ML4H Tutorial 2
Medical Image Analysis

Alain Ryser
01.03.2022

Paper presentations

- The papers for **images** and **time-series** have been assigned!
- We will post the selected papers in our website one week in advance
  https://mds.inf.ethz.ch/teaching/261-5120-00l-spring-2022
- You should have a look at them before the presentation!
- Active participation during paper presentations is highly encouraged!
  - Turn on the camera if you join via Zoom (requirement)
  - Say your name and surname when you ask a question (we will keep a list with students that actively participated into the discussions)
- Full paper list is now released (check Moodle).
- Remember: first come first serve.

Lecture Recap

Medical imaging methods

- *Superpixel - Simple Linear Iterative Clustering (SLIC)*
- Markov Random Field
- Convolutional Neural Networks (CNN)

Lecture Recap

Medical imaging methods

- *Superpixel - Simple Linear Iterative Clustering (SLIC)*
- **Markov Random Field**
- Convolutional Neural Networks (CNN)

Markov Random Field

- MRF is a graphical model over an undirected graph (G=(V,E)) with distribution P(V) over vertices
- Fulfills Markov property:
  - For $\{x_i \in V\}$ and $N(x_i) = \{x_j \mid j \in N_i\}$, $P(x_i \mid x_{V-\{i\}}) = P(x_i \mid x_{N_i})$
  - ⇒Random Variables only locally dependent

- Define MRF for image segmentation:

$$P(\mathbf{x}) = \frac{1}{Z} exp(-E(\mathbf{x}))$$

$$E(\mathbf{x}) = \underbrace{\sum_{i \in V} \psi_i(x_i)}_{\text{Data Term}} + \underbrace{\sum_{i \in V, j \in N_i} \psi_{i,j}(x_i, x_j)}_{\text{Smoothness Term}}$$

Foreground / Background Estimation

$$\boxed{E(\mathbf{x}) \;=\; \sum_{i\in V} \psi_i(x_i) + \sum_{i \in V, j \in N_i} \psi_{i,j}(x_i, x_j)}$$

- $x_i = 0 \rightarrow$ i is in background
- $x_i = 1 \rightarrow$ i is in foreground

- Data term:

$$\begin{aligned}\psi_i(0) &= -\log P(x_i \in BG)\\ \psi_i(1) &= -\log P(x_i \in FG)\end{aligned}$$

- Smoothness term:

Pixel values

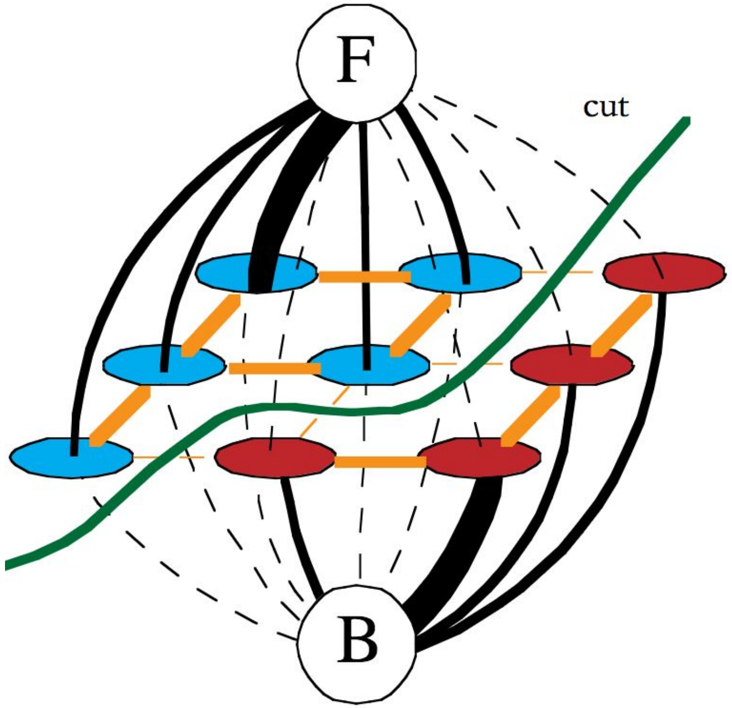
$$\begin{aligned}\psi_{ij}(x_i, x_i) &= K_{ij}\delta(x_i \neq x_j)\\ K_{ij} &= \lambda_1 + \lambda_2 \exp(-\beta(I_i - I_j)^2)\end{aligned}$$

- Intensity dependent smoothness

Foreground / Background Estimation

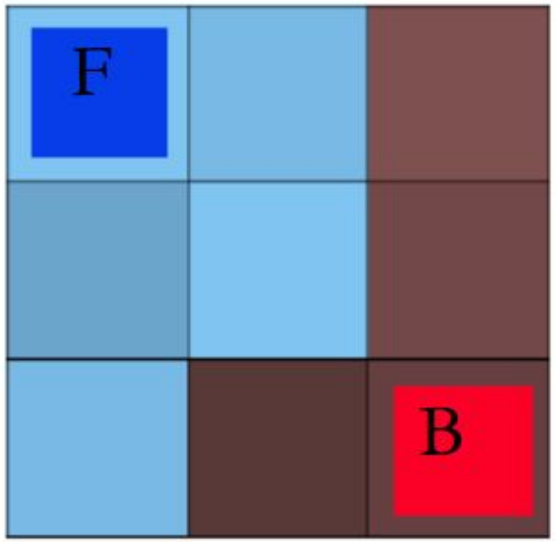
- We are looking for $x^* = \text{argmax}_x P(x) = \text{argmin}_x E(x)$

- This optimization problem can be solved by transforming the energy function into a min-cut problem and solve it.



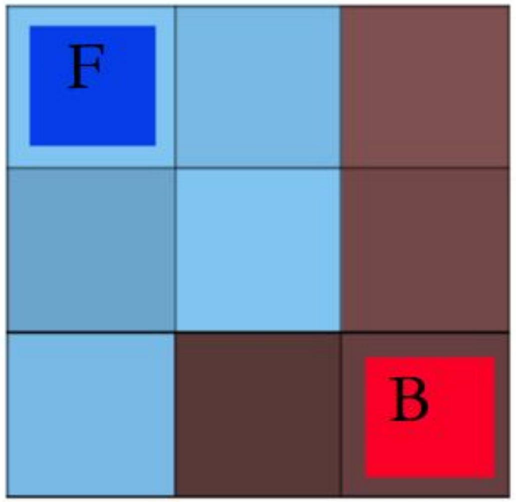
Graph Cuts for Optimal Boundary Detection

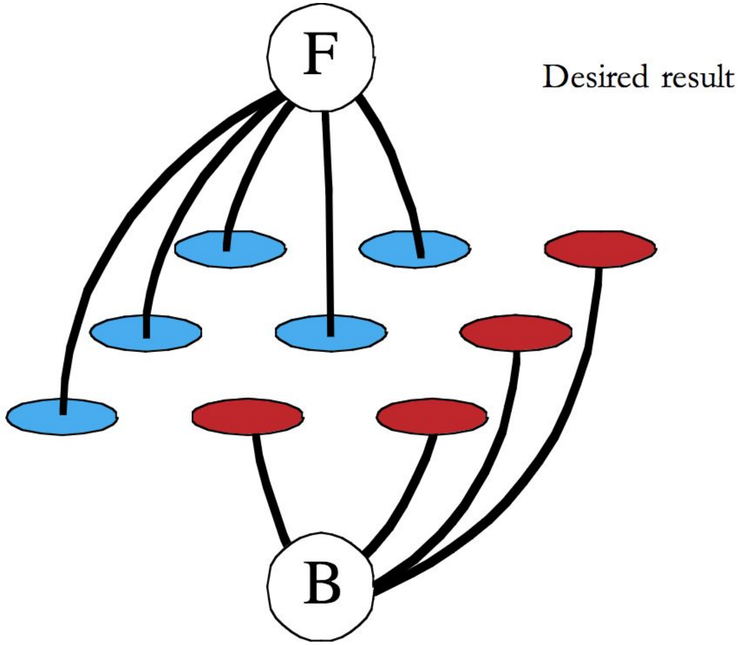
- Binary label: foreground vs. background
- User labels some pixels



Graph cut

- Each pixel = node
- Add two nodes F & B
- Labeling: link each pixel to either F or B



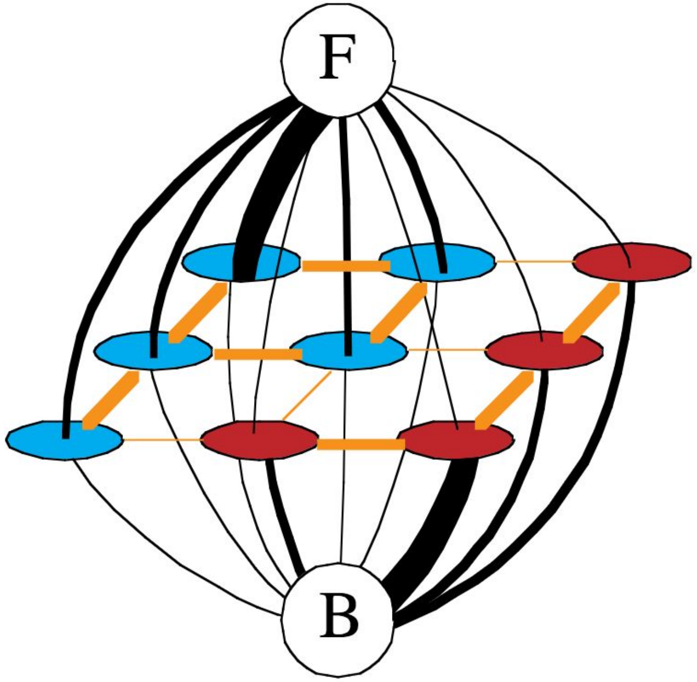


Configuring the edges

$$E(\mathbf{x}) = \sum_{i \in V} \psi_i(x_i) + \sum_{i \in V, j \in N_i} \psi_{i,j}(x_i, x_j)$$

$x^* = \mathrm{argmin}_x\, E(x)$

1. Add one edge between each pixel and both F & G
2. Set the weight of these edges to be: $-\psi_i(x_i)$
3. Add an edges between each neighbor pair
4. Set their weights to be: $\psi_{ij}(x_i, x_j)$



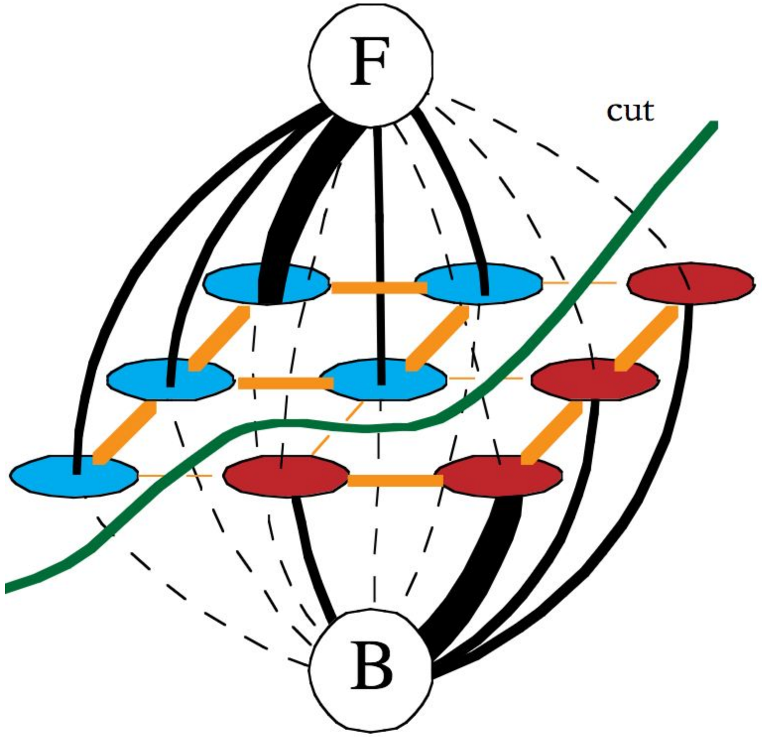
Configuring the edges

$$E(\mathbf{x}) = \sum_{i \in V} \psi_i(x_i) + \sum_{i \in V, j \in N_i} \psi_{i,j}(x_i, x_j)$$

$x^* = \text{argmin}_x\ E(x)$

1. Add one edge between each pixel and both F & G
2. Set the weight of these edges to be: $-\psi_i(x_i)$
3. Add an edges between each neighbor pair
4. Set their weights to be: $\psi_{ij}(x_i, x_j)$

- Minimum of E(x) corresponds to sum of weights of removed edges in graph min-cut
- Vertices on one side of cut are FG, other side BG



Lecture Recap

Medical imaging methods

- *Superpixel - Simple Linear Iterative Clustering (SLIC)*
- Markov Random Field
- **Convolutional Neural Networks (CNN)**

Segmentation with Deep Learning

- *MRFs* assumed prior knowledge about data distribution of foreground/background distributions of individual and neighboring pixels

- Deep Learning makes only limited assumptions about distribution but assumes set of samples, i.e. training data, of the distribution is available at our disposal

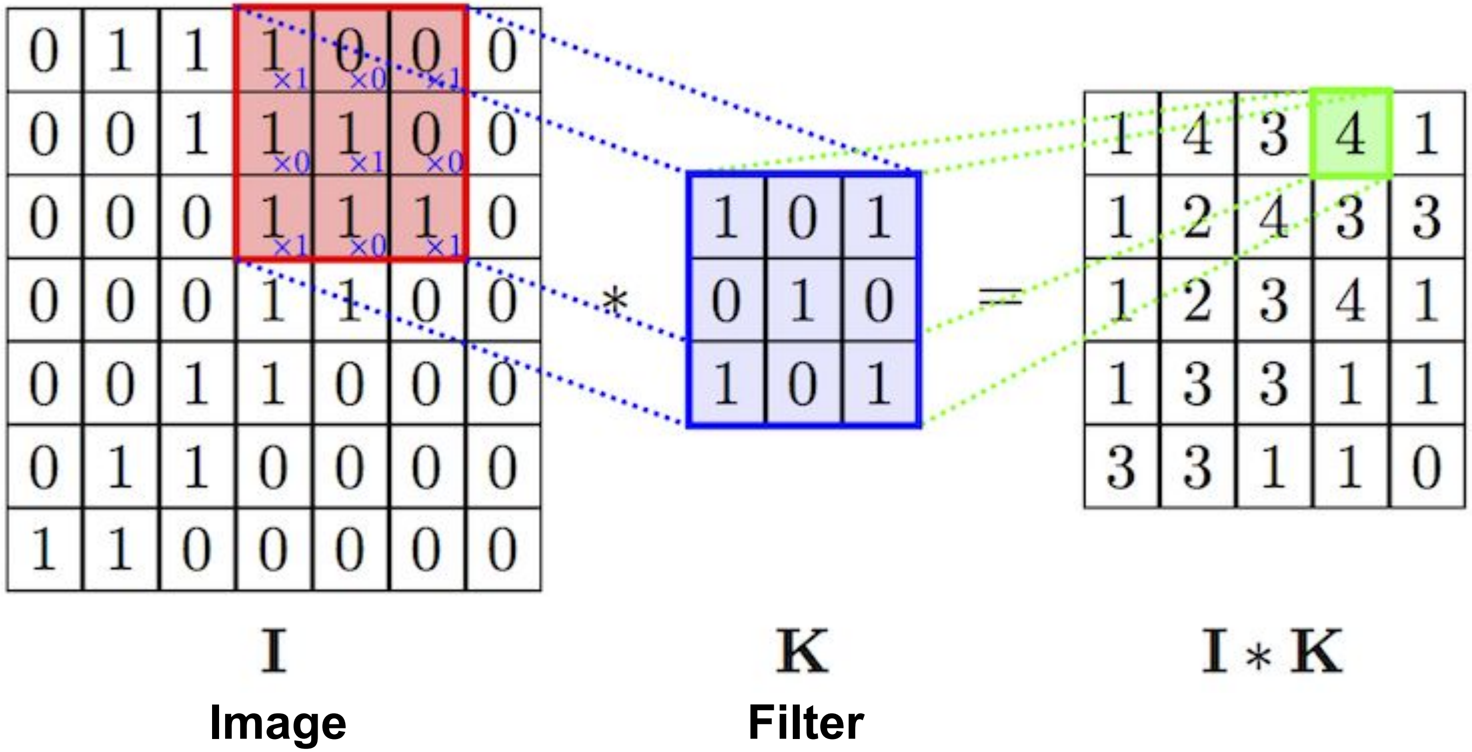
- Learn segmentation of specific sample distributions in data driven way

- Prominent segmentation models: Convolutional Neural Networks (CNNs)

How do convolutions work?

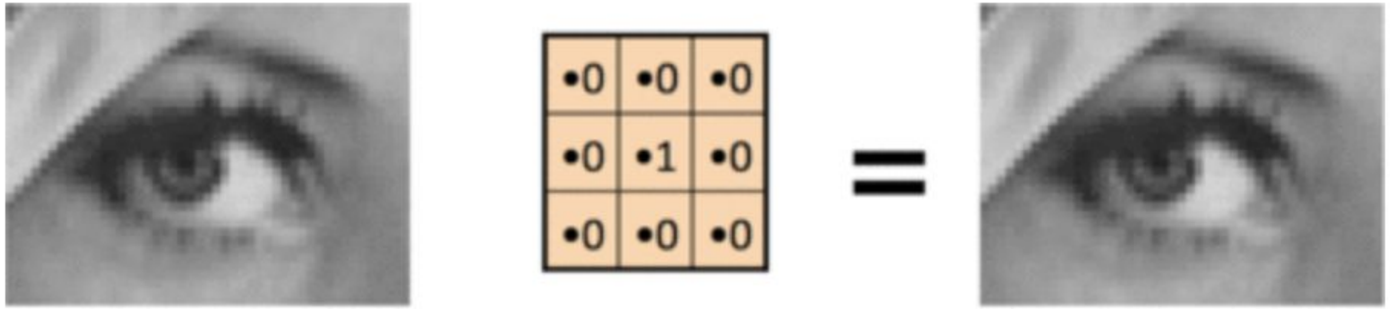
Convolutional layers

- consist of a set of small-size filters
- extract local features from the input layer, or outputs from the previous layer



Petar Veličković, Cambridge Spark

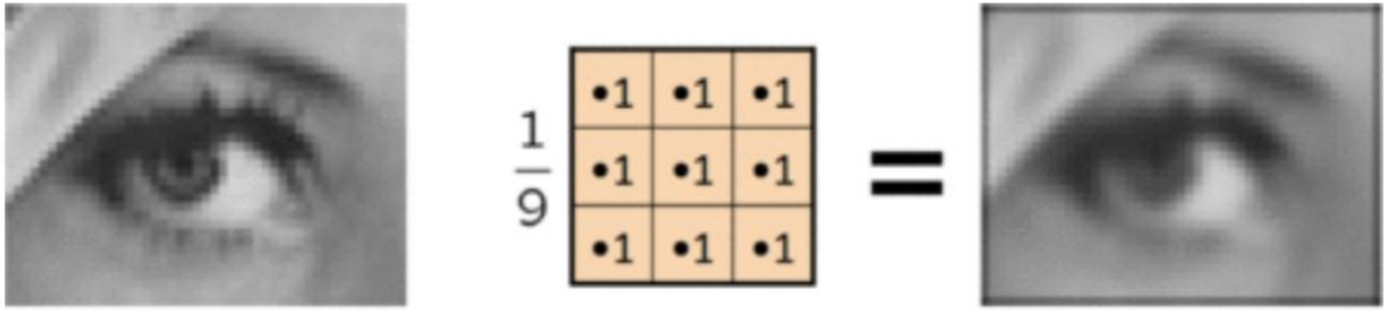
Examples - Identity



Original

Filtered
(no change)

Examples - Smoothing (Box Filter)



Original

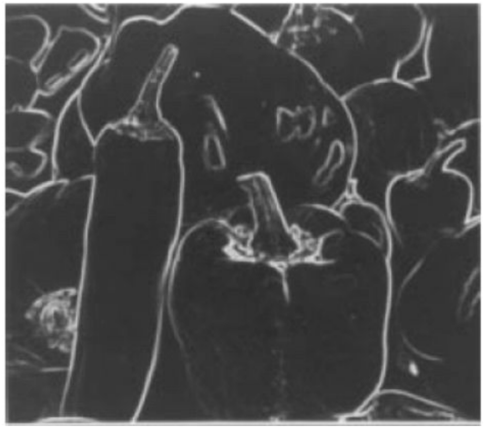
Blur (with a box filter)

Examples - Smoothing (Gaussian Filter)

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

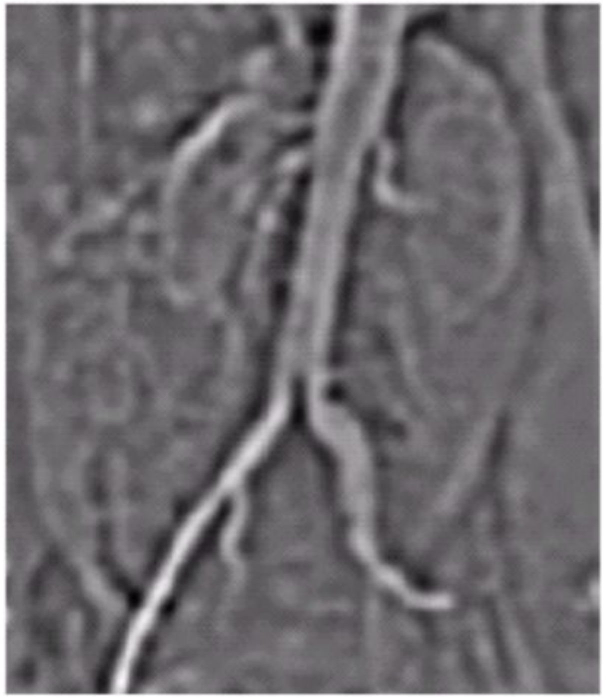
Examples - (Vertical) Edge detection

$$\frac{1}{3}\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

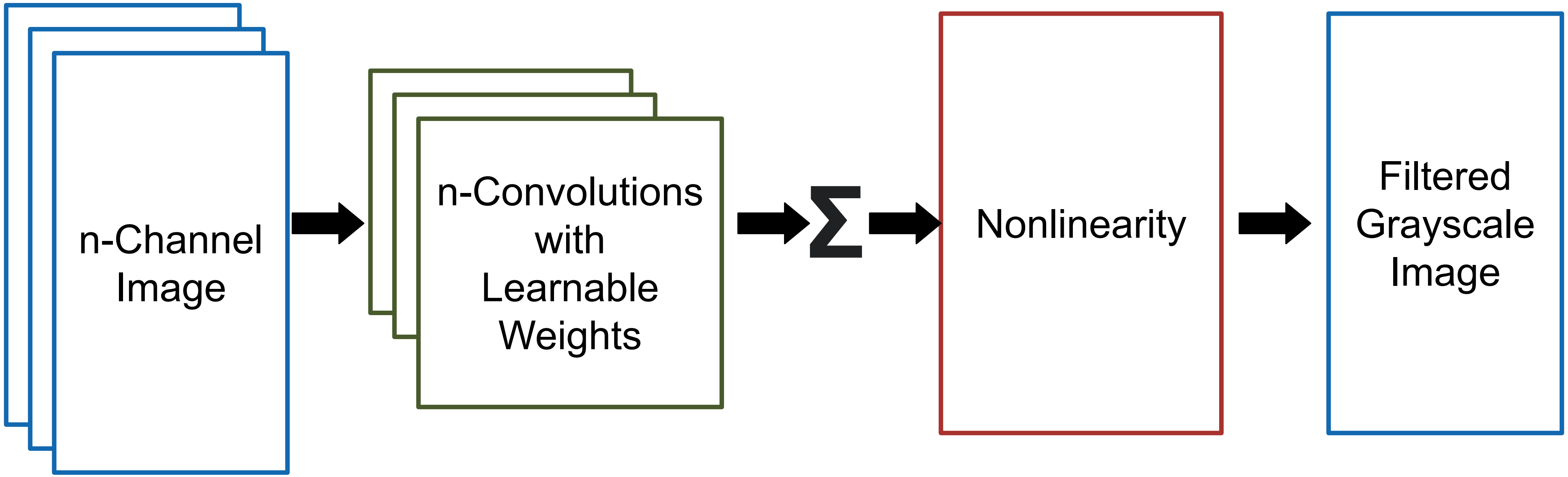


Examples - Smoothing & Edge Detection (LoG - Laplacian of Gaussians)

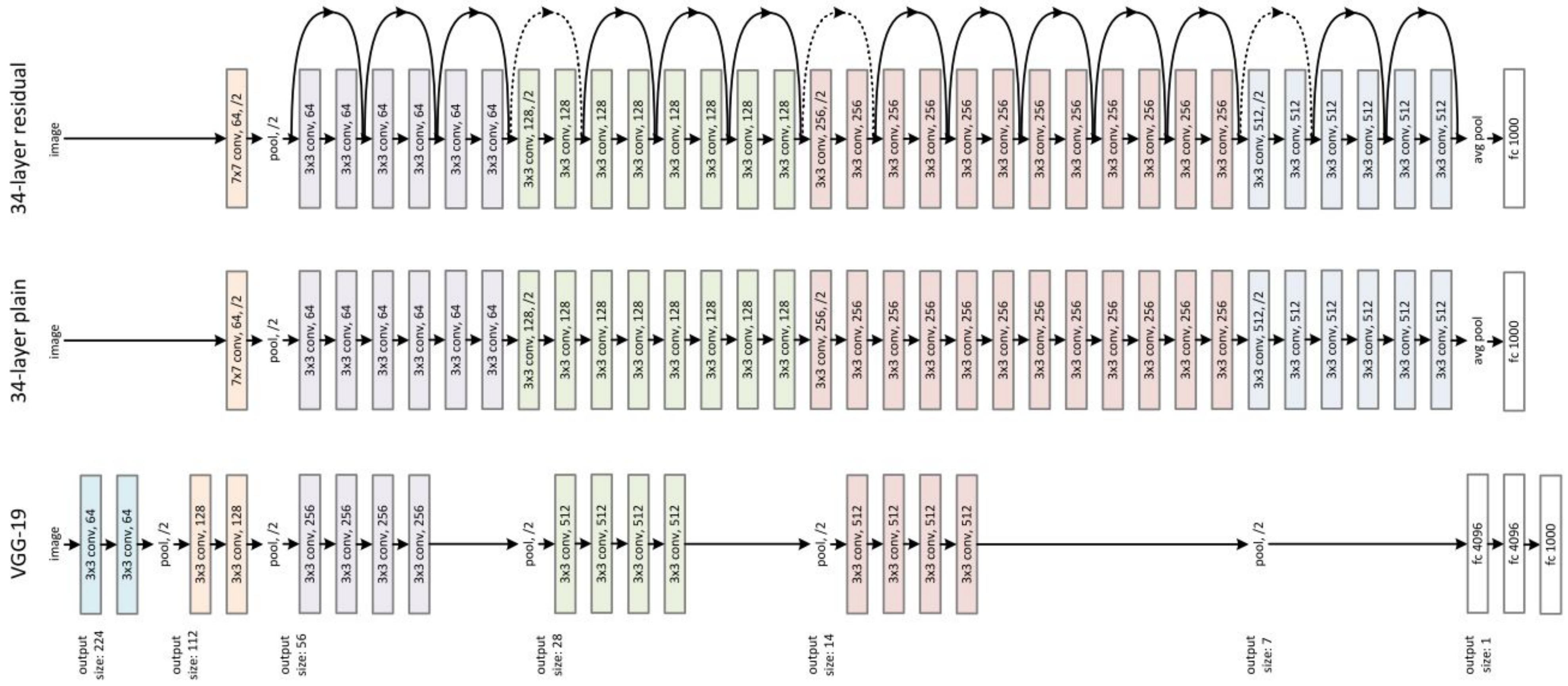
0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0



Standard Convolutional Layers



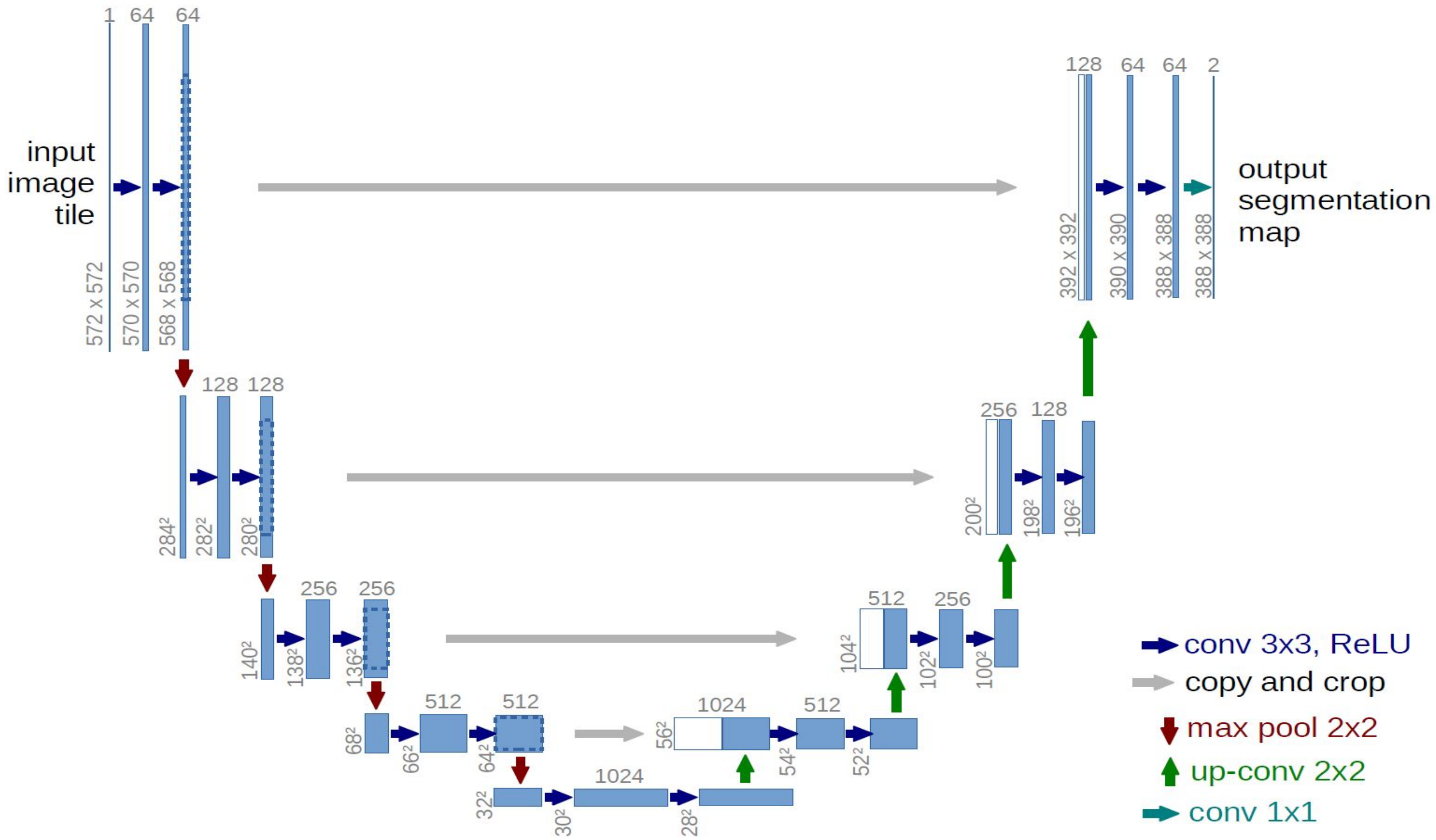
Modern ConvNets
Horizontal and Vertical Stacking of Convolutional layers



ConvNets - De facto standard for many imaging tasks

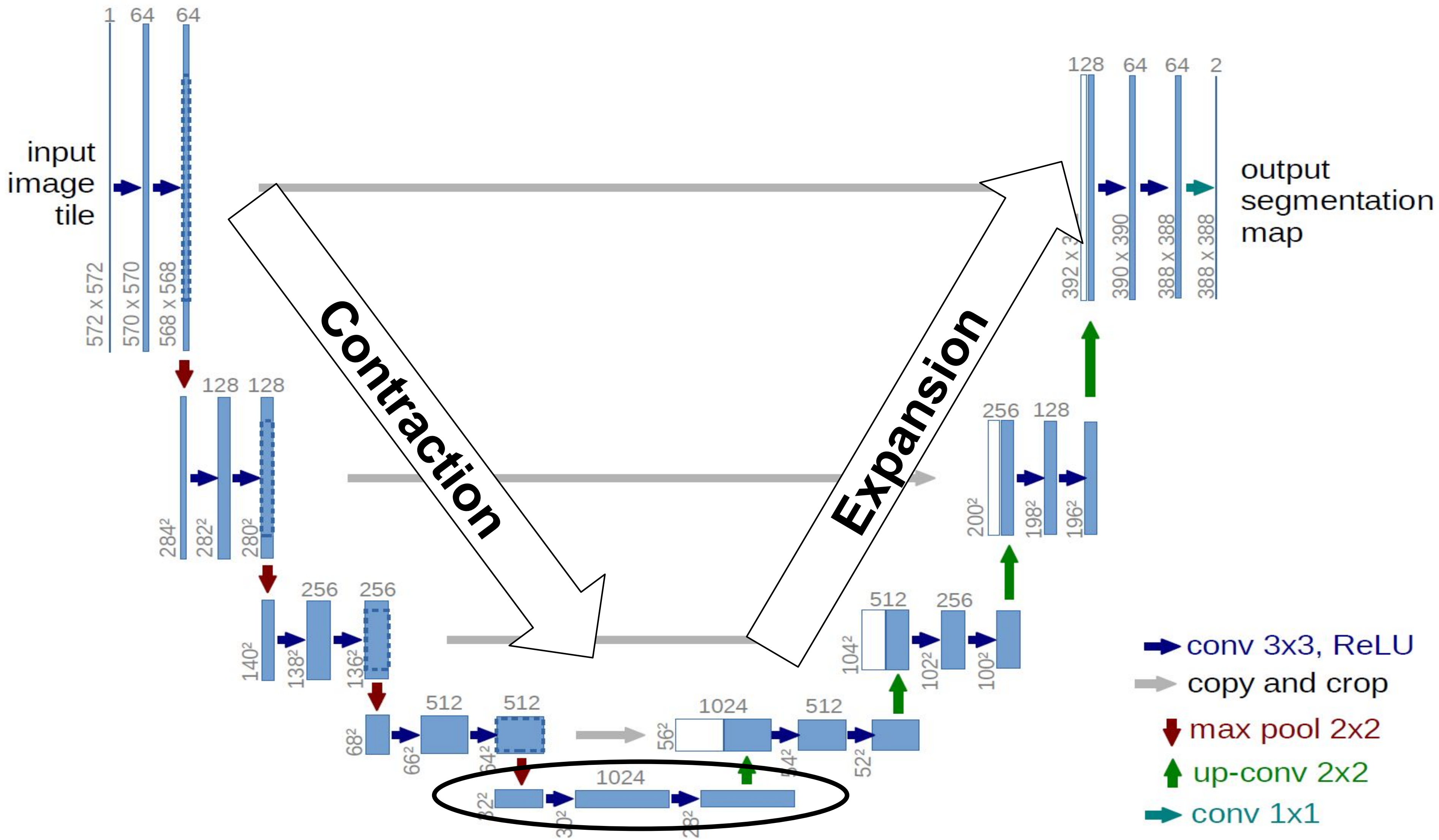
- Classification
- Object localization
- Synthetic image generation
- Anomaly Detection
- **Segmentation**
- …

U-Net, CNN architecture for image segmentation



https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/

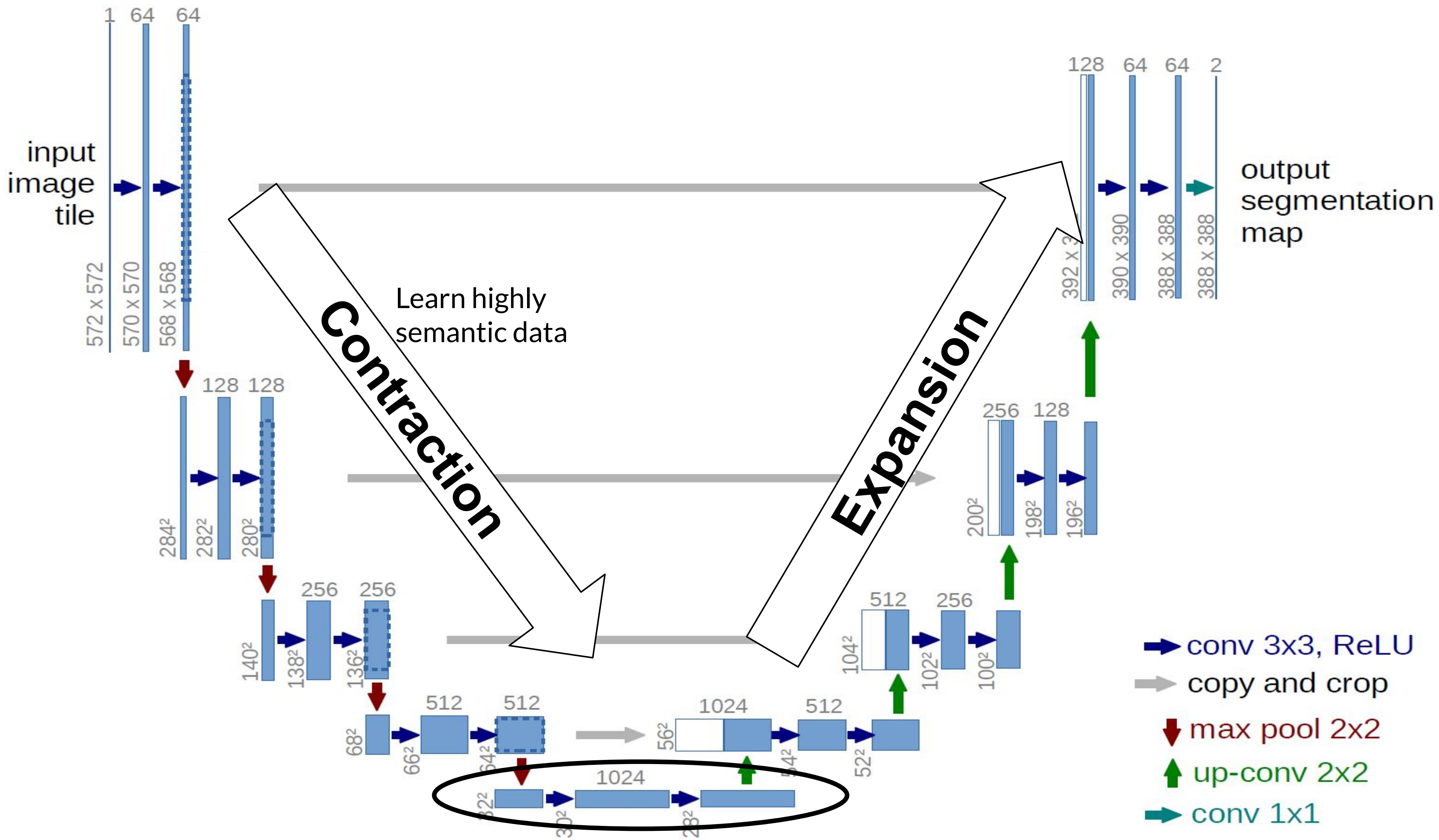
Learn highly semantic data by introducing data bottleneck



https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/

Bottleneck

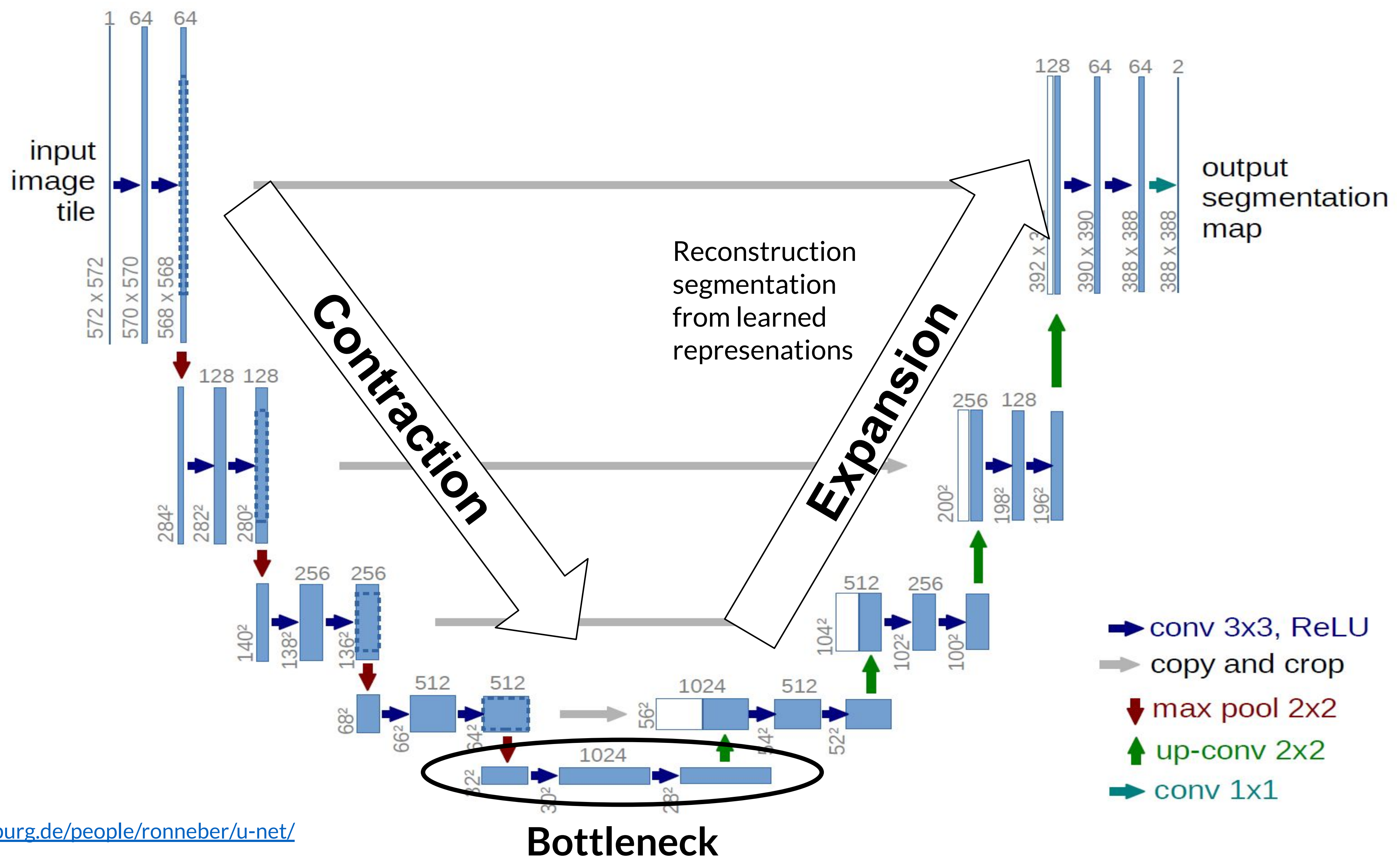
Learn highly semantic data by introducing data bottleneck



Bottleneck

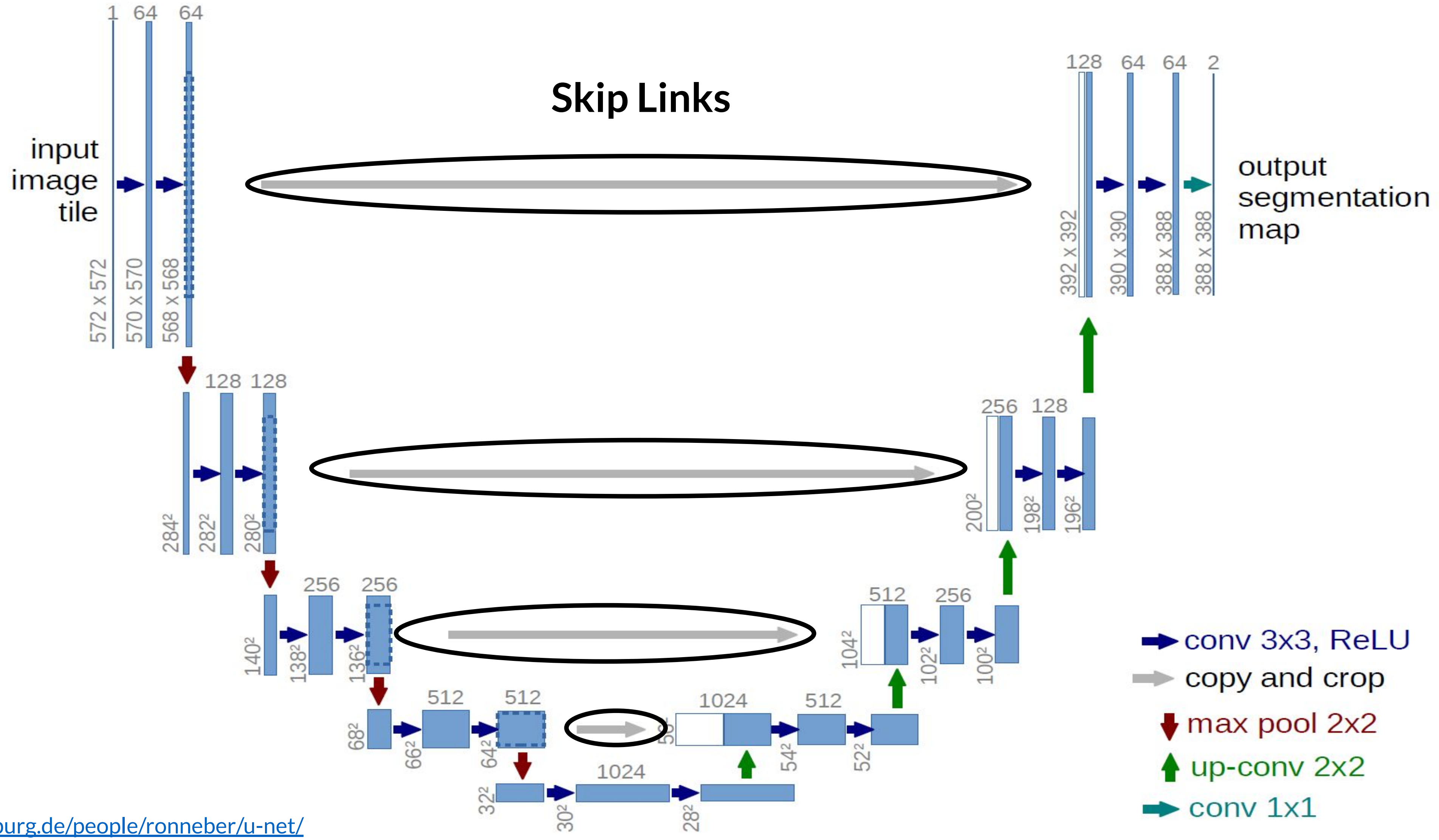
https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/

Learn highly semantic data by introducing data bottleneck



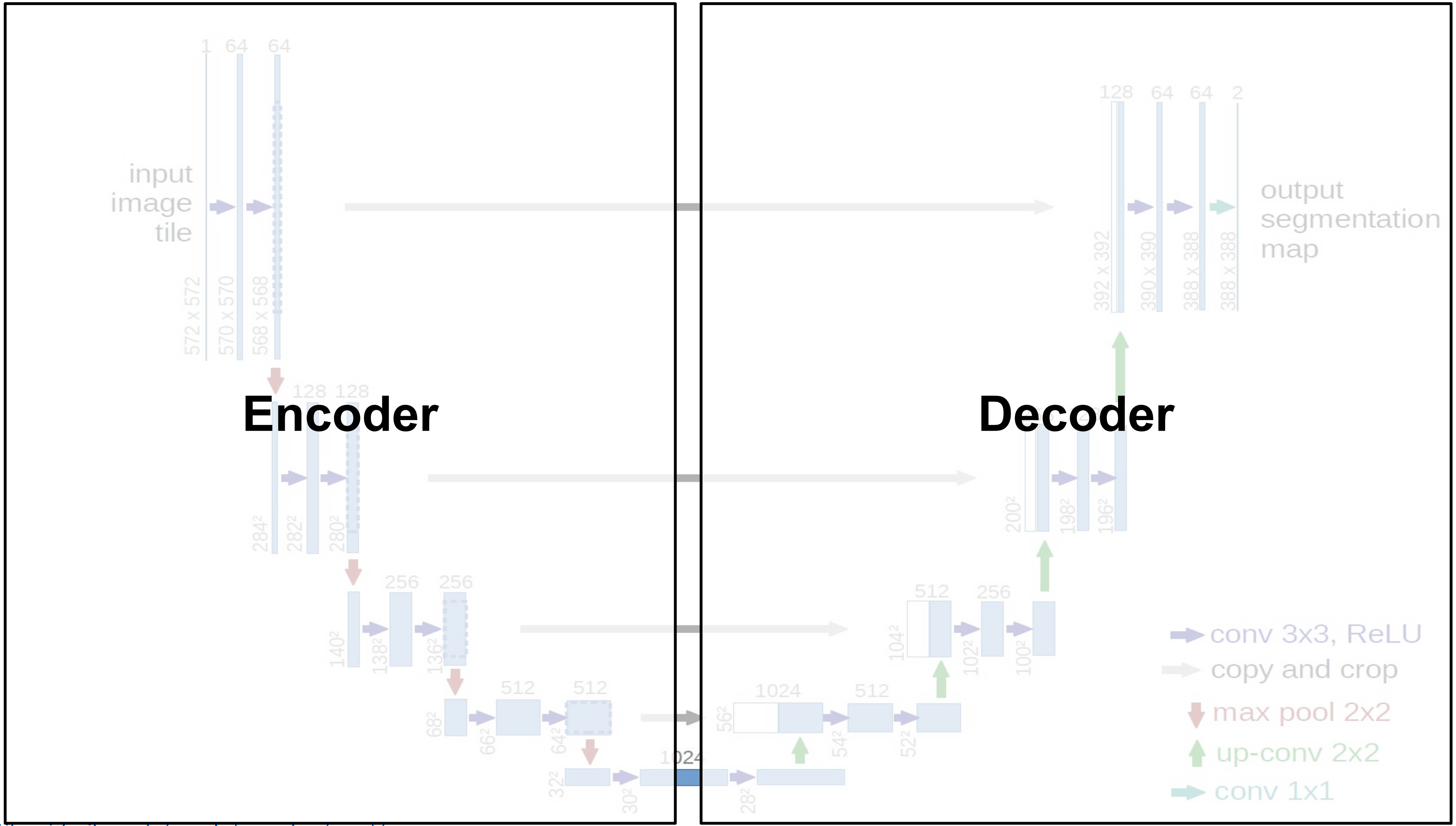
https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/

High resolution segmentation maps by passing high spatial resolution features decoding path



https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/

Symmetric architecture



https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/

Common Problem in Medical Applications - Overfitting

- Acquiring Medical Data is hard
- Need domain experts to collect data and label data => Expensive and time consuming
- Patients possibly don't want to share their medical records
- Some diseases are quite rare => Not many samples in datasets
- Often need big models to capture complex feature relations within medical data

⇒Big Models and few data lead to danger of overfitting

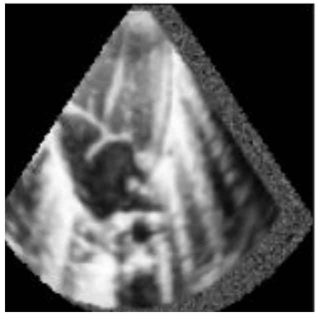
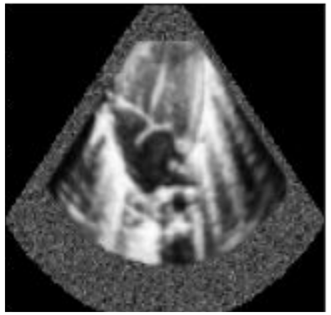
Regularization techniques need to be employed

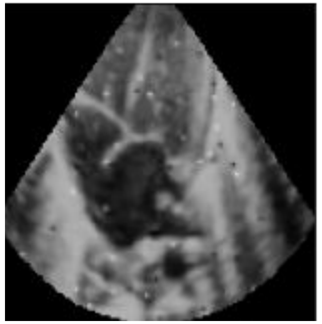
- **Data Augmentation**
- L1/L2 regularization
- Early stopping
- Dropout
- Batchnormalization

Data Augmentation - Transform samples before forward pass

- Rotation
- Translation
- Scaling
- Gaussian Noise
- Salt & Pepper Noise
- Brightness adjustments
- Smoothing
- Sharpening
- Color Jittering
- …



Regularization techniques for NN overfitting problem

Too much parameters, not enough training data

- Data Augmentation
- **L1/L2 regularization**
- Early stopping
- Dropout
- Batchnormalization

$$\mathcal{J} = \mathcal{L} + \lambda \|\mathbf{W}\|_2^2$$

$$\mathcal{J} = \mathcal{L} + \lambda \|\mathbf{W}\|_1$$

Objective function

Loss term

Regularization term

Regularization techniques for NN overfitting problem

Too much parameters, not enough training data

- Data Augmentation
- L1/L2 regularization
- **Early stopping**
- Dropout
- Batchnormalization

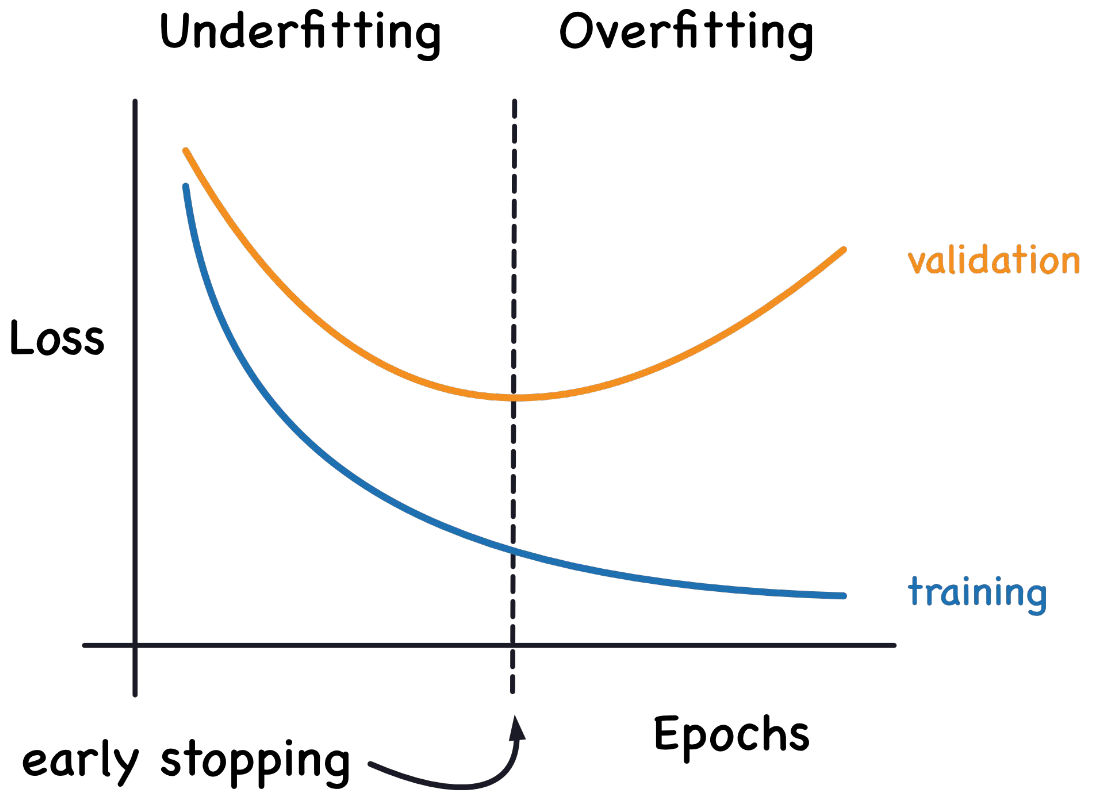
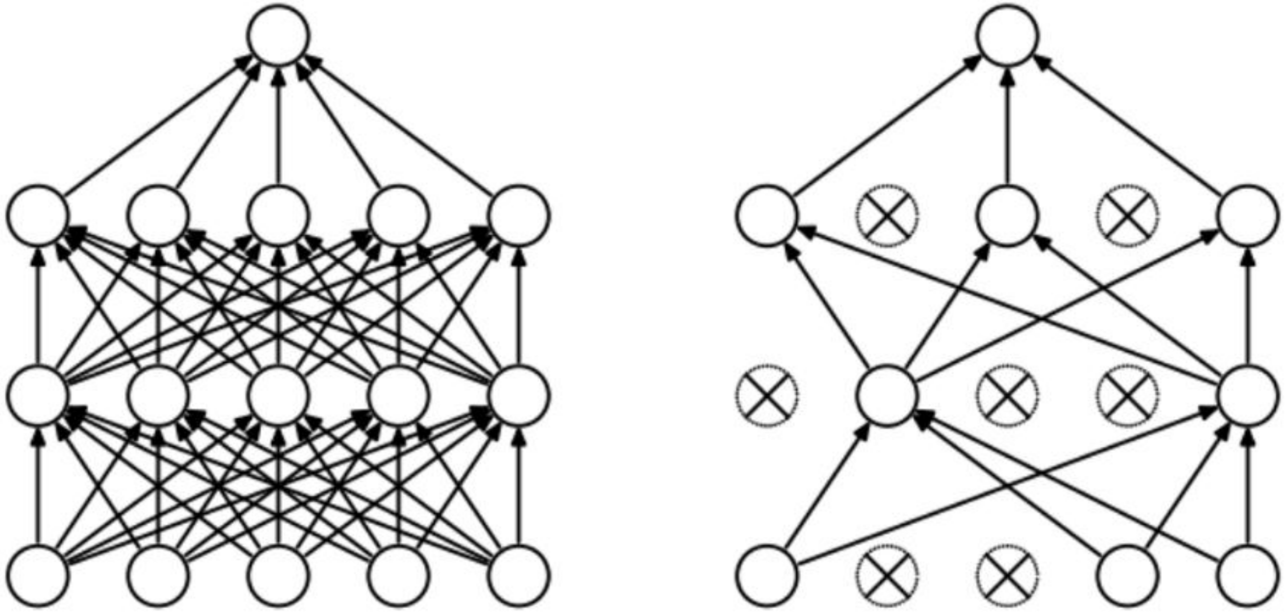


Image source

Regularization techniques for NN overfitting problem

Too much parameters, not enough training data

- Data Augmentation
- L1/L2 regularization
- Early stopping
- **Dropout**
- Batchnormalization



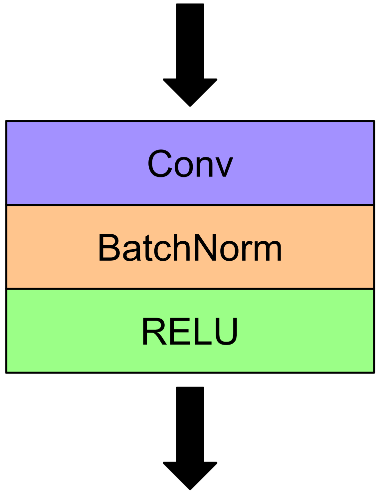
(a) Standard Neural Net (b) After applying dropout.

image source

Regularization techniques for NN overfitting problem

Too much parameters, not enough training data

- Data Augmentation
- L1/L2 regularization
- Early stopping
- Dropout
- **Batchnormalization**



Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

ML4H Tutorial 2

Medical Image Analysis

Alain Ryser

01.03.2022

SLIC - Basic Idea

- Find visually homogeneous superpixels with *k-means* clustering

SLIC - Basic Idea

- Find visually homogeneous superpixels with *k-means* clustering

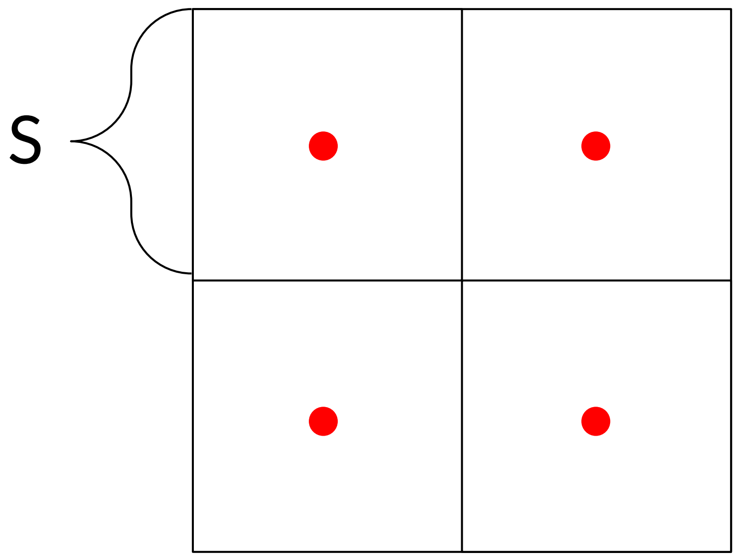
- Associate each pixel to the feature vector:

$$\Psi(x, y) = [\lambda x, \lambda y, I(x, y)]$$

where I(x,y) corresponds to the pixel intensity (e.g. RGB) of the image at location (x,y). The λ coefficient serves as a regularizer to balance spatial and intensity based similarities within clusters.

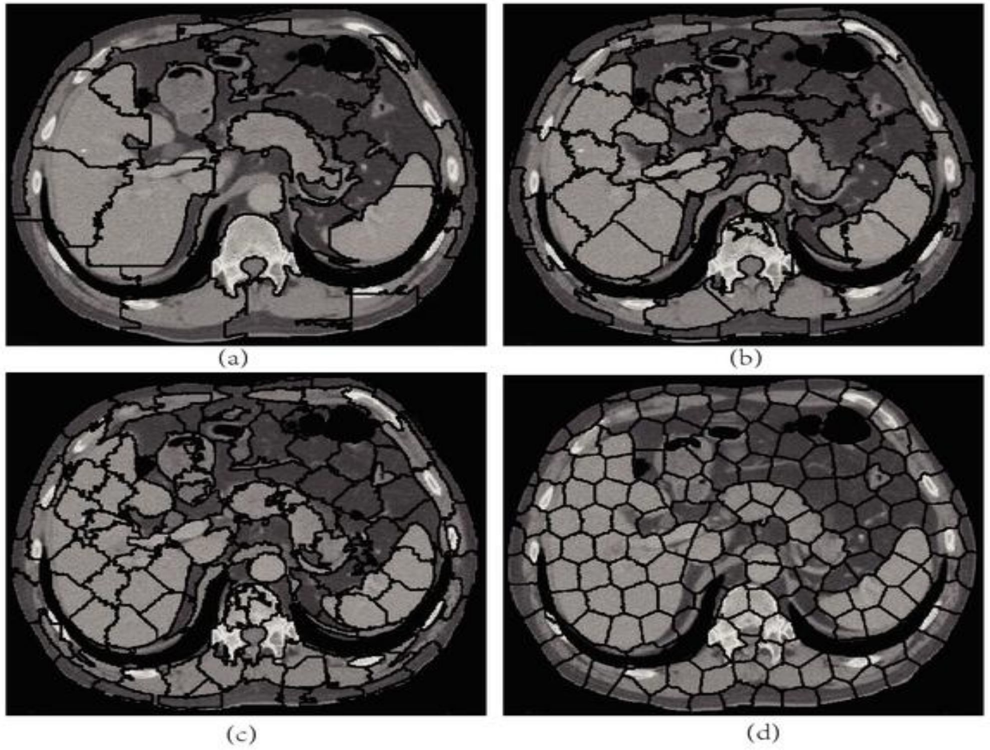
SLIC - Basic Idea

- To initialize *k-means*, define grid size S based on *k* and initialize cluster centers at grid center



SLIC - Basic Idea

- Run *k-means* on this initialization, resulting clusters correspond to superpixels that can later be used in various algorithms, e.g. Markov Random Fields



Source: L. Zhang et. al. An improved method for pancreas segmentation using SLIC and interactive region merging