ROBOTIC PERCEPTION GROUP

CTU CZECH TECHNICAL UNIVERSITY IN PRAGUE

Image segmentation Using Neural Networks

J. Škovierová, R. Škoviera
julia.skovierova@cvut.cz
B-604b, Jugoslávských partyzánů 1580/3,
160 00 Prague 6, Dejvice

FBMI: F7PMBZAO, F7PMIMA-S

Semantic Segmentation

- Classifies each pixel into a category (e.g., sky, tree, road).
- No distinction between individual instances of the same class.
- Example: all cars = one “car” region.

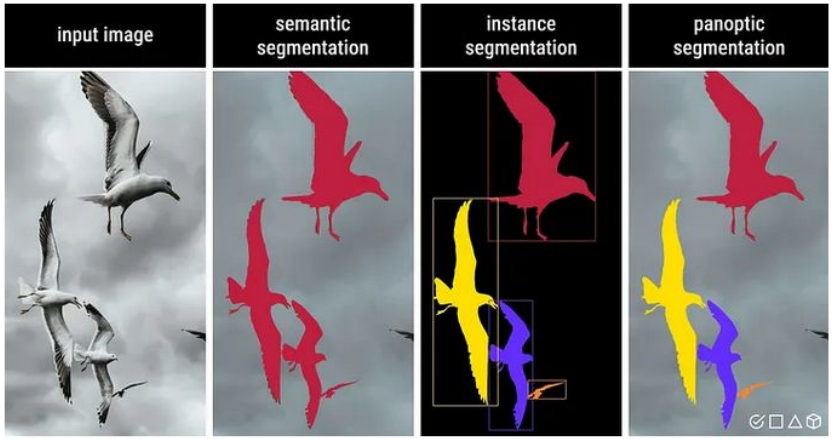


Image source

Instance Segmentation

- Identifies each individual object separately.
- Combines detection + segmentation (e.g., Mask R-CNN).
- Example: distinguishes car #1 vs car #2.

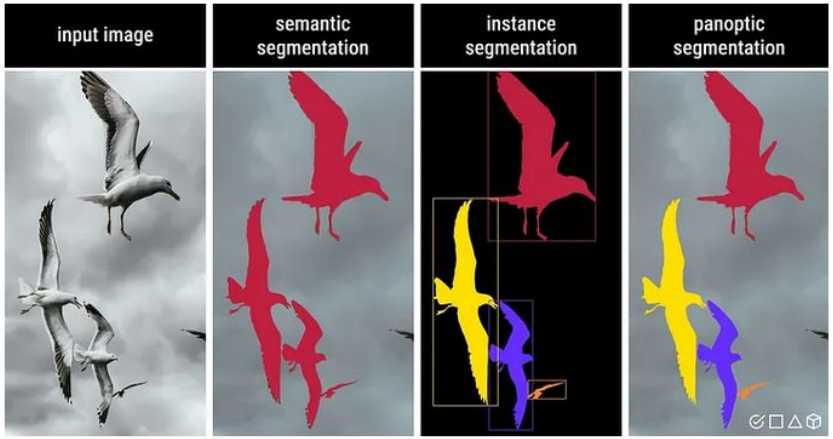


Image source

Panoptic Segmentation

- Unifies semantic and instance segmentation:
  - Semantic segmentation for amorphous “stuff” (sky, road, grass)
  - Instance segmentation for countable “things” (person, car, tree).

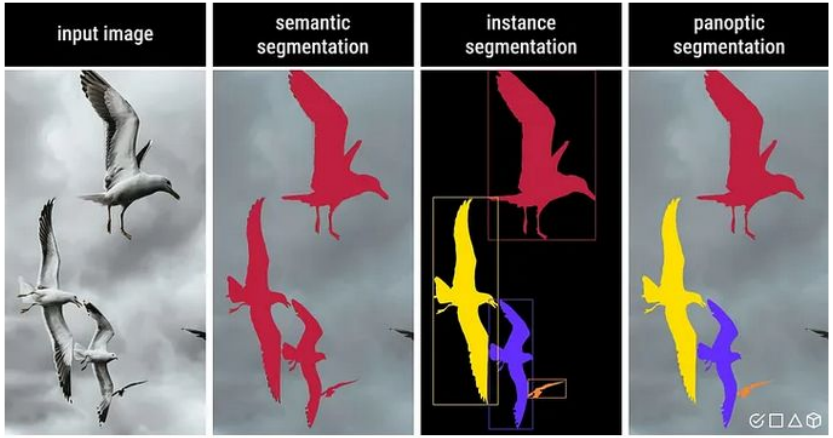


Image source

Motivation for Neural Networks in Segmentation

- Traditional methods: thresholding, k-means, mean-shift
- Limitations: sensitive to noise, lighting, texture

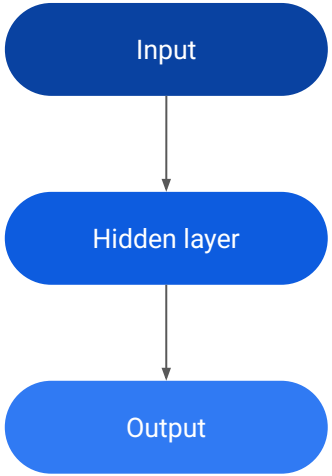
- Neural networks learn features automatically
- Applications: medical imaging, autonomous driving, satellite images

Biological neural networks - inspiration

- Success in many other fields
  - basically, nature had time to develop some pretty good systems (~evolution)
- Human brain:
  - neurons are slow compared to logic gates/transistors (order of milliseconds)
  - yet the brain is faster and more accurate on some tasks
    → massive parallelism
    - 10^{11} neurons
    - 10^{15} synapses (approx. 10 000 synapses per neuron)
  - robustness & good generalization
  - (biological) neural networks with similar structure can adapt to perform different tasks on different modalities

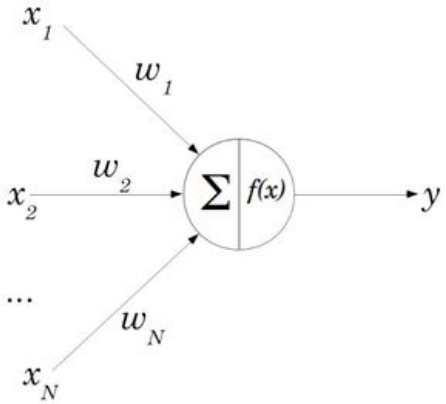
Fundamentals of Neural Networks

- Neural network = interconnected neurons
- Key components:
  - weights, activations, layers
- Training:
  - Backpropagation / gradient descent
- Activation functions:
  - ReLU, sigmoid, softmax
- Output:
  - classification, regression,...



Perceptron

- Biological inspired mathematical model of artificial neuron
- Frank Rosenblatt (1957) neural model perceptron
- Combination of the McCulloh & Pitts model and Hebbian learning



$$y = f\left(\sum_{i=1}^{n} x_i w_i - \theta\right)$$

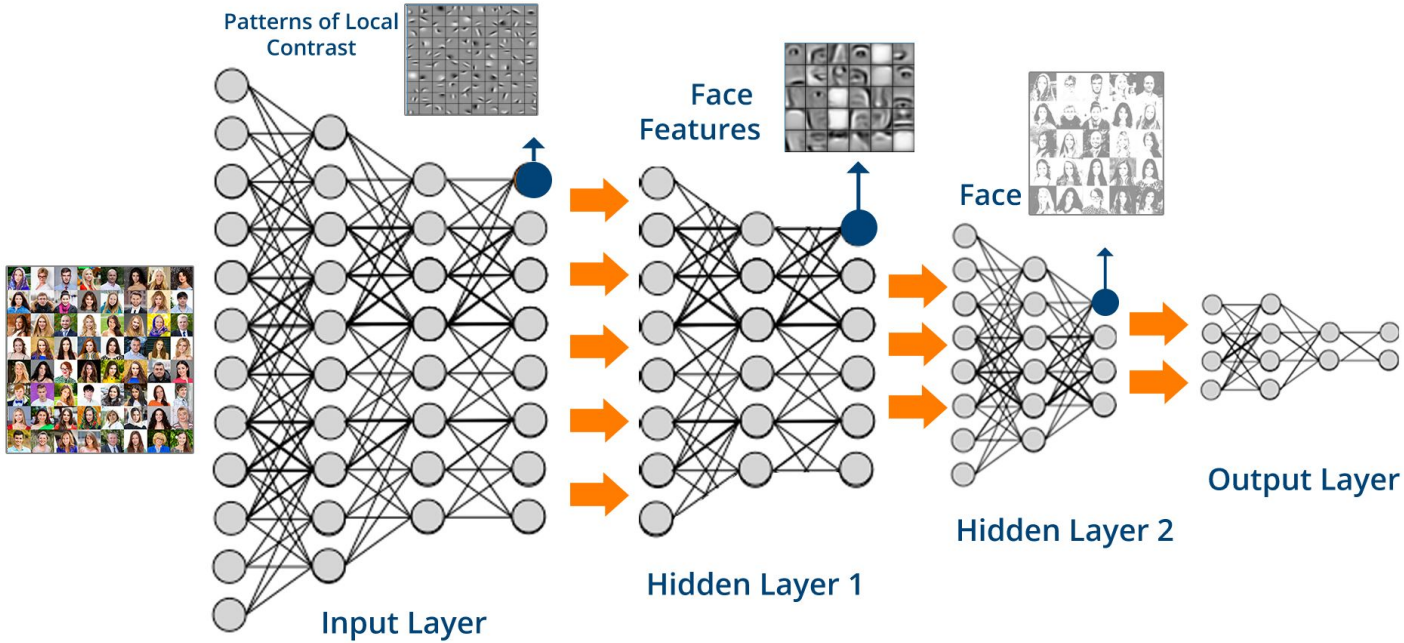
or

$$y = f\left(\sum_{i=1}^{n+1} x_i w_i\right), x_{n+1} = -1$$

- The **activation function** f has to be bounded, monotonous, differentiable; e.g. the sigmoid function σ (originally, step function was used, differentiation not necessary for early day perceptron)

Deep neural networks

- Hinton (2006)
- number of layers > 3 (i.e., more than one hidden layer);
- today ≫ 3



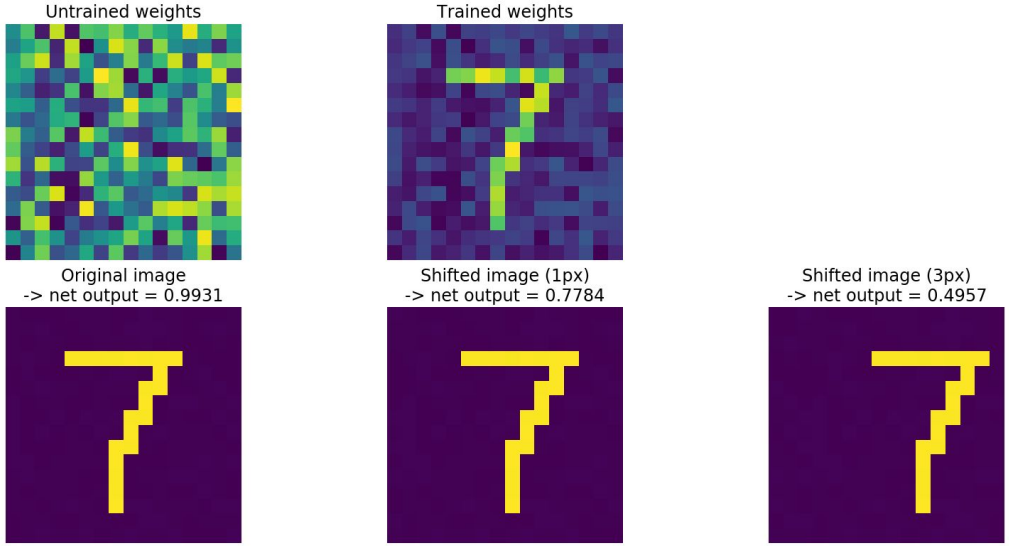
Types of learning in ANN

- **Supervised learning**: weights are adjusted according to the **desired** output (perceptron, MLP, RBF, RNN)
- **Reinforcement** learning: extension of the supervised learning; weights are adjusted according to the reward (MLP, RBF)
- **Unsupervised** learning: weights are adjusted according to the **statistics** of the input (Hebbian net, SOM)
- Other possibilities:
  - hyper-parameter learning (auto-generated neural networks)
  - Genetic algorithm: not a learning paradigm but can be used for parameter or even hyper-parameter training

Fully Connected Artificial Neural Networks

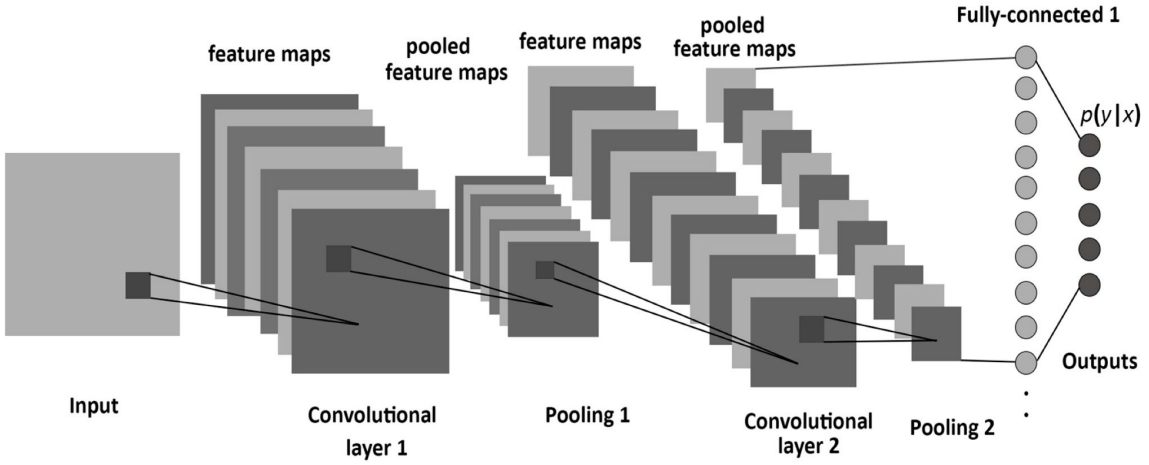
- MLP, recurrent networks, and all classical networks are fully connected (FC)
- work well in many cases but...

- Example of FC network on images:
  Input: 16 x 16 grayscale image
  Network: single linear neuron with 256 inputs (162) and sigmoid activation function
  Training on one image (target response = 1) vs random noise (target response = 0)
  (weights reshaped to a 16 x 16 image, original shape: 1 x 256 vector)



Convolutional neural networks

- (some) invariance to geometric transformations in FC network small (spatial) change in input = large change in output
- sparse connection, parameter sharing
- preserving spatial information (to some degree)
- larger input size:
  - RGB images weights per single neuron in FC network:
    32 x 32 x 3 ≅3000
    256 x 256 x 3 ≅ 200 000



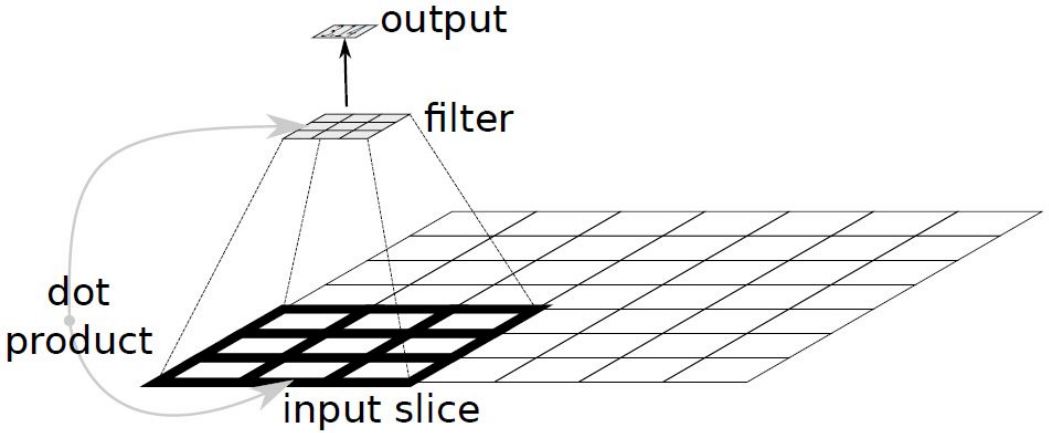
CNN - history

- ~1960: Discovery of the function of neurons in visual cortex
  - Visual cortex has simple (convolution) and complex (pooling) cells
- 1980: First articial convolutional neural network:
  Fukushima, Kunihiko. "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaected by shift in position." Biological cybernetics 36, no. 4 (1980): 193-202.
- 1989: Early convolutional models
- 1998: LenNet5(4) on MNIST (handwritten digits) outperforms the compared methods (rst LeNet version in 1989)
- 2004 → GPU accelerated CNN implementations
- 2012: AlexNet on ImageNet wins ILSVRC2012 (use of GPU allowed fast deep net)
- 2020→ : CNNs with dozens of layers outperform all other models in image recognition tasks (and many others)...given enough data

Convolution in NN

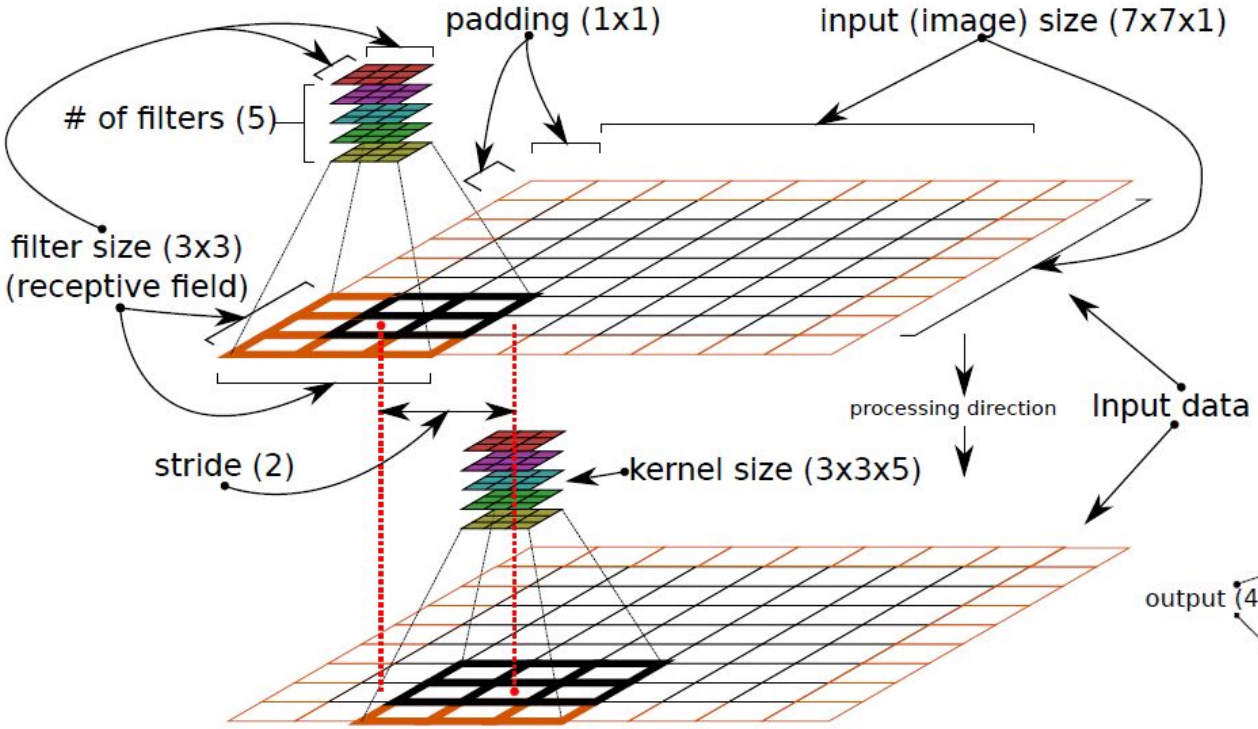
Filter - a matrix of weights
Input slice - a slice of the full input equal to the size of the filter
Output - (of a single step of the convolution) a single number

- dot product between the filter and the input slice

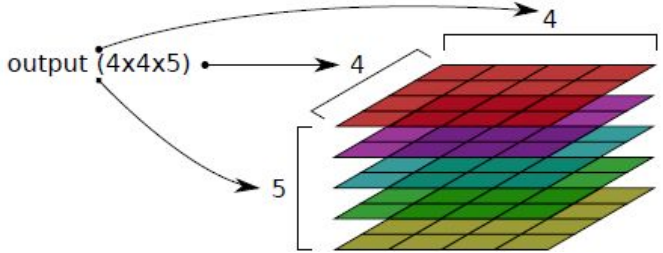


- Full output is a matrix with the individual outputs
- Activation function is applied after convolution (same as ANN)

Convolution in NN

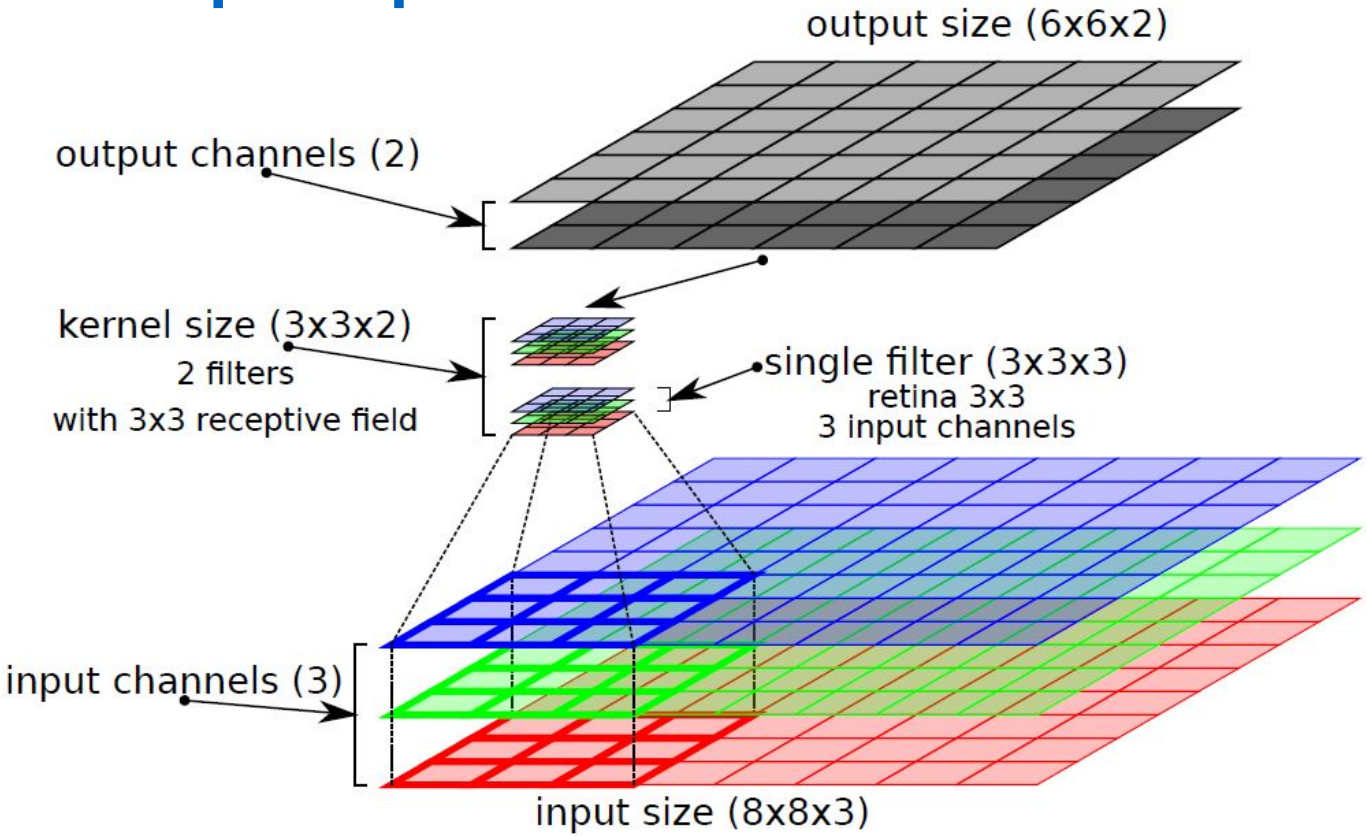


- Output: feature maps (activation maps)



Convolution on multiple input channels

- each filter is applied to all input channels at once (through the depth of input)
- number of output channels not dependent on number of input channels, only on number of filters



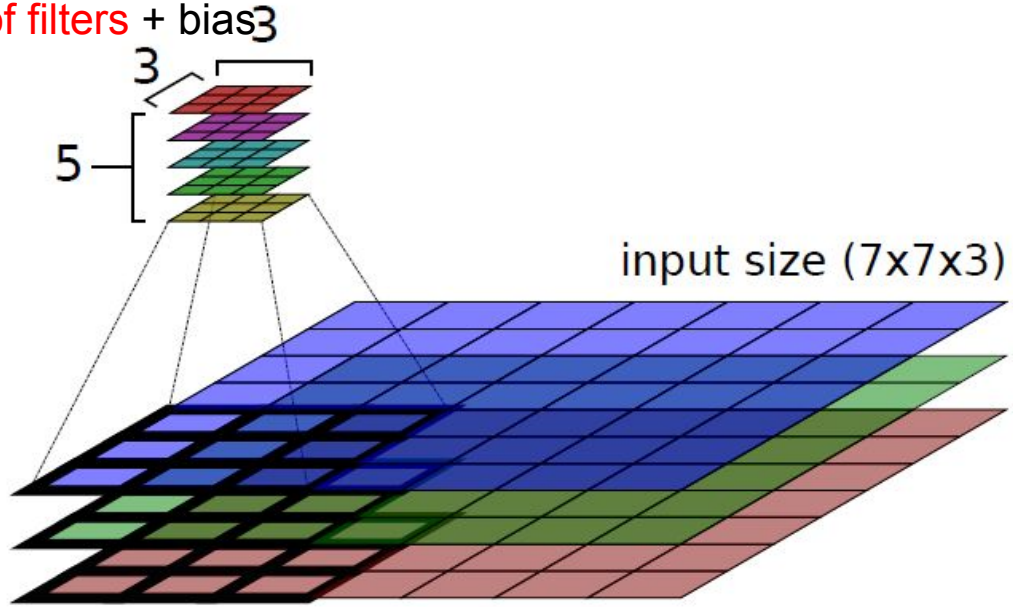
Convolution parameter sharing

Filter size: (receptive eld) 3x3
Number of filters: 5
Input size: (width x height x depth) 7x7x3
Number of parameters:

receptive field x input depth x number of filters + bias

$3 * 3 * 3 * 5 + 5 = \mathbf{140}$

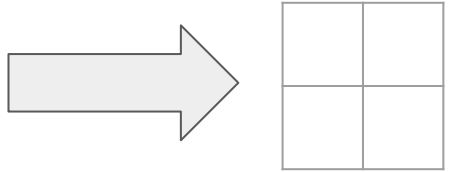
(Fully connected network with
5 neurons: **740** parameters)



Pooling

- Downsampling
- Usually done after activation
- Reduces size of feature maps
- Increases spatial invariance and robustness against noise
- Size and stride can be specfiied just as with convolution
- Maintains the number of channels
- Example of max pooling (size = 2 x 2, stride = 2):

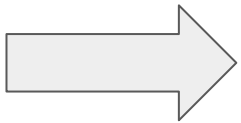
1	5	4	6
2	9	8	7
6	4	5	3
6	7	2	1



Pooling

- Downsampling
- Usually done after activation
- Reduces size of feature maps
- Increases spatial invariance and robustness against noise
- Size and stride can be specied just as with convolution
- Maintains the number of channels
- Example of max pooling (size = 2 x 2, stride = 2):

1	**5**	4	6
2	**9**	8	7
6	4	5	3
6	7	2	1

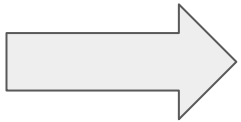


9	

Pooling

- Downsampling
- Usually done after activation
- Reduces size of feature maps
- Increases spatial invariance and robustness against noise
- Size and stride can be specied just as with convolution
- Maintains the number of channels
- Example of max pooling (size = 2 x 2, stride = 2):

1	5	**4**	**6**
2	9	**8**	**7**
6	4	5	3
6	7	2	1

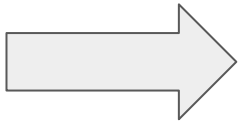


9	8

Pooling

- Downsampling
- Usually done after activation
- Reduces size of feature maps
- Increases spatial invariance and robustness against noise
- Size and stride can be specied just as with convolution
- Maintains the number of channels
- Example of max pooling (size = 2 x 2, stride = 2):

1	5	4	6
2	9	8	7
6	**4**	5	3
6	**7**	2	1

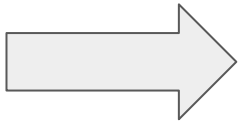


9	8
7	

Pooling

- Downsampling
- Usually done after activation
- Reduces size of feature maps
- Increases spatial invariance and robustness against noise
- Size and stride can be specied just as with convolution
- Maintains the number of channels
- Example of max pooling (size = 2 x 2, stride = 2):

1	5	4	6
2	9	8	7
6	4	**5**	**3**
6	7	**2**	**1**



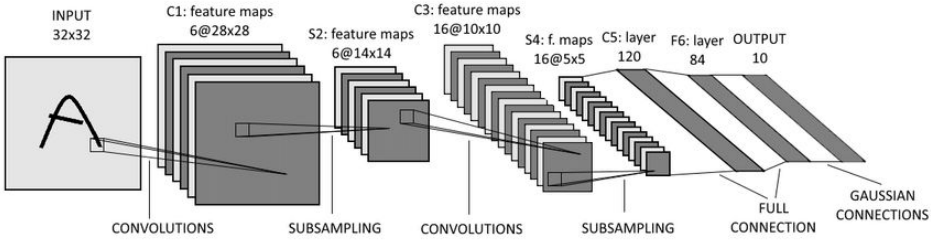
9	8
7	5

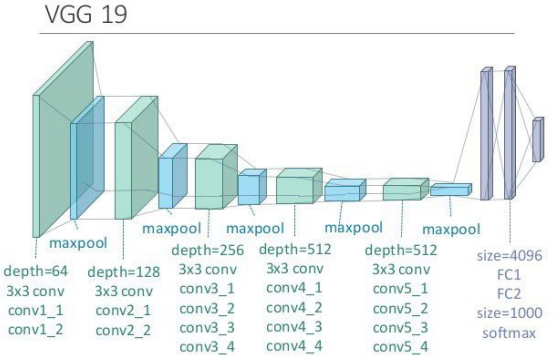
Composition of CNN

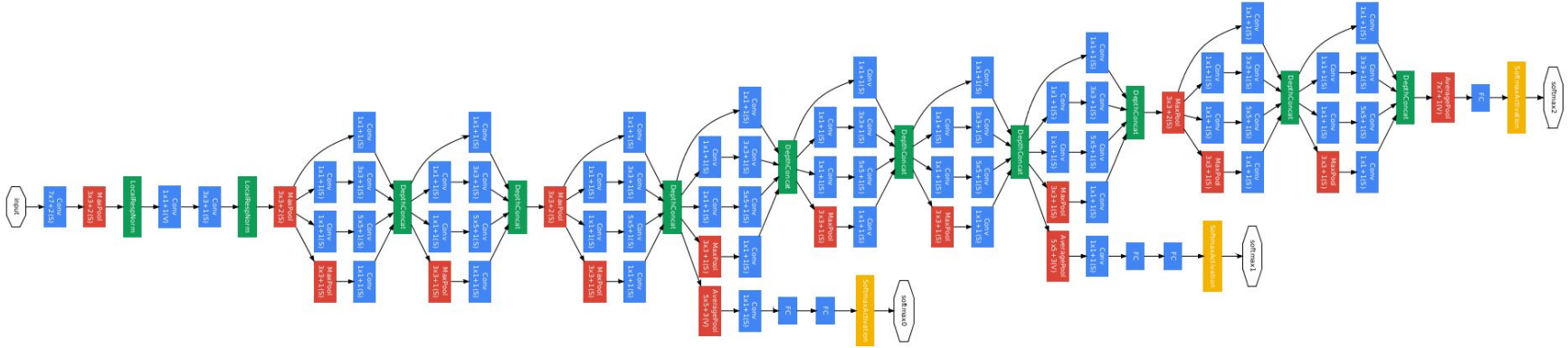
- **Hyperparameters**
  - # and types of layers (types of activations)
  - kernel/filter size (size of receptive elds of neurons)
  - # of filters = neurons (# of output channels)
  - padding (size and type)
  - stride
- **Types of layers**
  - Convolutional layer
  - Activation layer (same as in FC networks)
  - Pooling layer: Max, Mean
  - Upsampling, deconvolution layers (not mathematical deconvolution)
  - CNN often used in combination with FC layers at the end (classification task)
  - Modern CNNs:
    - modular approach swappable backbones
    - multiple pathways (i.e. not only single path for processing flow)

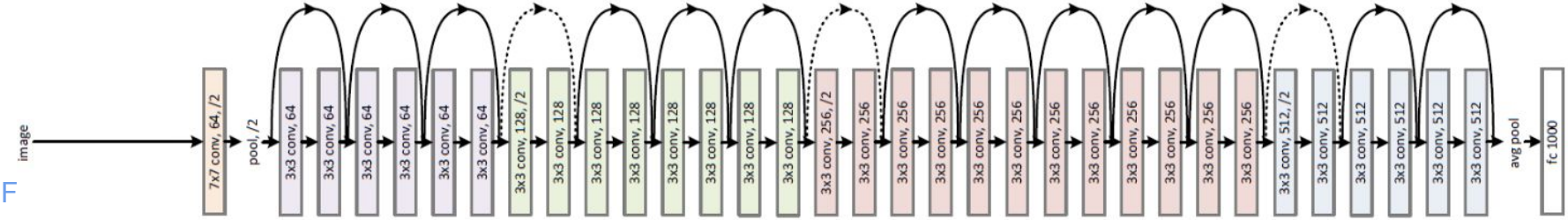
ROBOTIC PERCEPTION GROUP

CNN Zoo









F

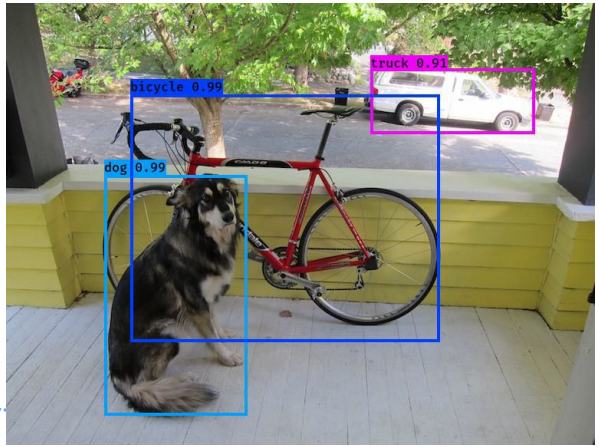
Basic structures & Tasks & Applications

- Fully convolutional network (FCN)
  - only convolutional layers (well, except pooling, activation, recurrent,... simply no fully connected layers)
  - output is usually a 1D vector
  - feature extraction, labeling, transformation
- FCN + fully connected layers at the end
  - a simple MLP on top of the 1D vector output from FCN
  - classification, labeling
- FCN + region proposal networks (RPN)
  - more complex architecture, RPN usually convolutional in nature
  - pre-selects regions where additional processing should happen
  - object recognition/detection
- Encoder-decoder architecture
  - usually fully convolutional
  - input size ≅ output size, unlike FCN
  - semantic segmentation

Basic structures & Tasks & Applications

- On images
  - as a whole:
    - Image recognition
    - Labeling
    - Feature extraction
  - on regions
    - object recognition/detection
  - on pixels
    - semantic segmentation
    - instance segmentation
    - panoptic segmentation
- On speech, text, radio signals, bio signals (EEG, EKG,..),...

Basic structures & Tasks & Applications



Autoencoders (1)

- Originally: Non-linear PCA (with mapping and demapping sections)
- Network is trained to generate the same values on output as were provided on input
- Forced to reduce the information amount (hopefully reduces redundant information)
  - **bottleneck** → reduced representation of the input - **latent space**

$X \rightarrow L \rightarrow X$

Autoencoders (2)

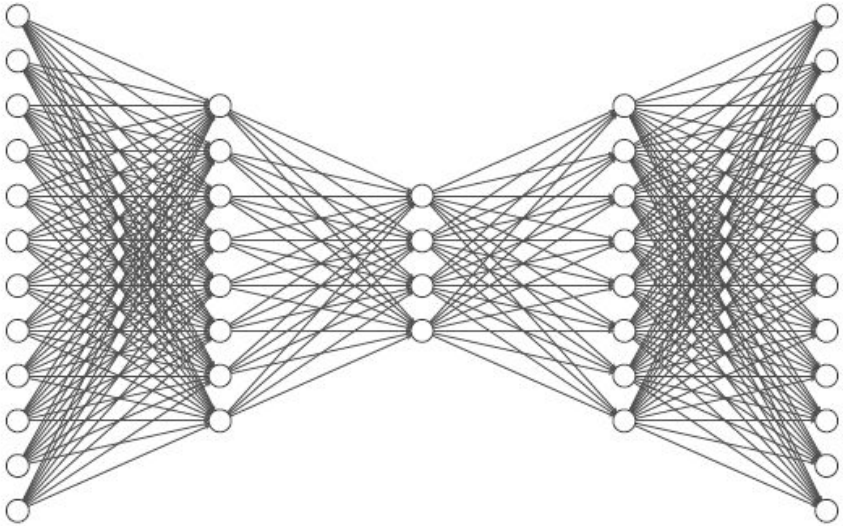
- Consists of two main sections:
  - encoder (convolution): transforms an input into the latent space $X \rightarrow L$
  - decoder (deconvolution): transforms a vector from the latent space into an output from original space $L \rightarrow X$
- Easy to gather data (no labeling needed, sometimes)

- Generalization: **Encoder-decoder network**
  $X \rightarrow L \rightarrow Y$
  - output does not necessarily have to be the same or of the same shape or from the same domain or of the same modality…

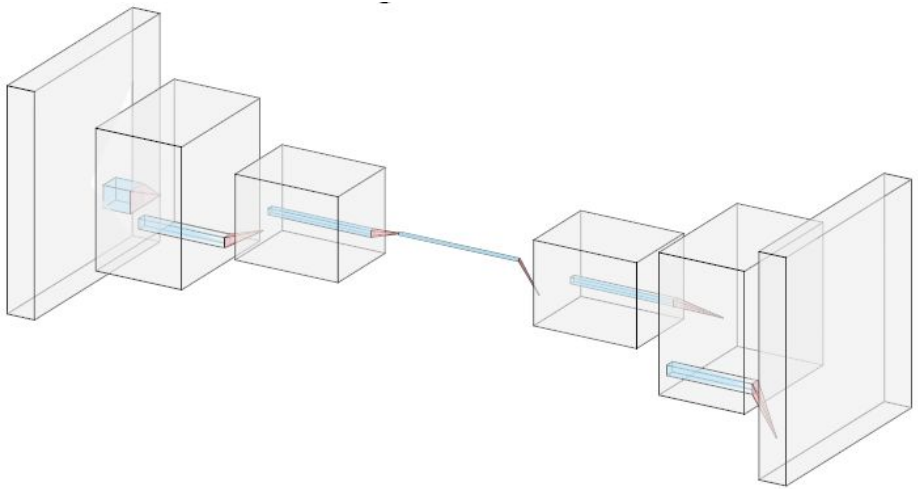
  e.g. segmentation

Autoencoders (3)

FC AE

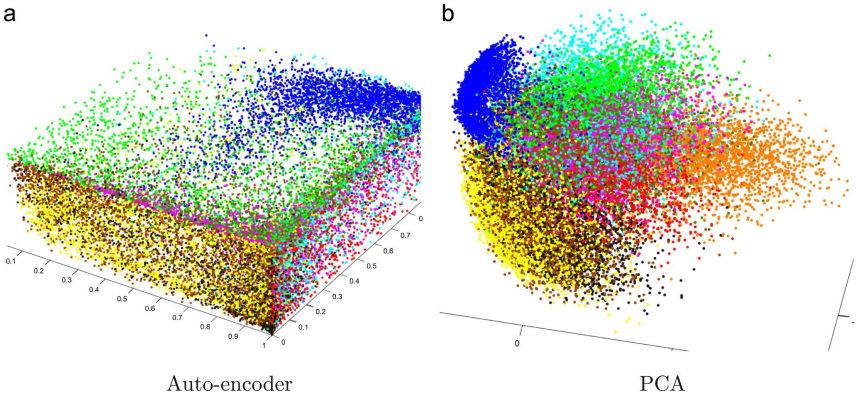


CNN AE

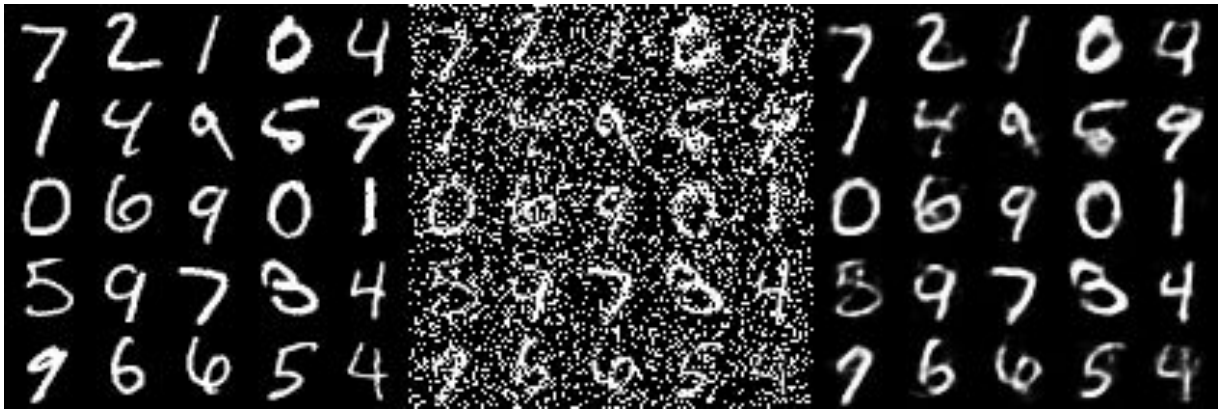


Autoencoder applications

- Dimensionality reduction (NLPCA)
  - visualization, compression

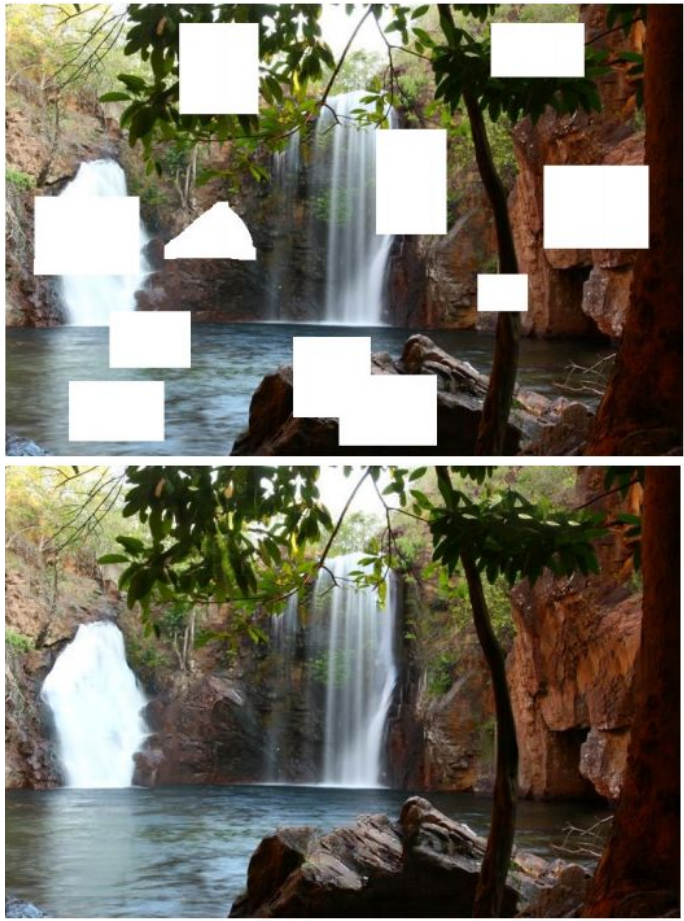


- Denoising
  - easy to generate data: input: image + noise ! target: image without noise (i.e. original image)



Autoencoder applications

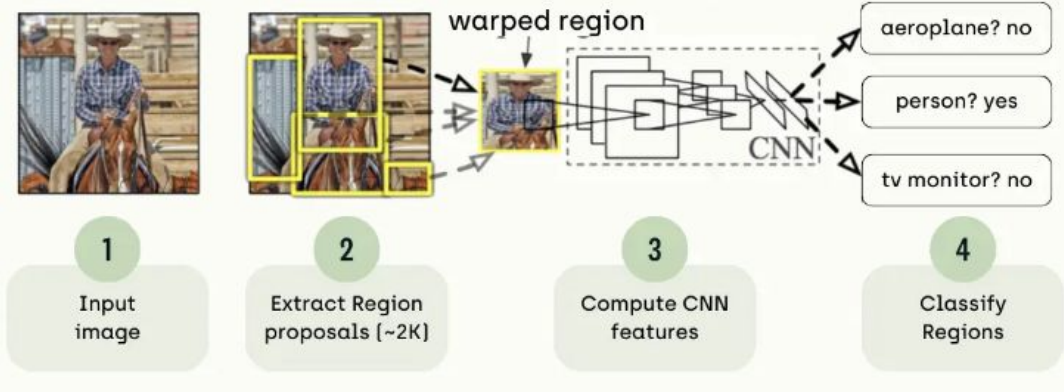
- Inpainting
  - image reconstruction:
    - Parts of images are left out, network tries to ll the missing parts.
  - image augmentation:
    - Network lls the missing parts with novel data



- Data generation (reconstruct from artificially generated latent space samples)
- Fault detection (assumption: reconstruction error on good data will be small)

RCNN

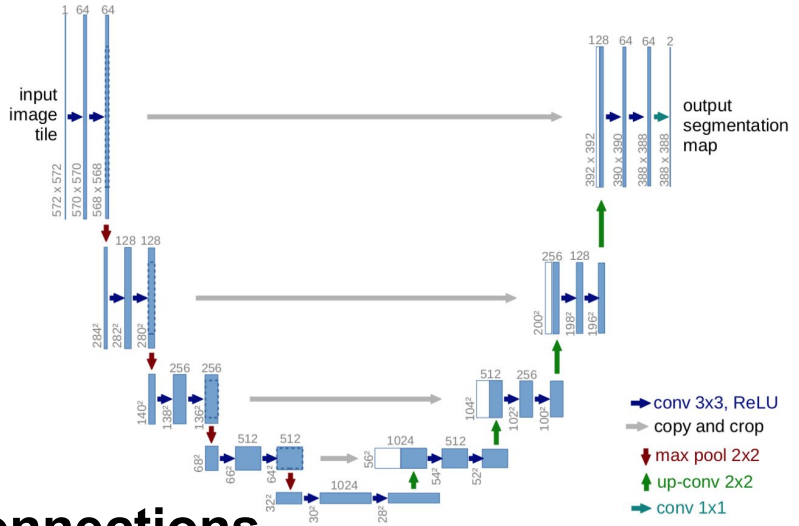
- Region Based Convolutional Neural Networks
- Object detection
  1) generate a set of candidate regions or "region proposals"
  2) use a CNN to classify each region as containing an object or not
  3) refine the object's location , classification
- Limitations:
  - high computational cost, lack of context
  - impractical for real-time object detection on low-powered devices



https://kili-technology.com/blog/yolo-algorithm-real-time-object-detection-from-a-to-z

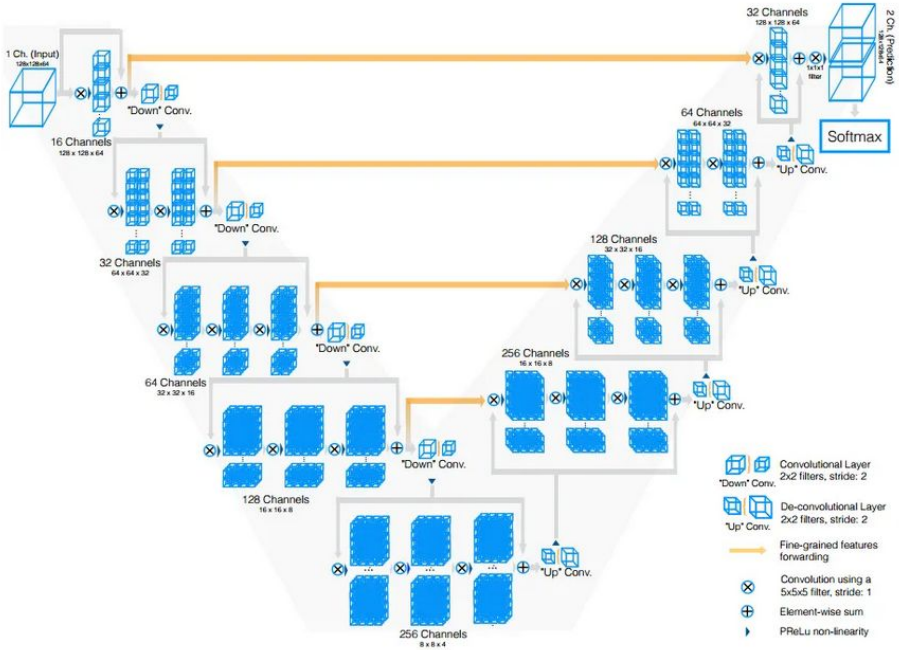
U-net

- Extension of the idea of an autoencoder
- Output does not need to be the same as input
- Used to transform image data from one domain to another;
  e.g., image/semantic segmentation
- Retains spatial information using **skip connections**
- Cannot be split, the two components are now referred to as **contraction** and **expansion**



V-net

- Volumetric Image Segmentation
- V-Net learns to predict segmentation for the entire volume at once
- Kernel is using 5x5x5 voxels to capture 3D context



V-net - example

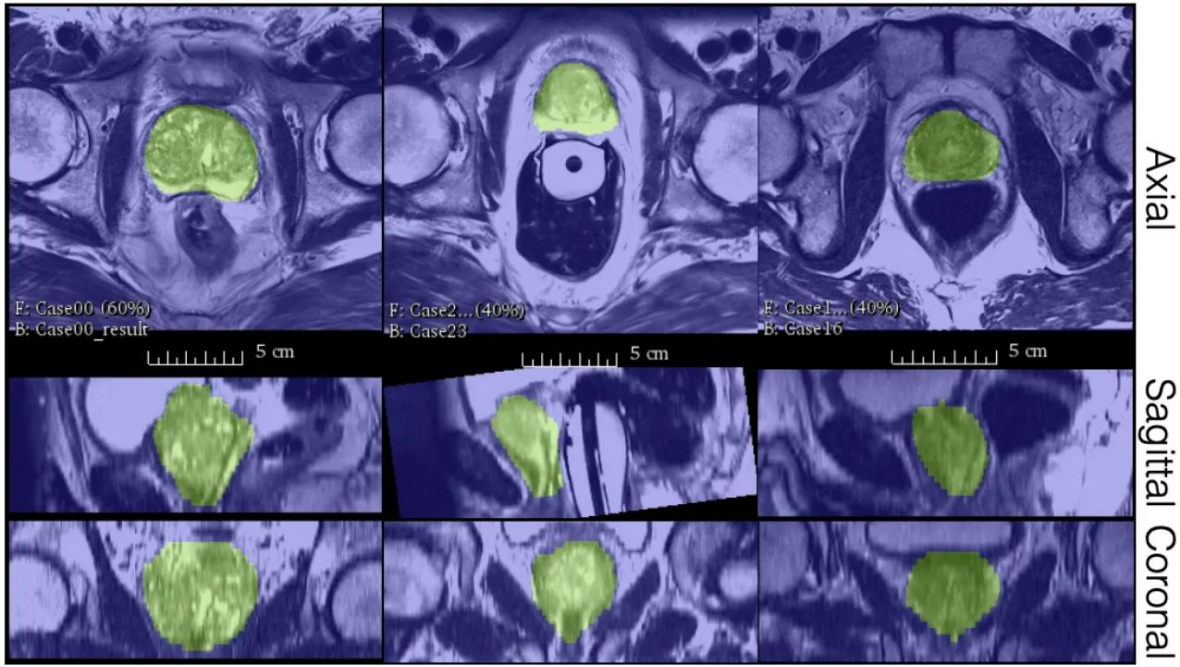


Fig. 4. Qualitative results on the PROMISE 2012 dataset [7].

https://medium.com/@r-rahulsingh/v-net-architecture-volumetric-image-segmentation-5e2a99c91dc6

A puppy or a muffin?