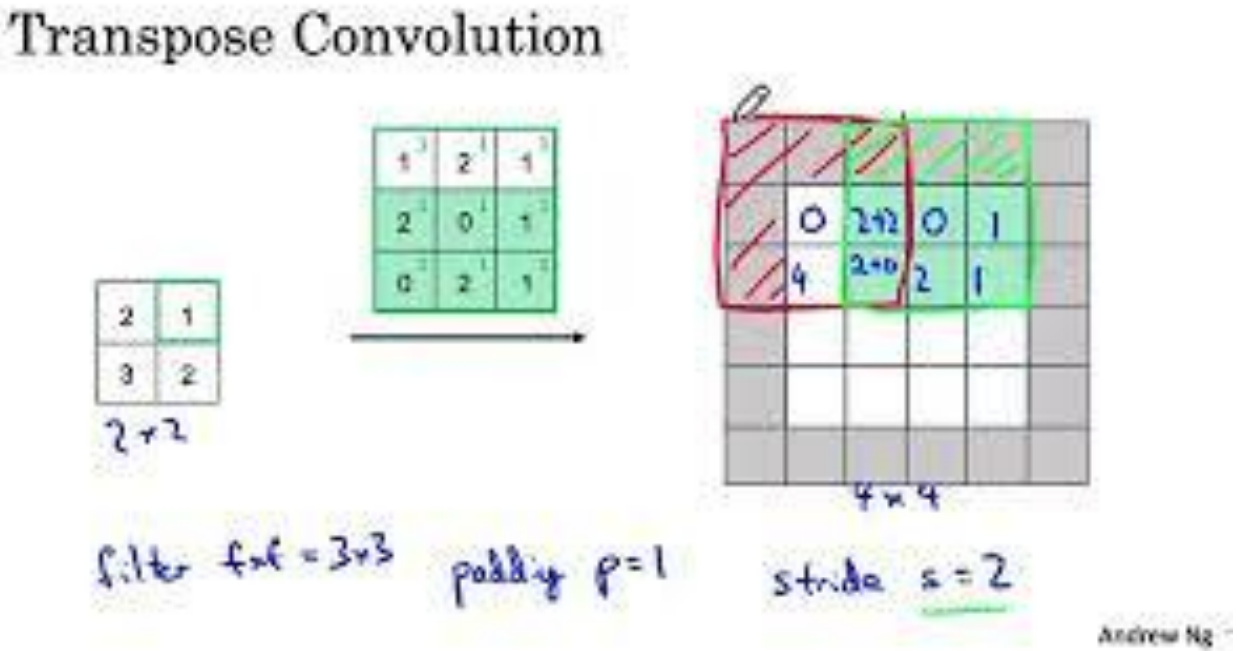
Transposed 2D convolution with padding, stride of 2 and kernel of 3





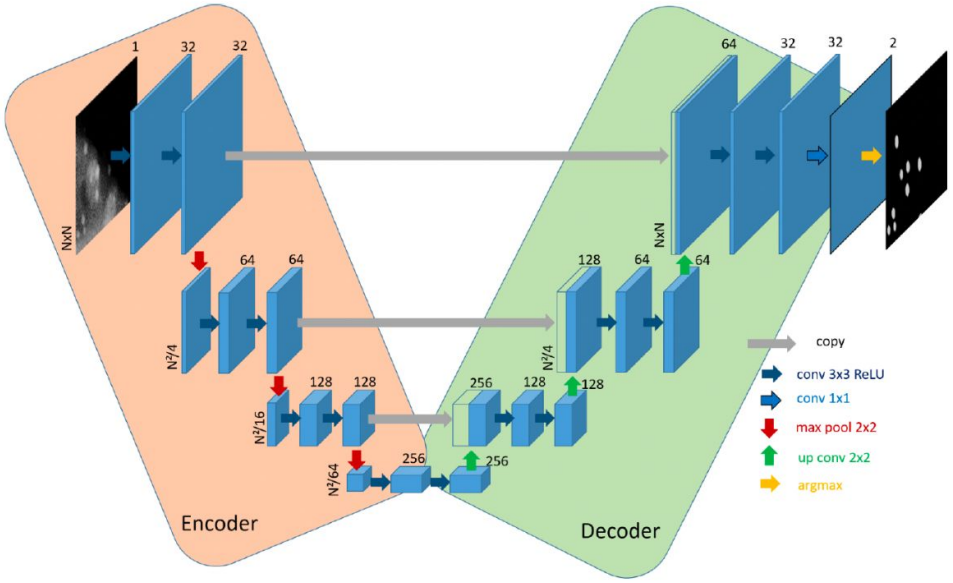
output = Conv2DTranspose(1, (3,3) , strides= (2,2), padding='same') (input)

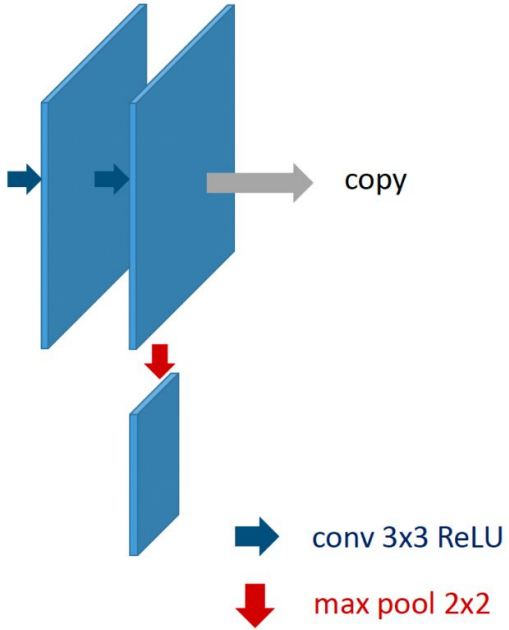
Transpose convolutions explained by Andrew Ng!



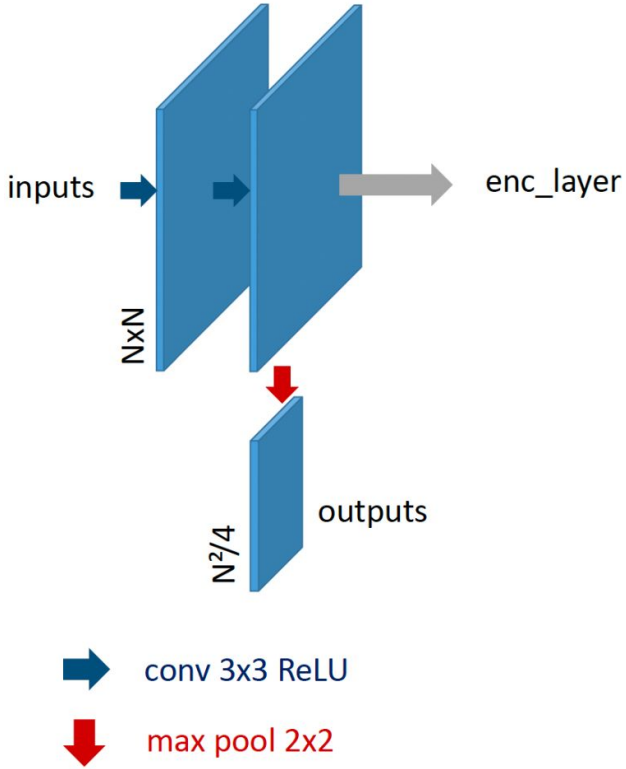
https://www.youtube.com/watch?v=qb4nRoEAASA&ab_channel=%D0%92%D0%B0%D0%BB%D0%B5%D0%BD%D1%82%D0%B8%D0%BD%D0%A0%D0%BE%D0%BC%D0%B0%D0%BD%D0%BE%D0%B2

UNet: Encoder



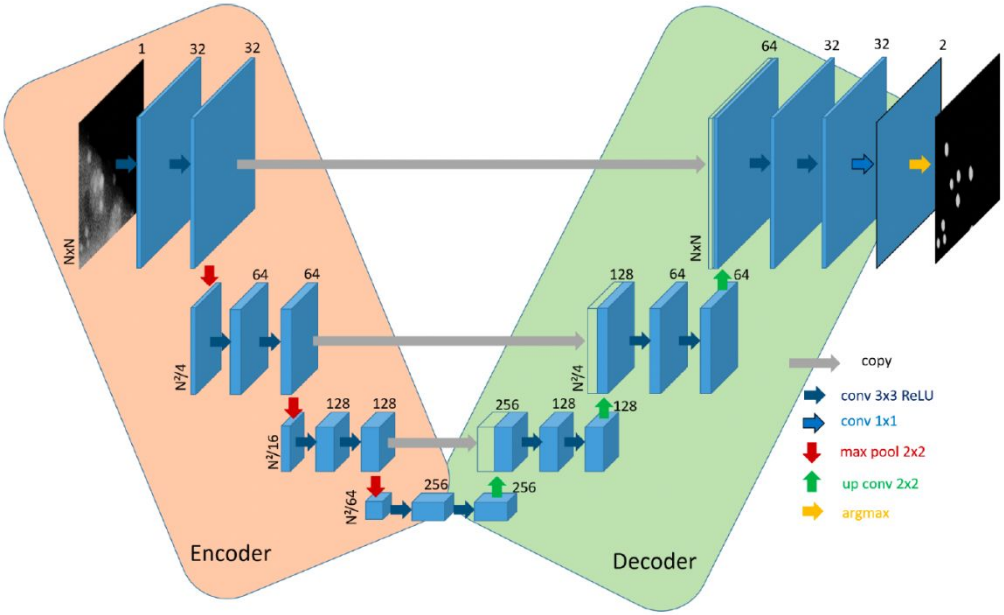


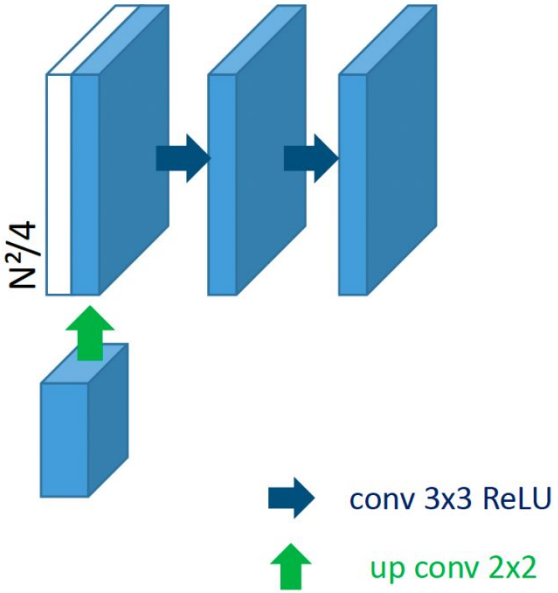
UNet: Encoder



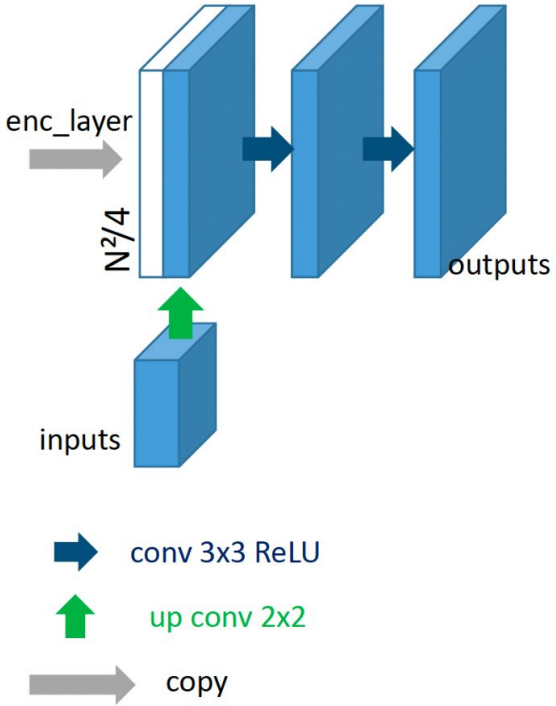

```
c = Conv2D(filters, (3,3), activation='relu',
kernel_initializer=kernel_initializer, padding='same') (inputs)

c = Conv2D(filters, (3,3), activation='relu',
kernel_initializer=kernel_initializer, padding='same') (c)

enc_layer = c

outputs= MaxPooling2D((2, 2)) (c)
```

UNet: Decoder



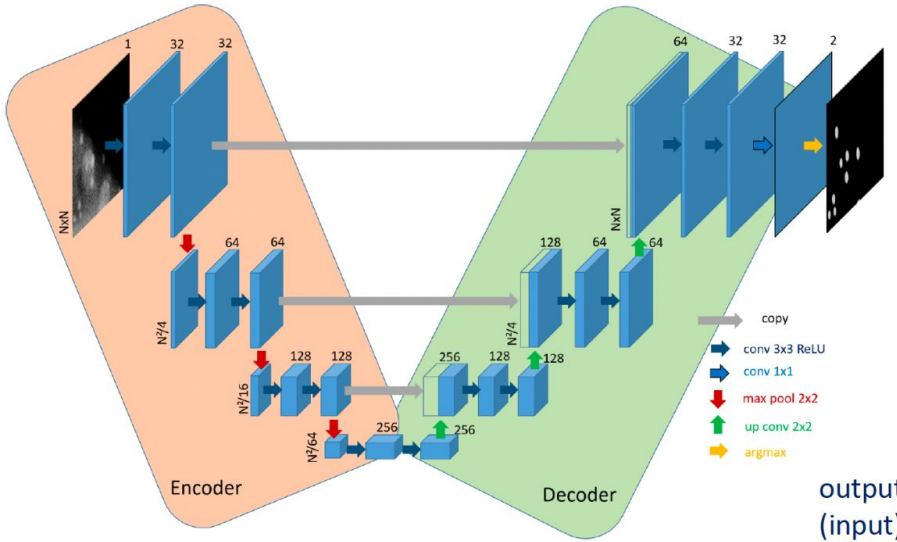


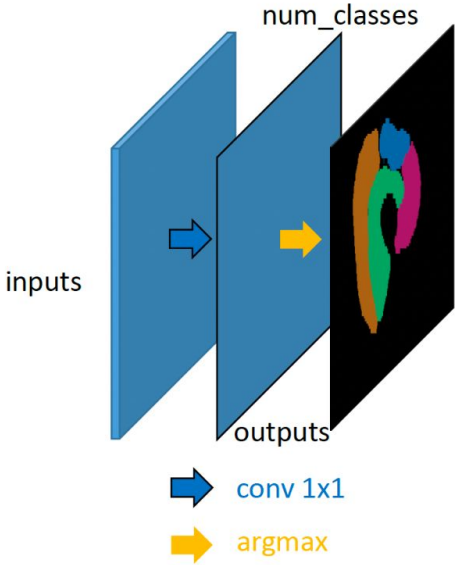
UNet: Decoder




```
c = Conv2DTranspose( filters, (2, 2), strides=(2, 2), padding='same') (input)

c = Concatenate()([c, enc_layer])

c = Conv2D(filters, (3,3), activation='relu', kernel_initializer=kernel_initializer,
padding='same') (inputs)

outputs = Conv2D(filters, (3,3), activation='relu',
kernel_initializer=kernel_initializer, padding='same') (c)
```

UNet: Decoder





outputs = Conv2D(num_classes, (1, 1), activation='sigmoid')(input)

The argmax is done outside the network (loss or metric, display, ...)