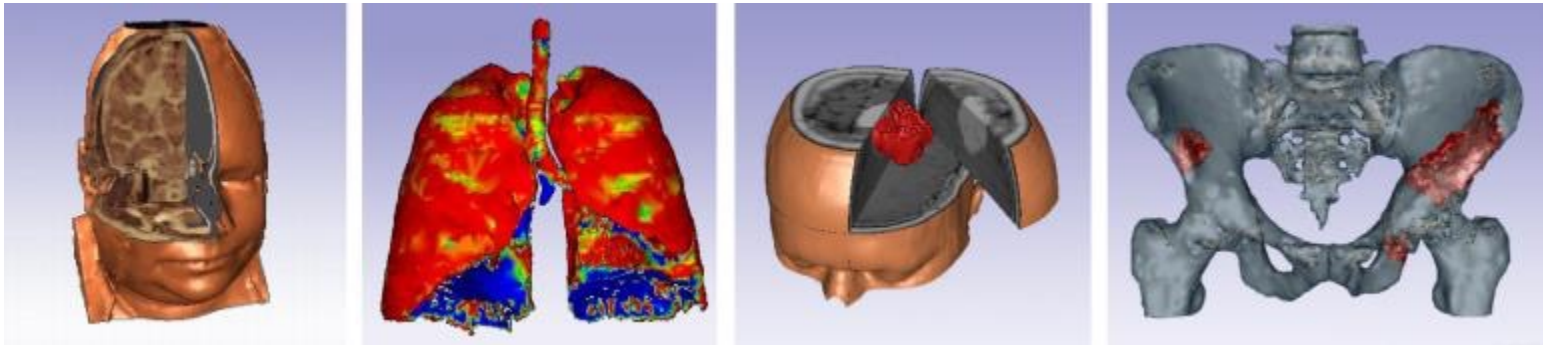


Modell- und KI-basierte Bildverarbeitung in der Medizin



KI-basierte Segmentierung in der medizinischen Bildverarbeitung

Dr.-Ing. Marian Himstedt
Institut für Medizinische Informatik
Universität zu Lübeck

Begriffseinordnung

Klassifikation

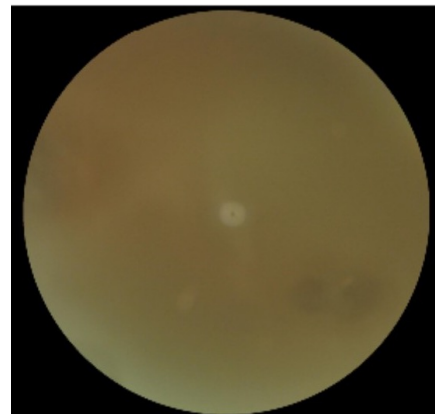
Ordnet dem Bild bzw. Bildobjekten eine von n möglichen Klassen zu.



Fahrradfahrer

Medizinisches Beispiel

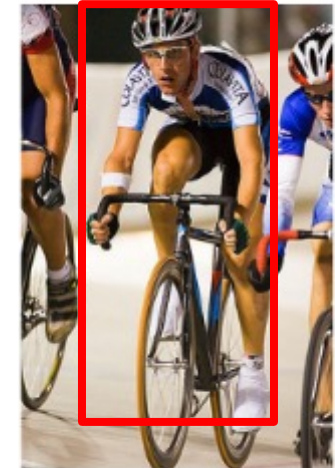
Klassifikation einer Augen-
erkrankung anhand eines
Augenhintergrundbildes



Grauer Star

Klassifikation + Lokalisation

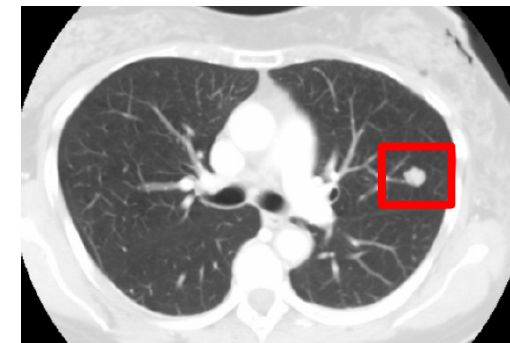
Zusätzlich zur Klassen-
zuordnung wird hier die
Lokalisation des Objektes
grob durch Bounding Boxes
beschrieben.



Fahrradfahrer

Medizinisches Beispiel

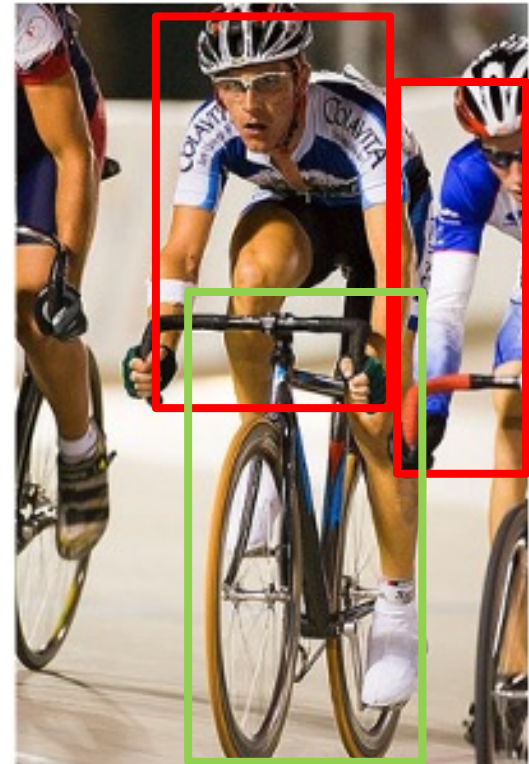
Klassifikation und
Lokalisation von
Tumorknoten in CT Bilddaten



Knoten

Objektdetektion

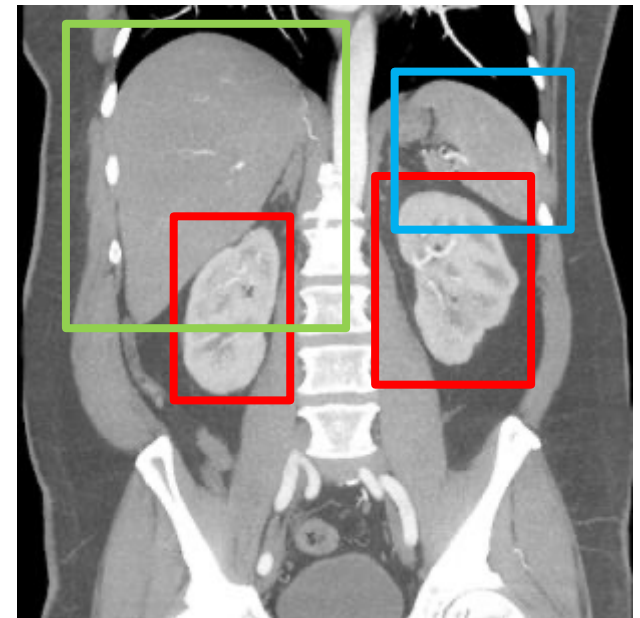
Detektiert mehrere Objekte
im Bild und klassifiziert diese.



 Person  Fahrrad

Objektdetektion

Detektion und
Klassifizierung mehrerer
Organe in CT-Bilddaten



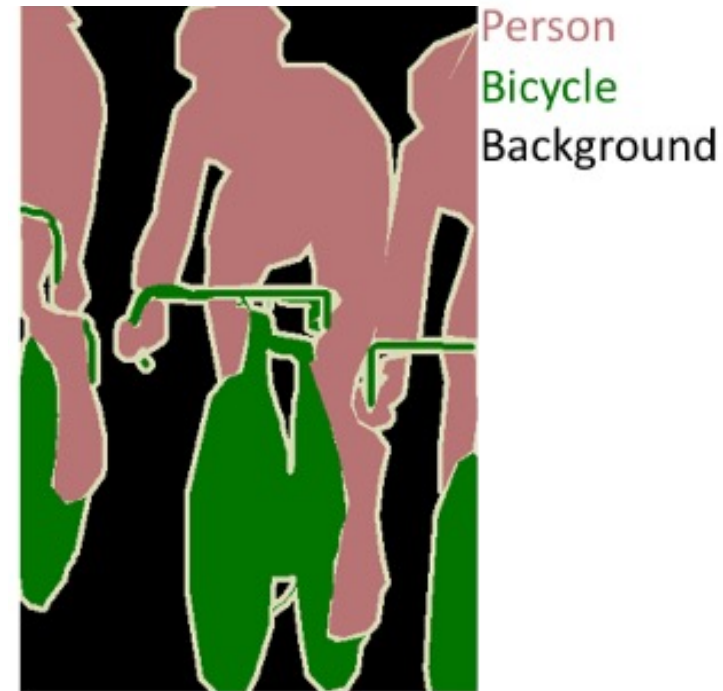
 Niere

 Leber

 Milz

Semantische Segmentierung

Ordnet jedem Pixel eine Klasse zu.
→ **Pixelweise Klassifikation**



- Oft kurz als Segmentierung bezeichnet
- Unterscheidet nicht zwischen Instanzen einer Klasse (z.B. Person1, Person2,...)
- → Semantische Instanz-Segmentierung

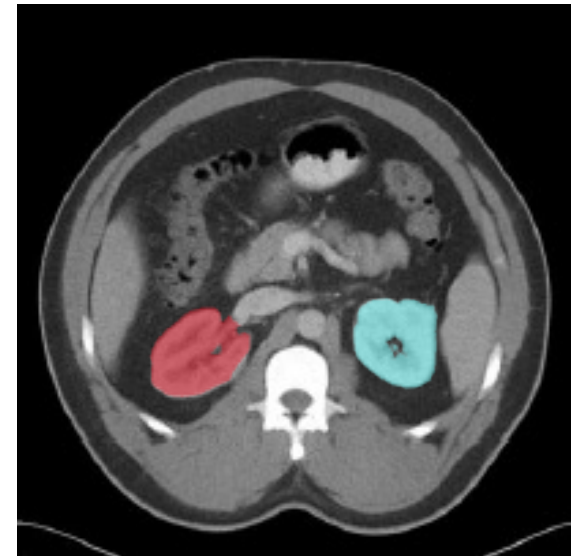
Semantische Segmentierung

Segmentierung der Nieren
in axialen CT Bilddaten



Semantische Instanzsegmentierung

Segmentierung der rechten
und linken Nieren in axialen
CT Bilddaten

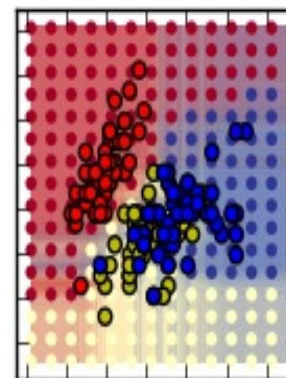


 Linke Niere

 Rechte Niere

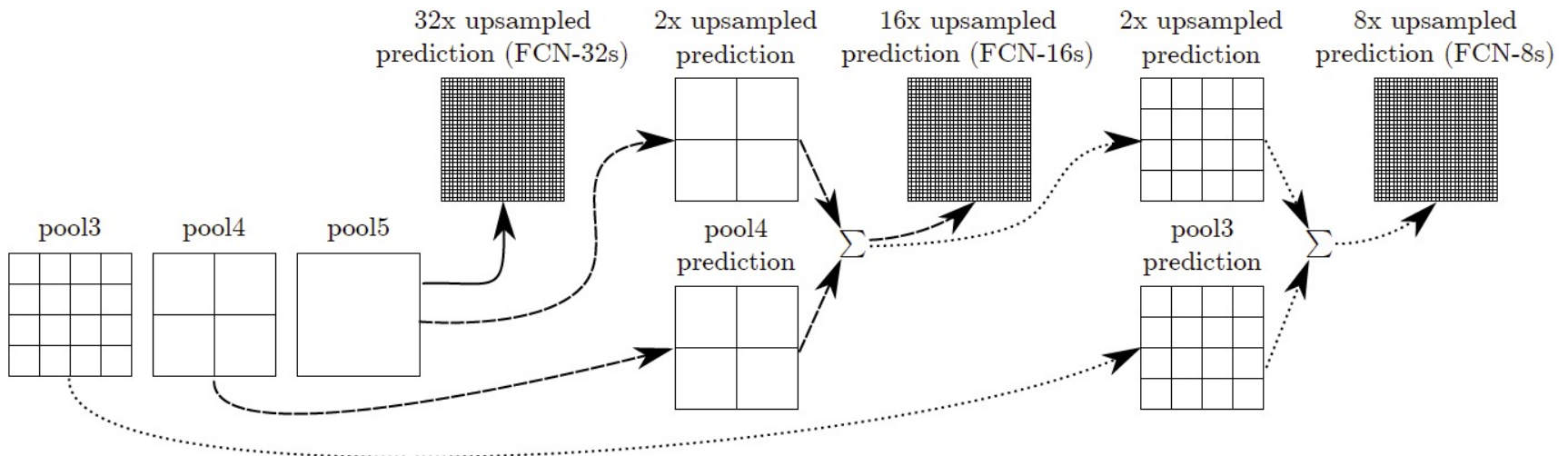
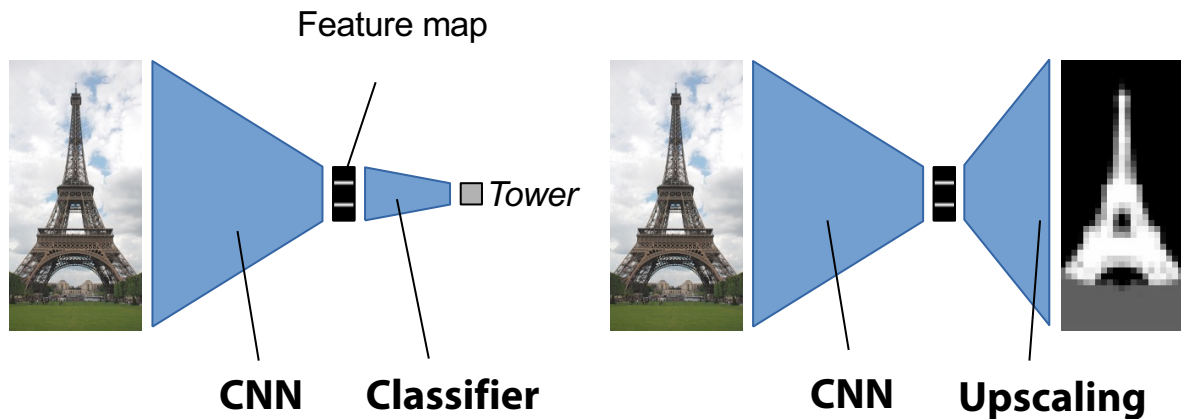
Klassifikatoren in der Medizinischen Bildverarbeitung

- Klassifikatoren in der Bildobjekterkennung
 - Bildobjekte wie z. B. Organe, Tumoren etc. werden durch Merkmalsvektoren charakterisiert.
 - Die Klassifikation der Bildobjekte anhand ihrer Merkmalsvektoren ermöglicht die automatische Erkennung und Benennung der Bildobjekte
- Klassifikatoren zur Bildobjektsegmentierung
 - Pixel werden durch Merkmalsvektoren charakterisiert.
 - Multispektrale Bilddaten
 - Durch pixelweise Klassifikation der Pixelvektoren erhält man eine Segmentierung

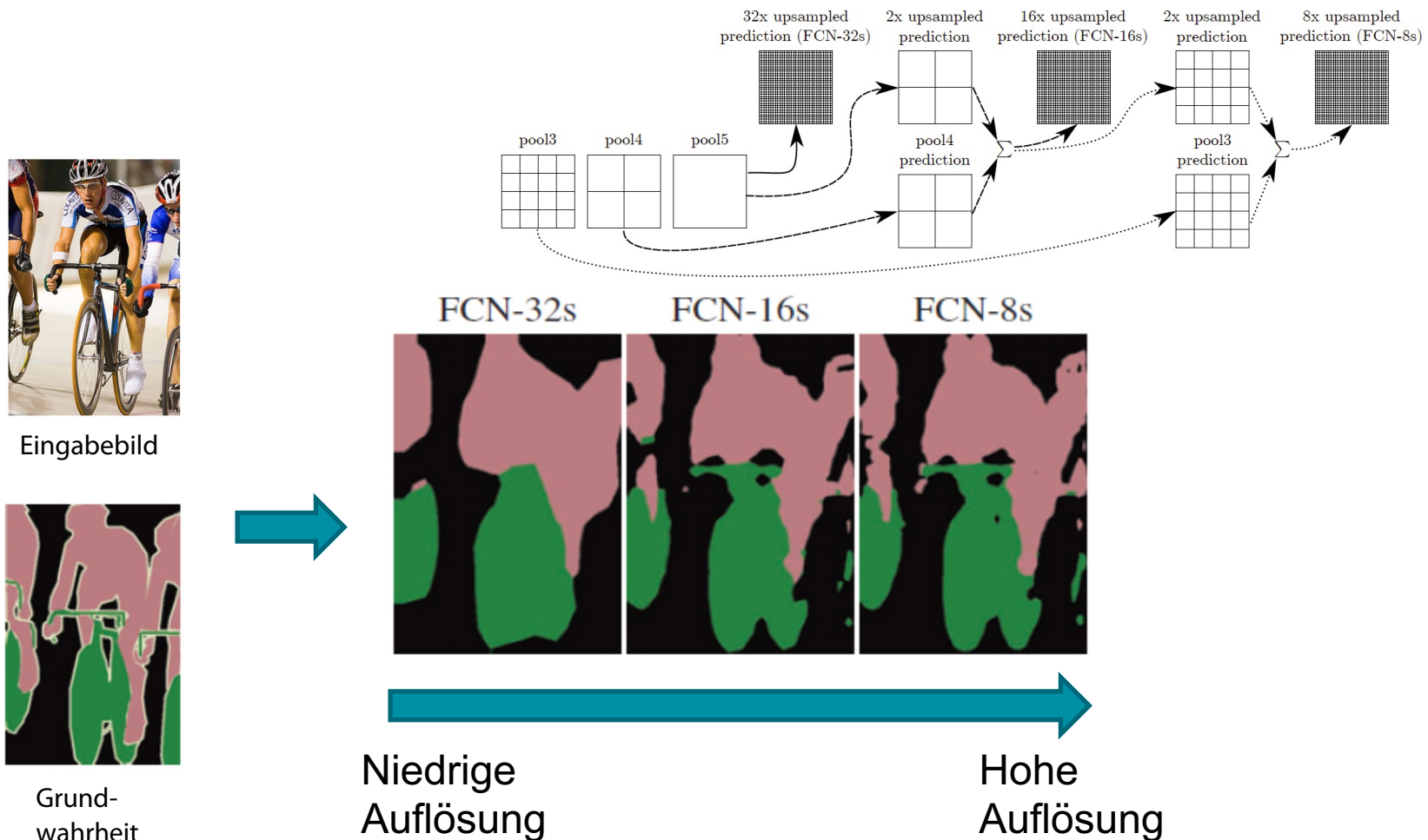


2-dimensionale Merkmalsräume
mit Klassenbereichen

Fully Convolutional Networks



Fully Convolutional Networks



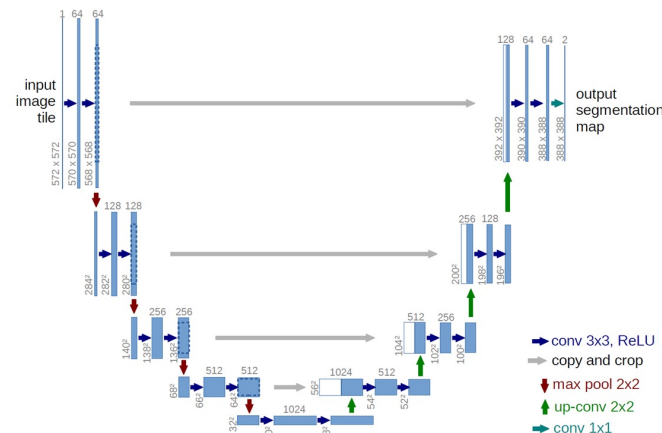
Geht es noch besser?



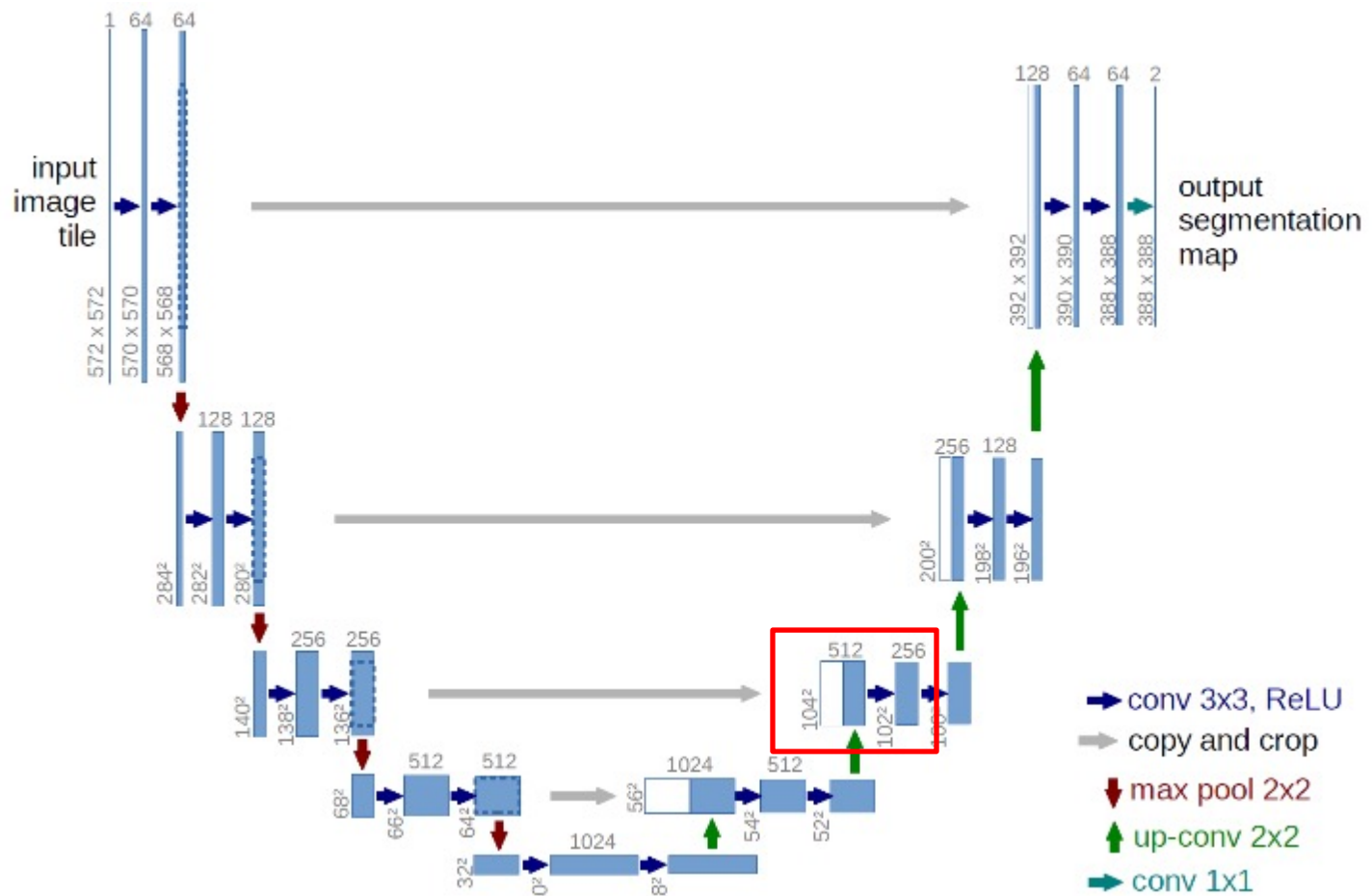
U-Nets

Einführung

- Ziel: Semantische Segmentierung medizinischer Bilder
 - entwickelt von O. Ronneberger et al., erstmals vorgestellt auf der MICCAI 2015
 - Olaf Ronneberger, Philipp Fischer, Thomas Brox, Medical Image Computing and Computer-Assisted Intervention (MICCAI), Springer, LNCS, Vol. 9351: 234 - 241, 2015
 - Meistzitiertester MICCAI-Beitrag aller Zeiten
- Überwachte Methode des Maschinellen Lernens
 - Von Experten segmentierte (gelabelte) Bilddaten werden benötigt.
- Name nach der U-förmigen grafischen Darstellung des Netzes



U-Net Architektur

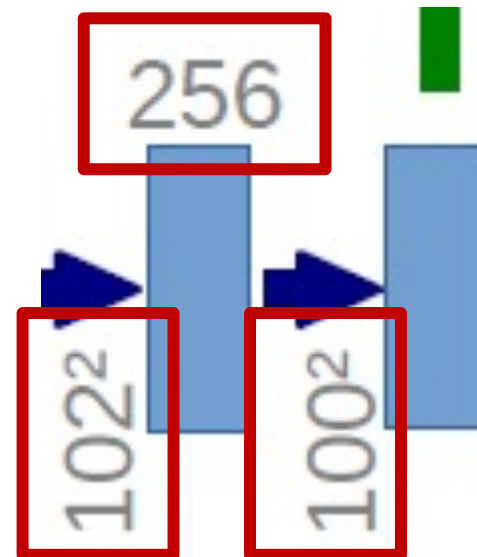


U-Net Kennzahlen

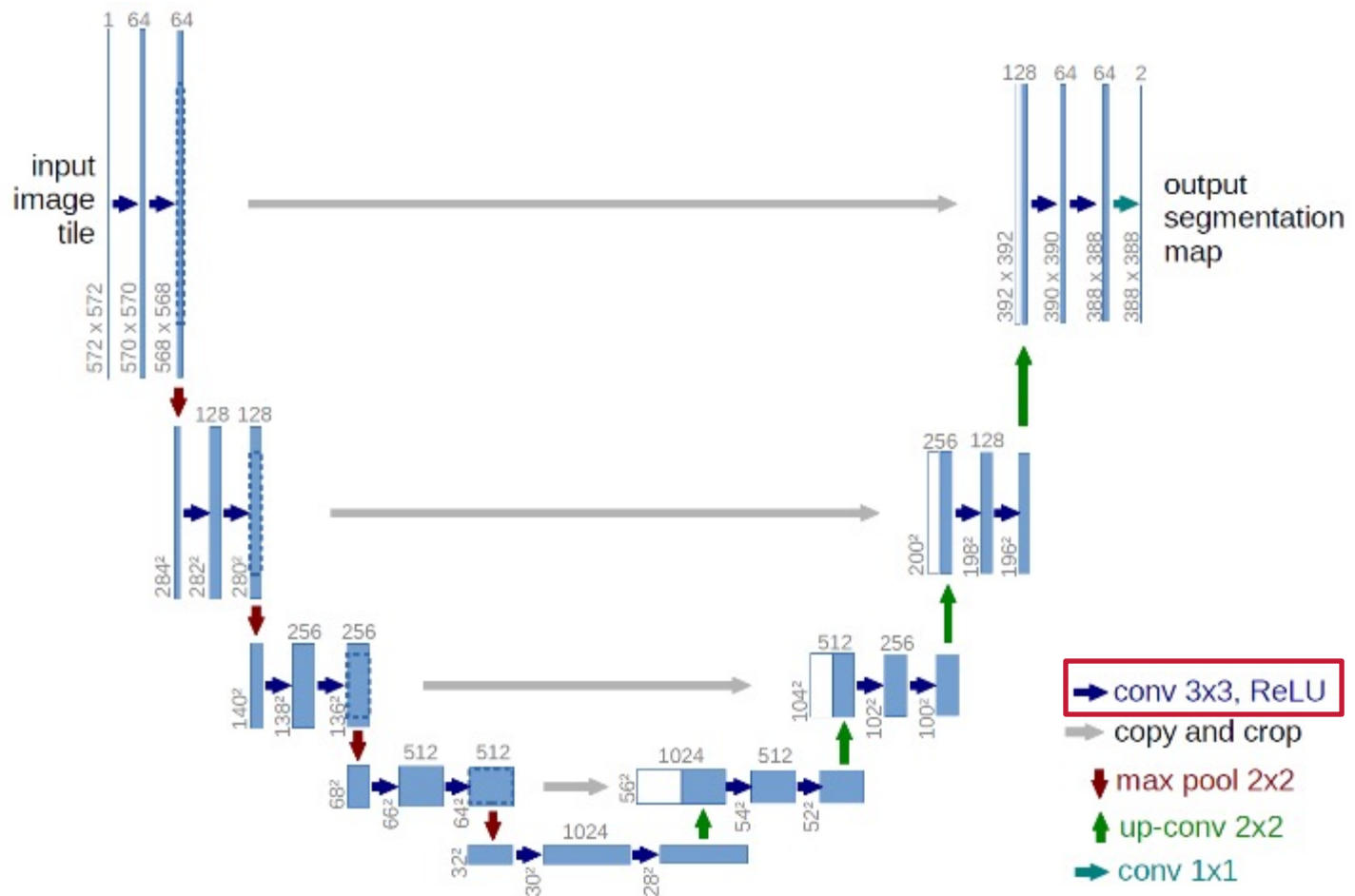
Anzahl der Channels bzw.
Feature Maps, hier: 256

Bildgröße, hier: 102x102

Bildgröße nach Faltung
mit 3x3 Kernel ohne
Padding, hier: 100x100



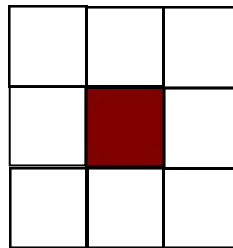
U-Net Architektur



Faltung (convolution)

$$f_{out}(x, y) = \sum_{i=-m}^m \sum_{j=-m}^m f_{in}(x + i, y + j) \cdot w(i, j)$$

- Größe des Kernels w
 - Beispiel 3x3:



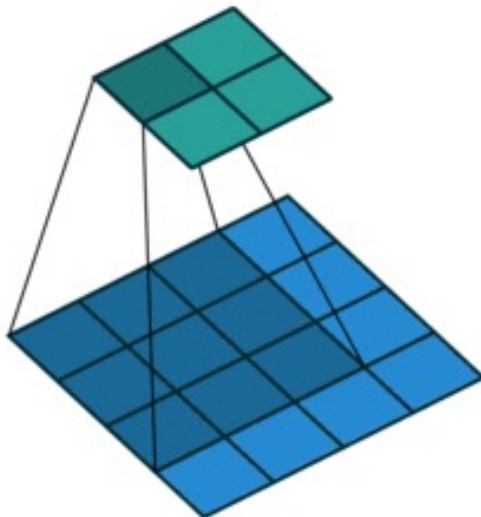
conv 3x3

- Erweiterungen
 - Stride (Schrittweitenvariation zwischen den Faltungen eines Bildbereichs)
 - Dilated Convolution (Abstände zwischen Positionen einer Faltung)
 - Padding (Randbehandlung)

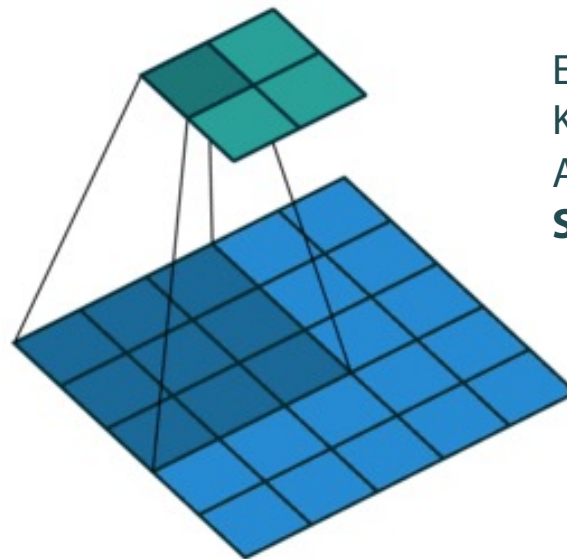
Stride bei der Faltung

- Der Stride (Schrittweite) gibt die Distanz zwischen zwei aufeinander folgenden Positionen aufeinanderfolgender Faltungen an, an denen der Kernel angewandt wird.
- Er beschreibt die Schrittweite des Faltungskernels bei einer Faltung.
- Der Stride beeinflusst die Größe der resultierenden Matrix, die durch die Anzahl der durchgeführten Faltungen festgelegt wird.
- Beispiele:

Eingabe: 4x4
Kernel: 3x3
Ausgabe: 2x2
Stride: 1



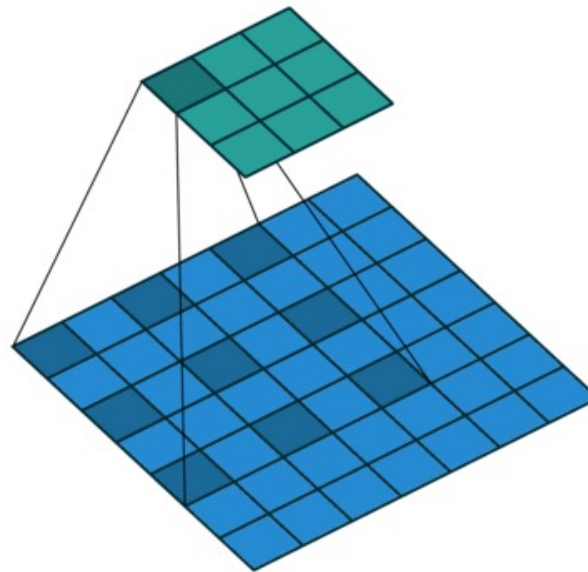
Eingabe: 5x5
Kernel: 3x3
Ausgabe: 2x2
Stride: 2



- Beim rechten Beispiel hätte bei Stride 1 die Ausgabematrix die Größe 3x3.

Dilated Convolution

- Dilation Rate (Dilatationsrate) definiert das Spacing bzw. den Pixelabstand zwischen den vom Kernel betrachteten Positionen bei einer Faltung.
 - Bei der Standardfaltung beträgt die Dilation Rate 1.

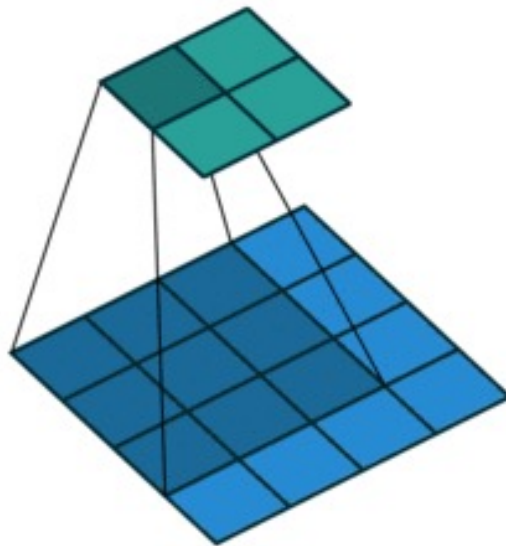


Dilation Rate: 2

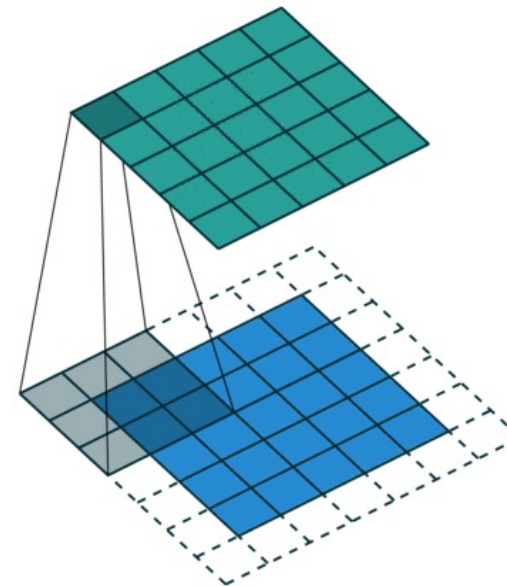
Padding bei der Faltung

- Padding beschreibt die Behandlung der Randpixel der Eingabedaten bei der Faltung.
- Hierdurch wird die Ausgabegröße des gefalteten Bildes beeinflusst.
- Beispiele:

Eingabe: 4x4
Kernel: 3x3
Ausgabe: 2x2
Padding: 0



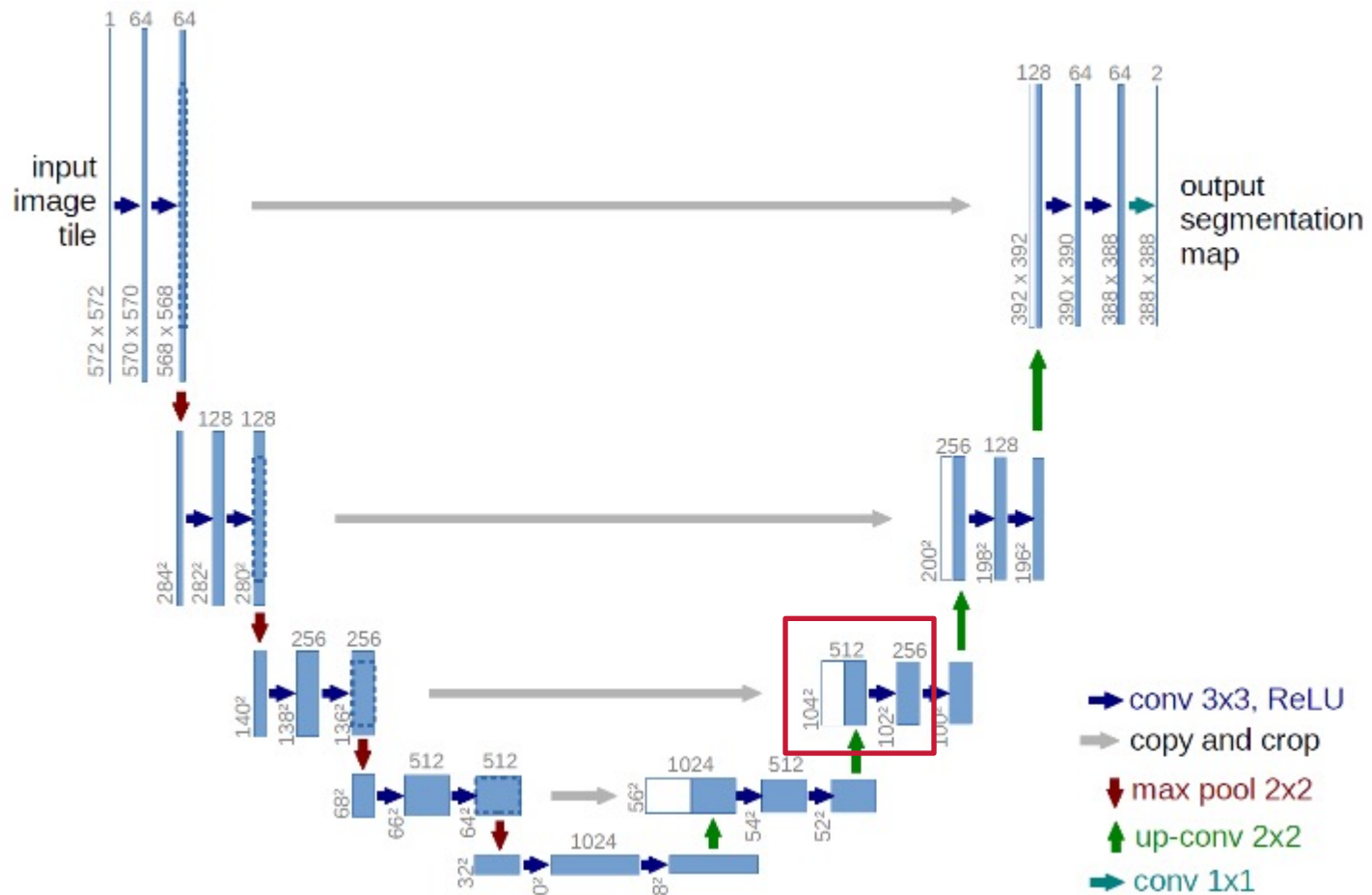
Eingabe: 5x5
Kernel: 3x3
Ausgabe: 5x5
Padding: 1



http://deeplearning.net/software/theano/tutorial/conv_arithmetic.html

- Padding 0: Reduktion der Größe des gefilterten Bildes (hier: von 16 auf 4 Pixel)
- Padding 1: Bildauflösung bleibt bei einem 3x3 Kernel erhalten.

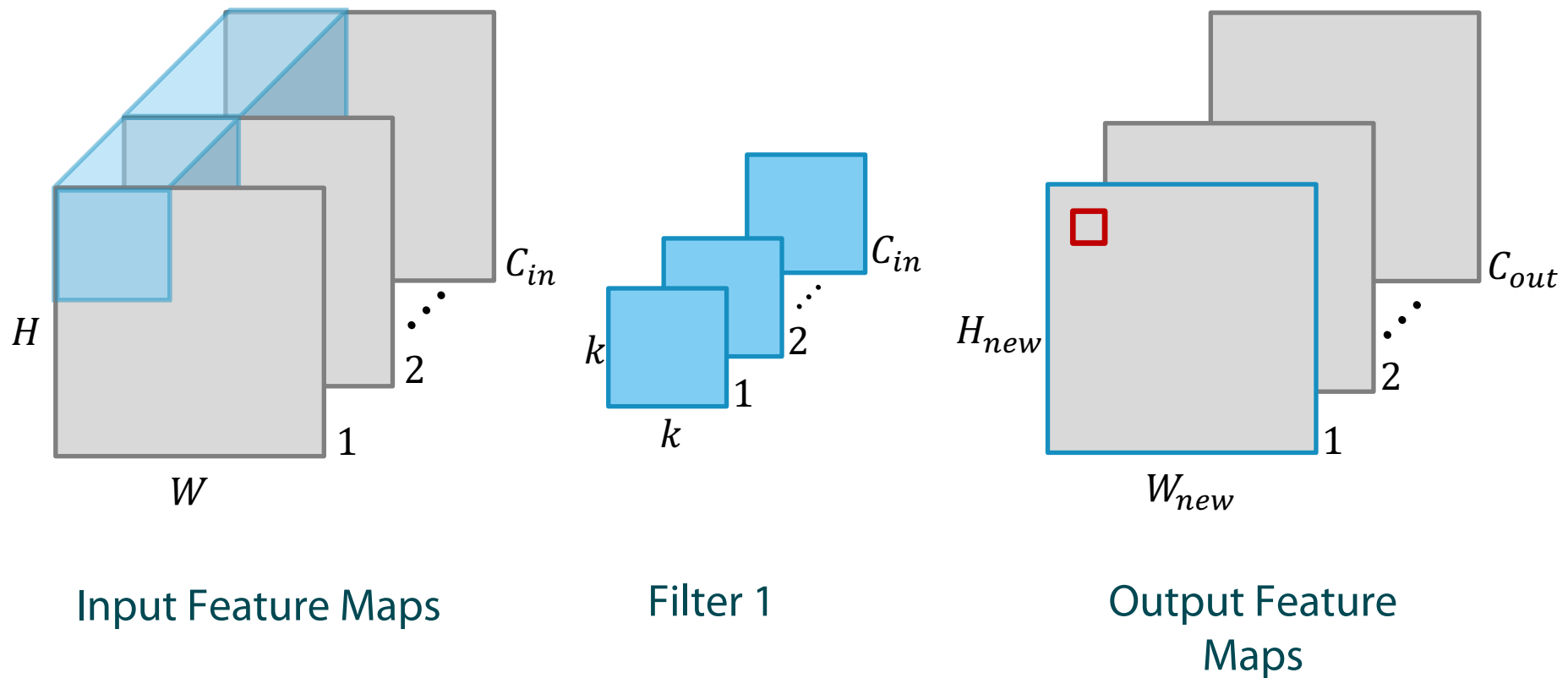
U-Net Architektur



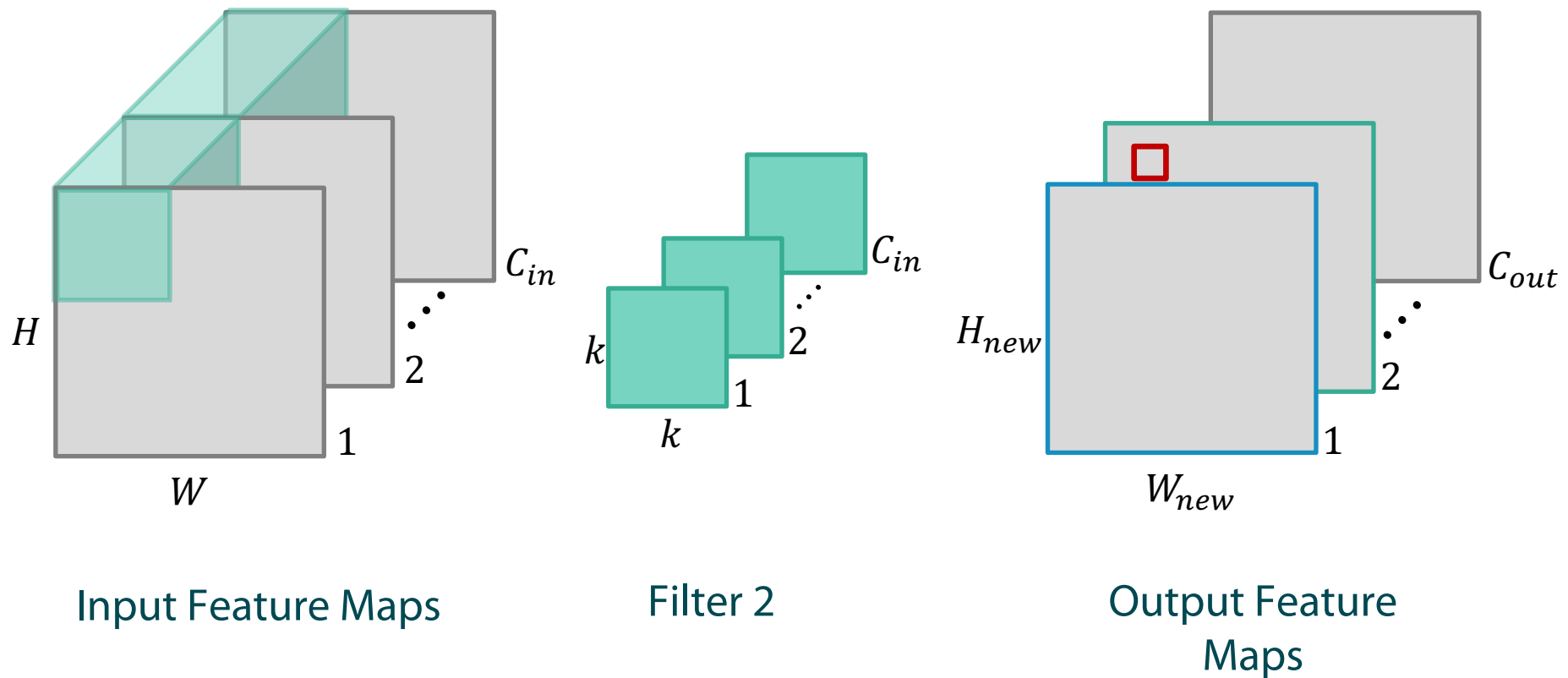
2D Convolution in neuronalen Netzen: Kennzahlen

- Angaben zur Größe und Anzahl der Feature Maps (Input):
 - H : Height, Höhe der eingehenden Feature Maps
 - W : Width, Breite der eingehenden Feature Maps
 - C_{in} : Anzahl der eingehenden Channels bzw. Feature Maps
- Angaben zur Größe und Anzahl der generierten Feature Maps (Output)
 - H_{new} : Height, Höhe der generierten Feature Maps
 - W_{new} : Width, Breite der generierten Feature Maps
 - C_{out} : Anzahl der generierten Channels bzw. Feature Maps
- Angaben der zur Größe und Anzahl der Filterkernel:
 - k : Kernelsize (square $\rightarrow k \times k$)
 - **Die C_{in} Filterkernel der Größe $k \times k$ werden als ein Filterkernel der Größe $k \times k \times C_{in}$ betrachtet!**
 - C_{out} : Anzahl der verwendeten $k \times k \times C_{in}$ Filterkernel
= Anzahl der generierten (neuen) Feature Maps

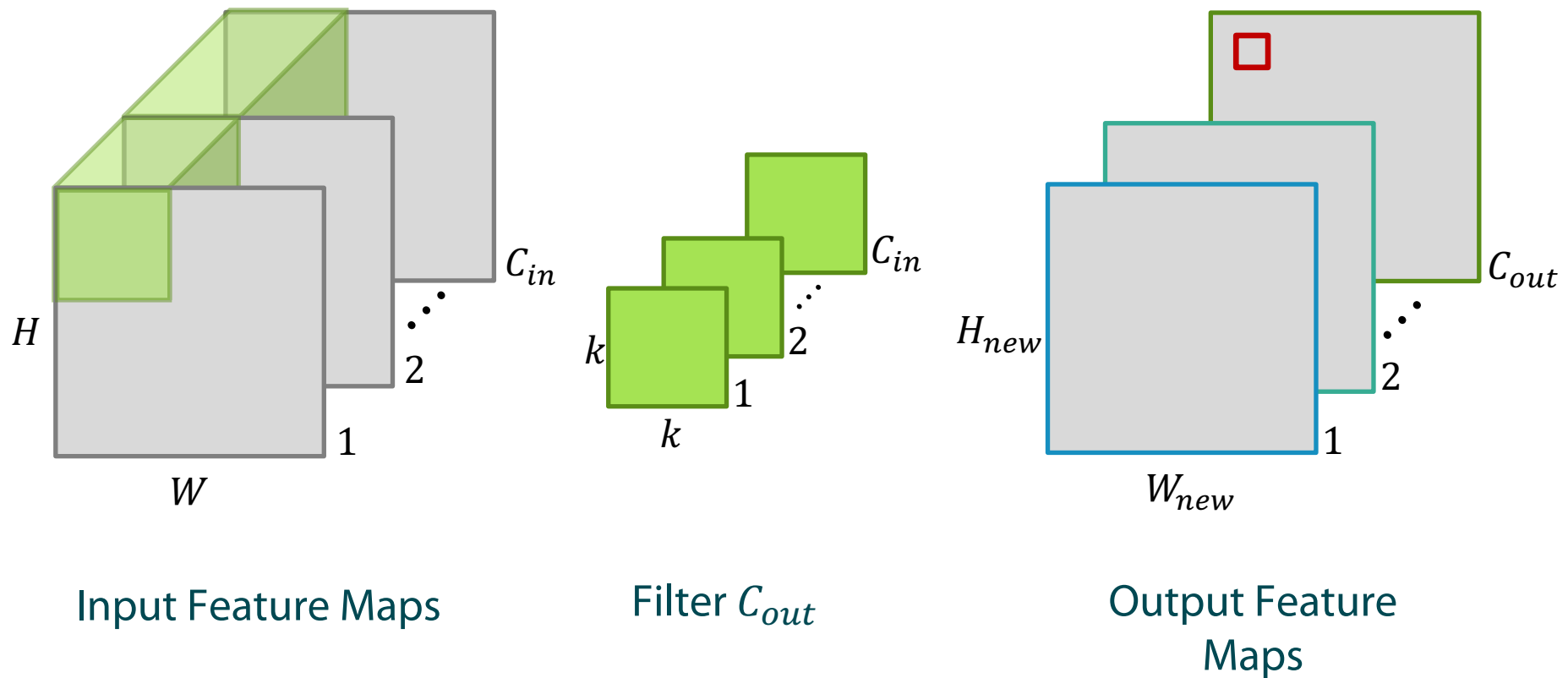
Anwendung: Generierung der 1. Output Feature Map



Anwendung: Generierung der 2. Output Feature Map



Anwendung: Generierung der letzten Output Feature Map



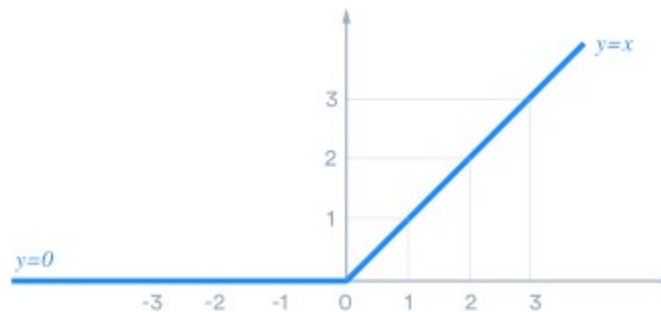
Eigenschaften

- Durch die Wahl der Anzahl der gewählten Filterkernel der Dimension $k \times k \times C_{in}$ wird die Zahl der generierten Feature Maps C_{out} festgelegt.
- Somit kann die Zahl der generierten Feature Maps C_{out} kleiner, gleich groß oder größer sein als die der eingehenden Feature Maps C_{in} .
- Die Größe der bei der Convolution eingehenden und der generierten Merkmalskarten sind je nach Padding-Technik gleich oder geringfügig kleiner.
- Die **Filterkernel** bzw. die hier verwendeten Gewichte werden während des Trainingsprozesses **gelernt**. Für eine einzige Faltungsoperation werden somit $C_{in} \times k \times k \times C_{out}$ Gewichte erlernt.
 - Beispiel:
 - $C_{in} = 256, k = 3, C_{out} = 128$
 - $\rightarrow 294\,912$ Gewichte sind als freie Parameter für diese Faltungsoperation zu erlernen.
- Nach der Convolution wird noch eine Aktivierungsfunktion punktweise auf die generierten Feature Maps angewendet (hier: ReLU).

Rectified Linear Unit als Aktivierungsfunktion

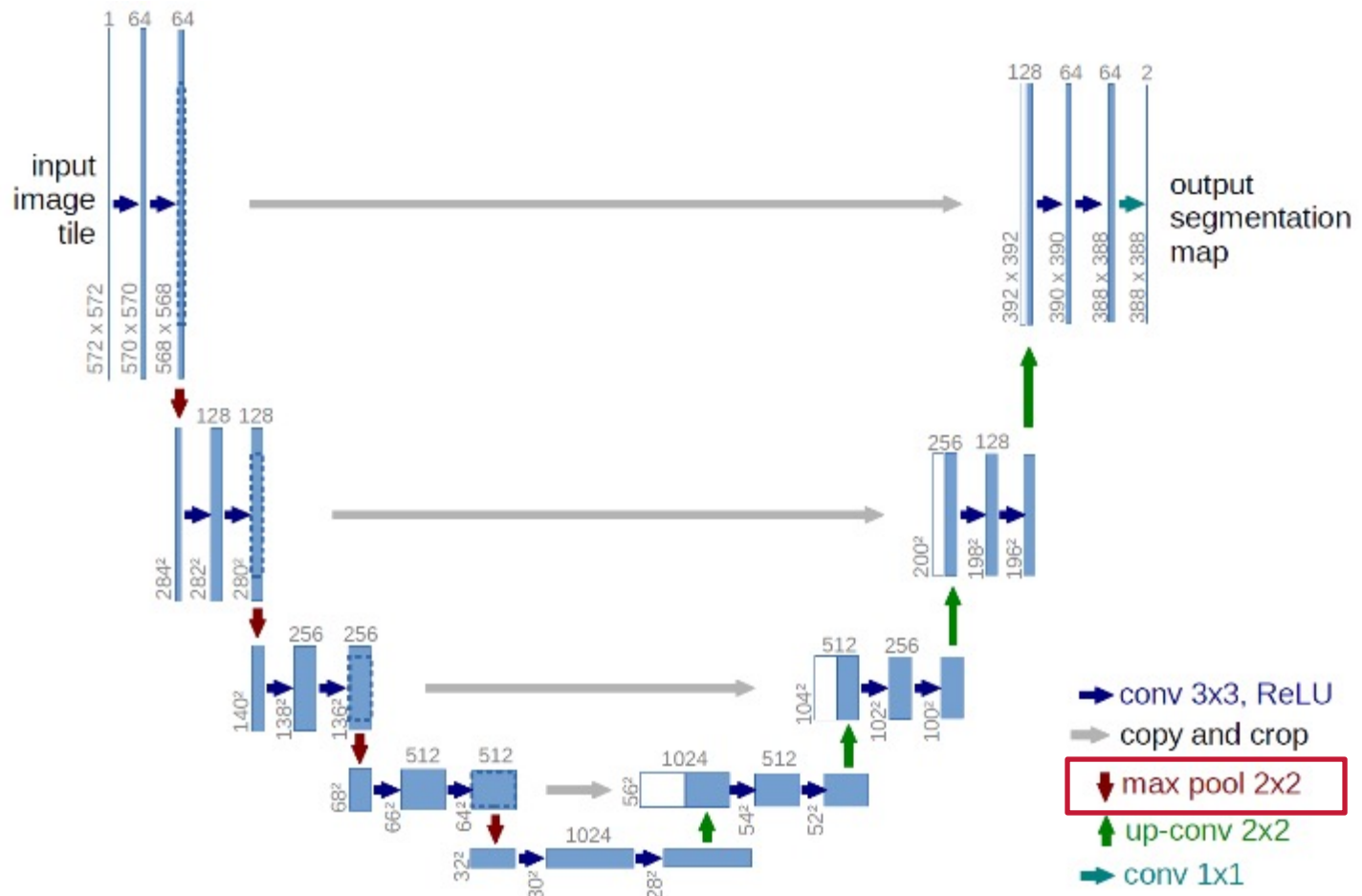
- Rectified Linear Unit (Abk.: ReLU), auch Rectifier genannt

$$f(x) = \max(0, x)$$



- Unterdrückt negative Erregungen und lässt positive Erregungen unverändert durch.
- Häufig verwendete Aktivierungsfunktion in DNNs, die insb. beim Training tiefer neuronaler Netze erfolgreicher war als die Sigmoid-Funktion.

U-Net Architektur



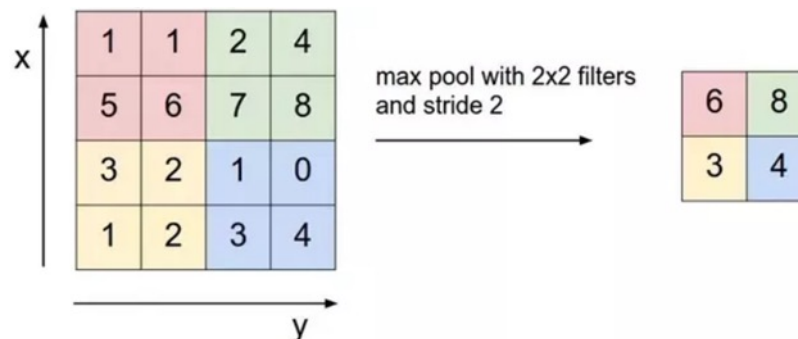
Max-Pooling zum Downsampling

Ziele:

- Reduzierung der Eingaberepräsentation unter Beibehaltung der wesentlichen Information
- Downsampling

Methode:

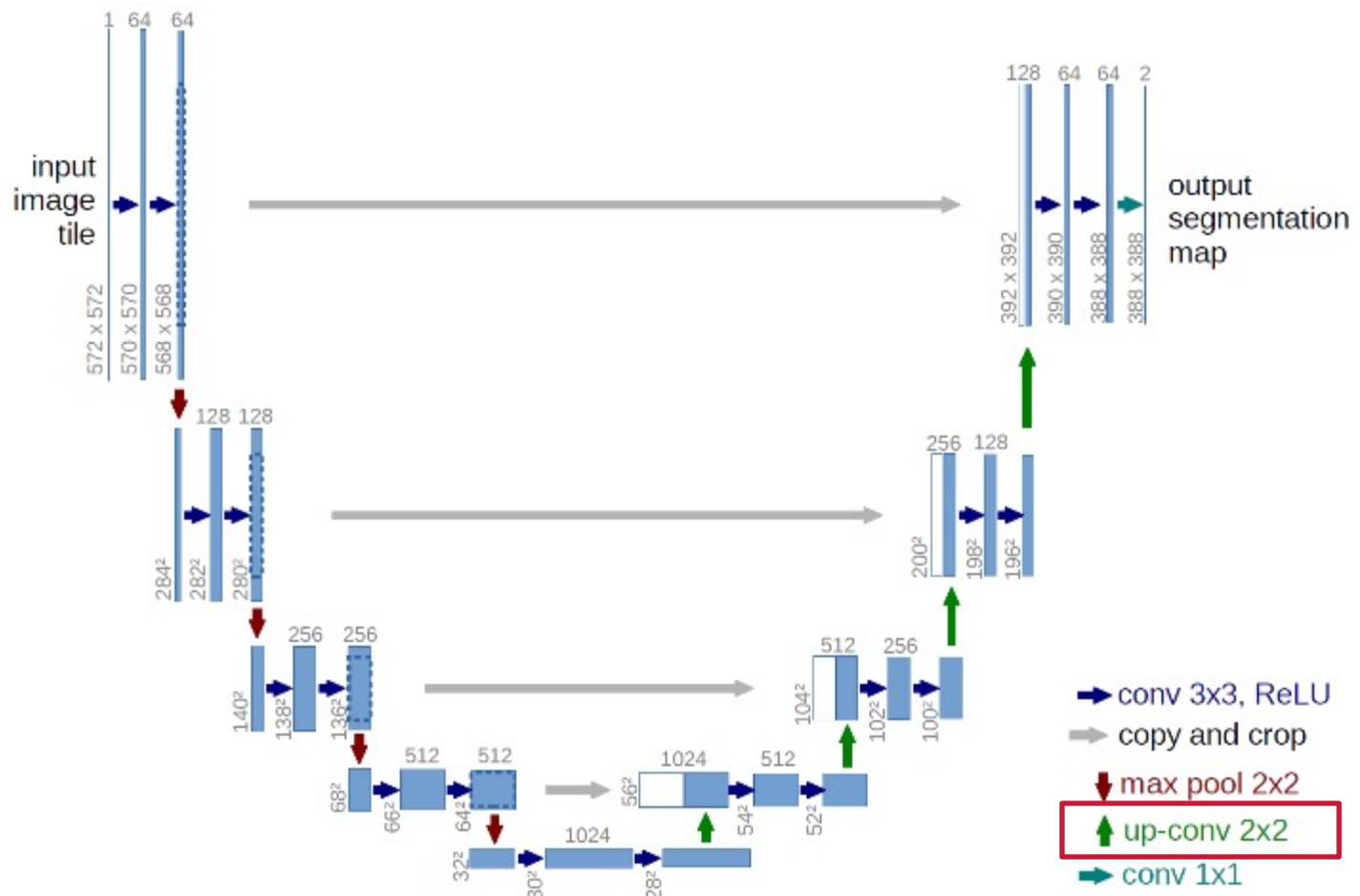
- Maximum-Filterung in lokaler Umgebung
- Nur die Aktivität des aktivsten Neurons wird beibehalten.



Vorteile:

- Geringerer Platzbedarf und erhöhte Berechnungsgeschwindigkeit
- Implizite Vergrößerung der rezeptiven Felder in tieferen Convolutional Layers, ohne dass die Größe der Faltungsmatrizen erhöht werden musste.

U-Net Architektur



Upsampling durch Unpooling (Variante 1)

- Unpooling (Nächster Nachbar Variante)

1	2
3	4

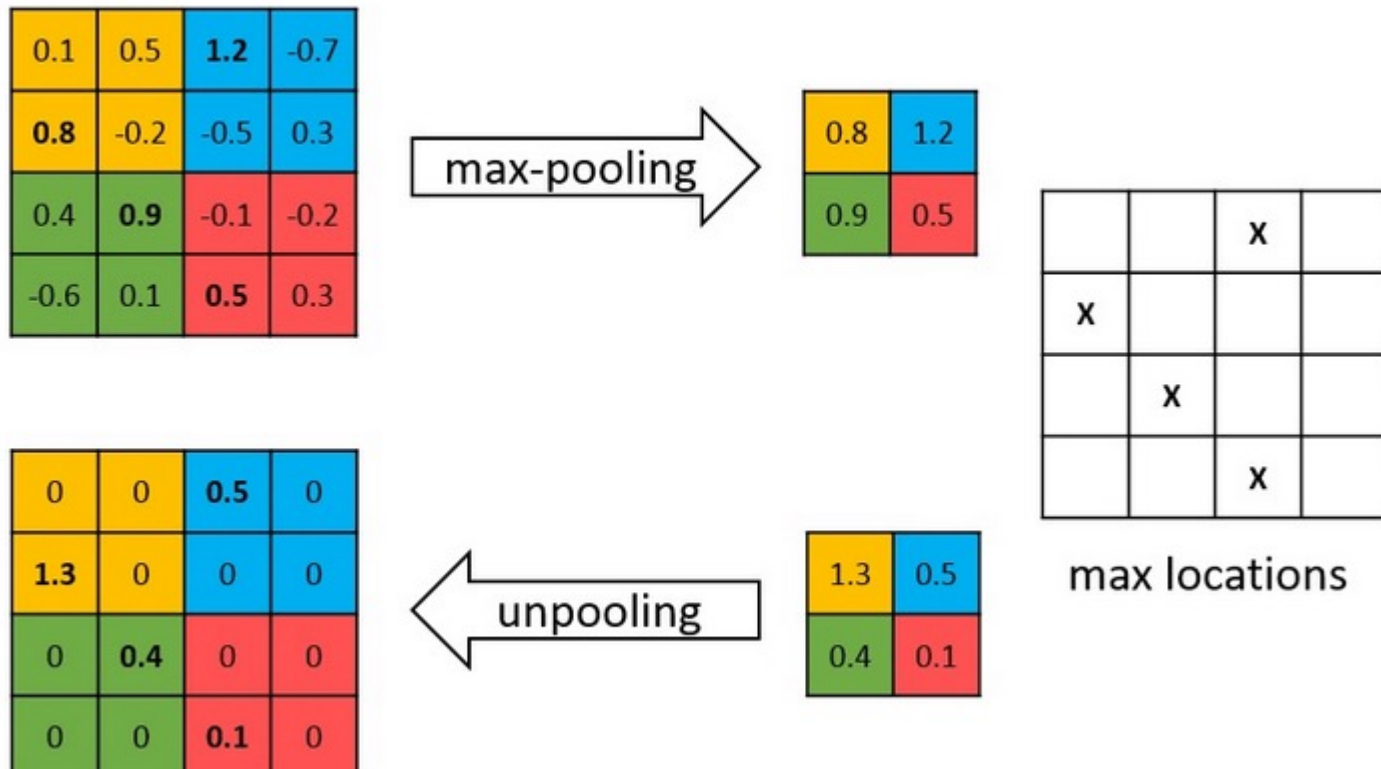


1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Eingabe: 2x2

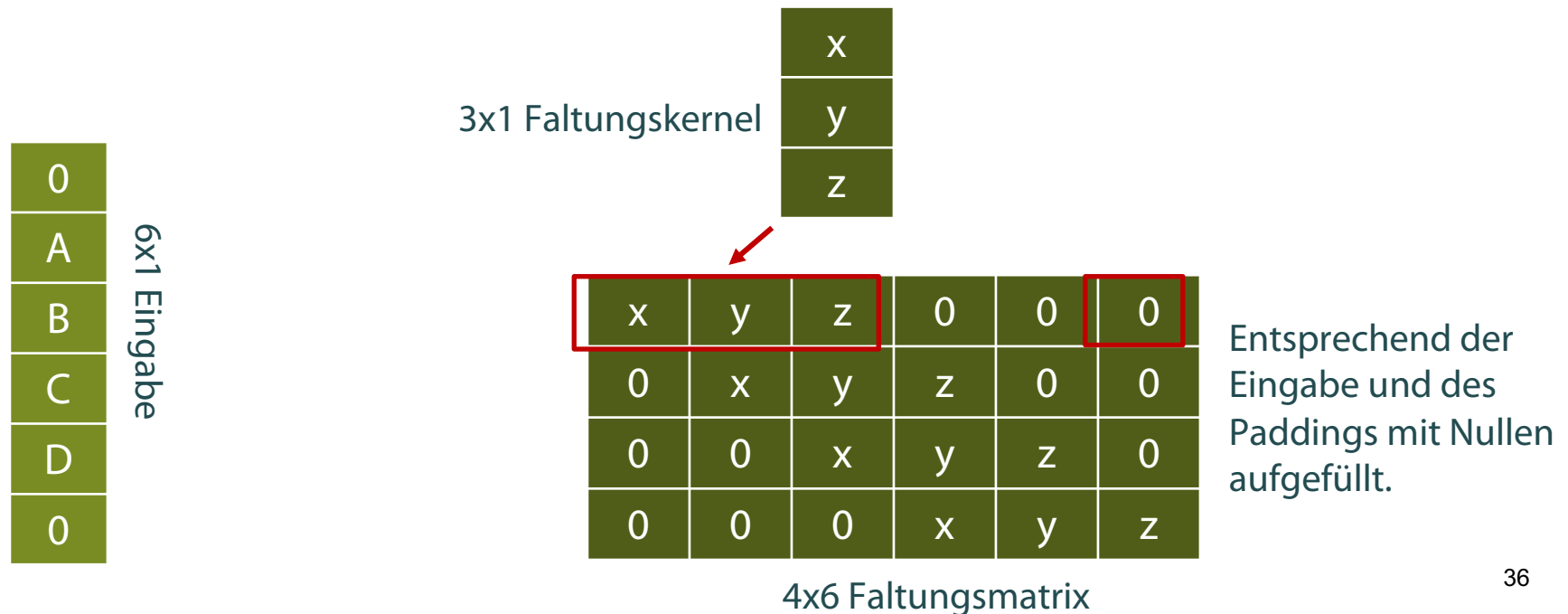
Ausgabe: 4x4

Upsampling durch Unpooling (Variante 2)



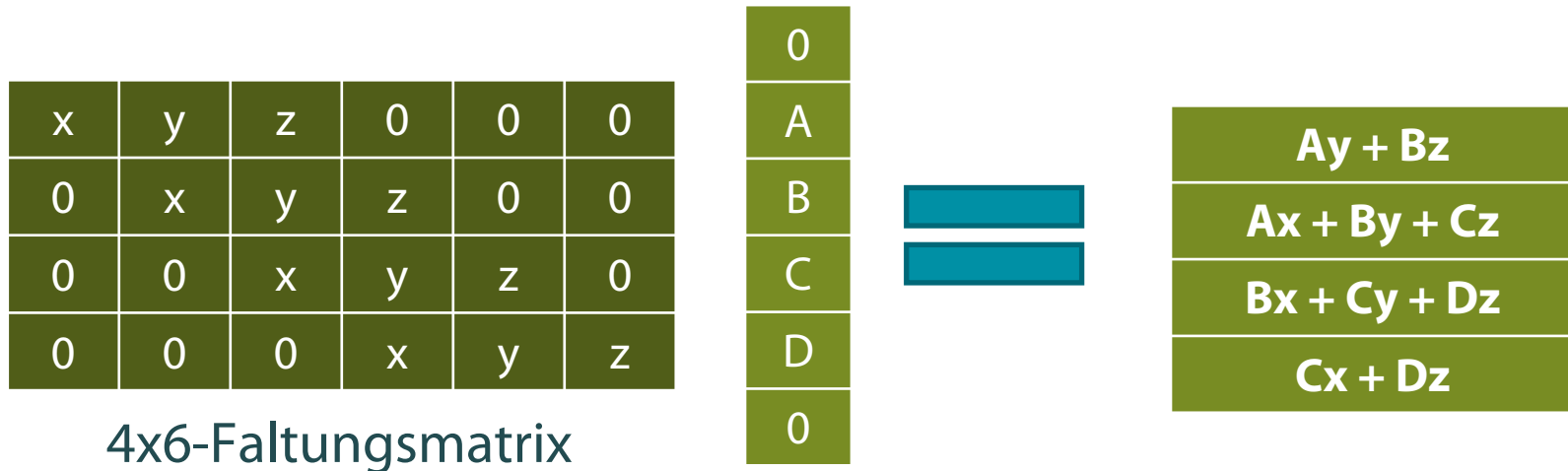
Upsampling durch Deconvolution: Faltung

- Eine Faltung (Convolution) lässt sich als Matrixmultiplikation beschreiben.
- Beispiel: 1D-Faltung, Kernel 3x1, Stride 1, Padding 1



Upsampling durch Deconvolution: Faltung

- Eine Faltung (Convolution) lässt sich als Matrixmultiplikation beschreiben.
- Beispiel: 1D-Faltung, Kernel 3x1, Stride 1, Padding 1



Deconvolution

- Transponieren der Faltungsmatrix
- Daher wird das Verfahren auch „transpose convolution“ genannt.

x	y	z	0	0	0
0	x	y	z	0	0
0	0	x	y	z	0
0	0	0	x	y	z



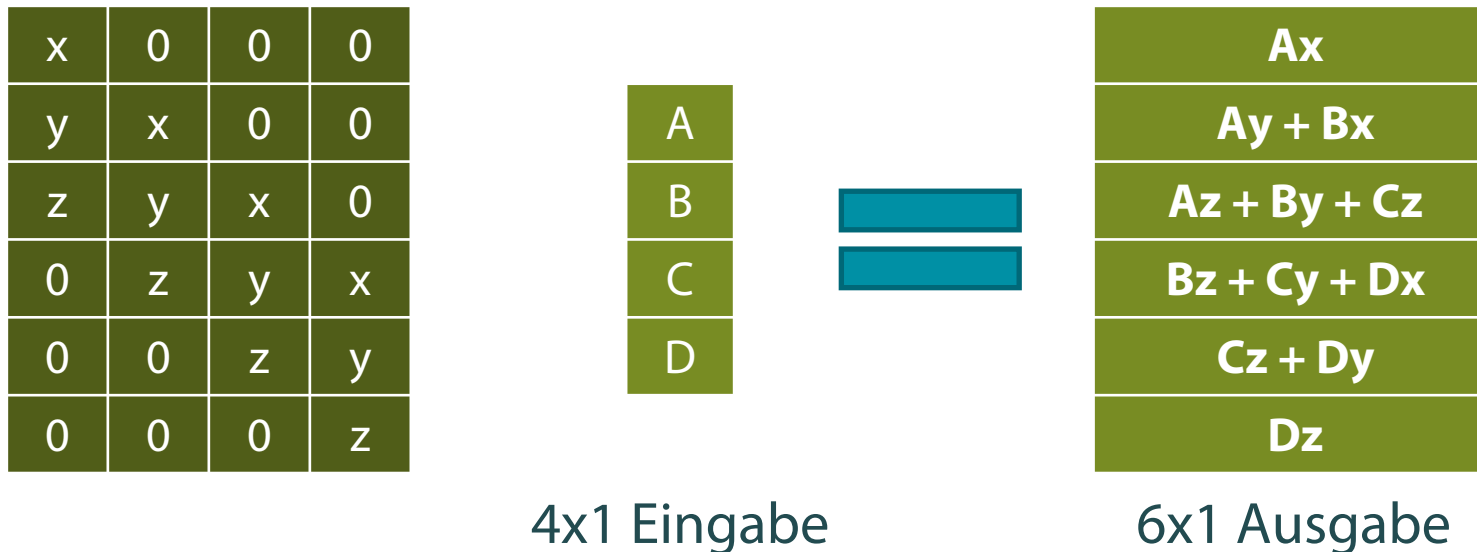
x	0	0	0
y	x	0	0
z	y	x	0
0	z	y	x
0	0	z	y
0	0	0	z

4x6
Faltungsmatrix

Transponierte
6x4- Faltungsmatrix

Deconvolution

- Beispiel: 1D Deconvolution, Stride 1, Padding 1

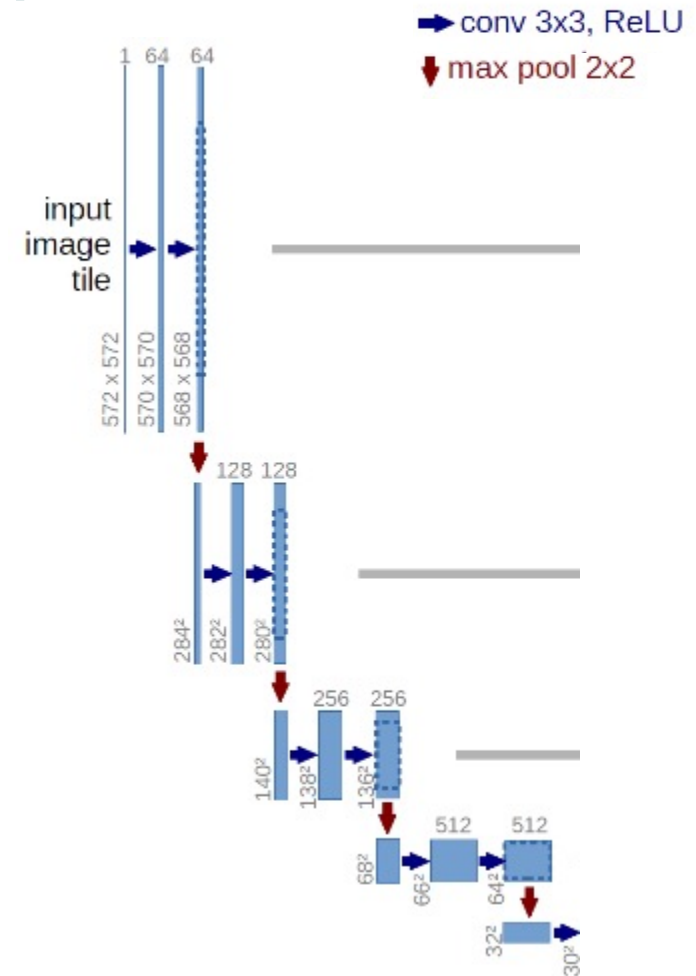


- Bei der Deconvolution werden im Gegensatz zu anderen Upsampling-Methoden lernbare Parameter x , y , z verwendet.
→ durch das Training **optimierbares Upsampling**

U-Net: Kontrahierender Pfad (KP)

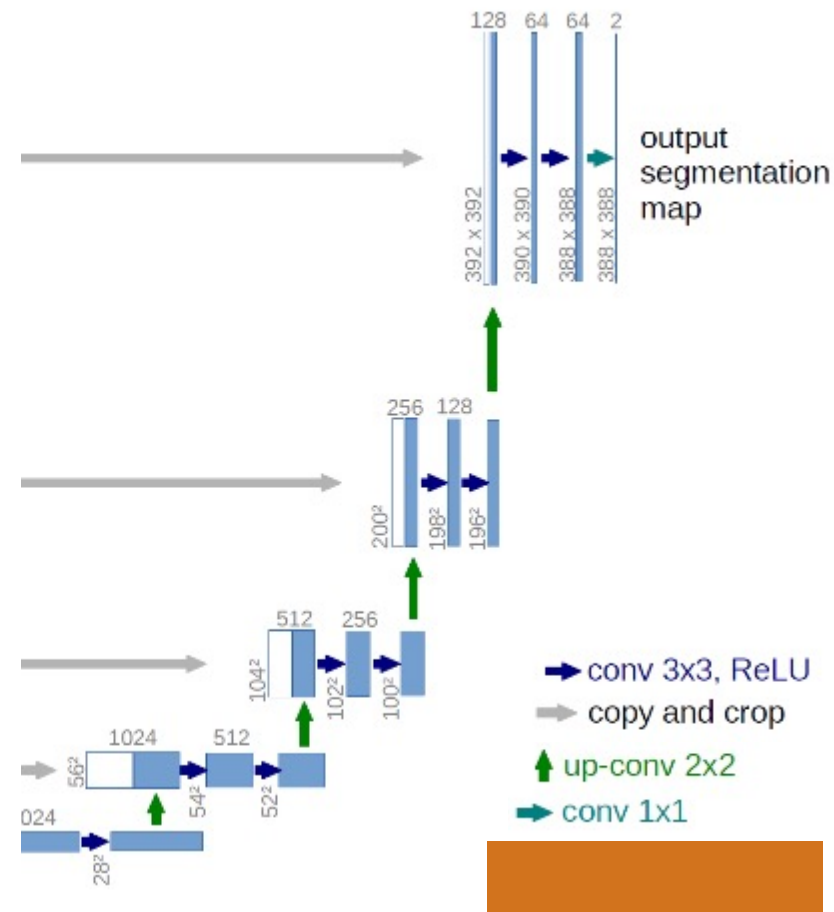
- Wiederholte Anwendung von Convolution-, ReLU- und Max-Pooling Operationen
- Sukzessives Verringern der räumlichen Auflösung (mittels Max-Pooling)
- Sukzessive Erhöhung der Information und Abstraktion in den Feature Maps (Merkmalskarten)

→ Erlernen von **Objektmerkmalen**
und des **Kontextes** des Bildobjektes

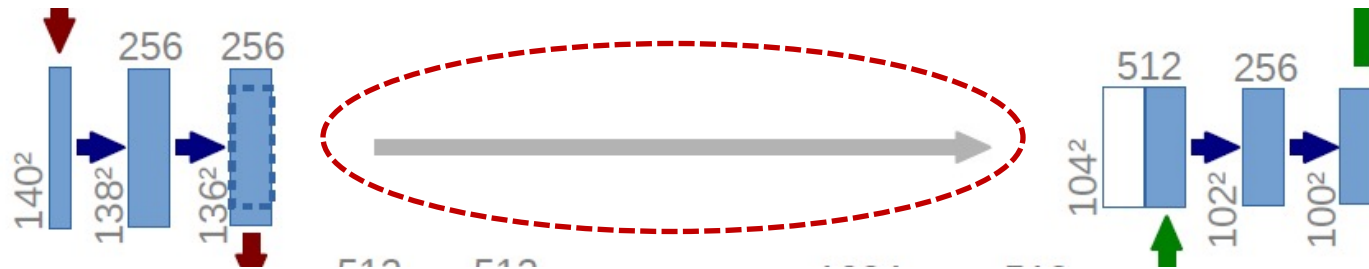


U-Net: Expandierender Pfad (EP)

- Semantische Segmentierung
 - output segmentation map:
 - Dimension 2 (Objekt/Hintergrund)
- Upsampling** zur sukzessiven Vergrößerung der räumlichen Auflösung
- Kombination der Merkmalsinformation und der räumlichen Information** durch Upscaling und Konkatination mit hochaufgelösten Merkmalen des kontrahierenden Pfads über Skip Connections

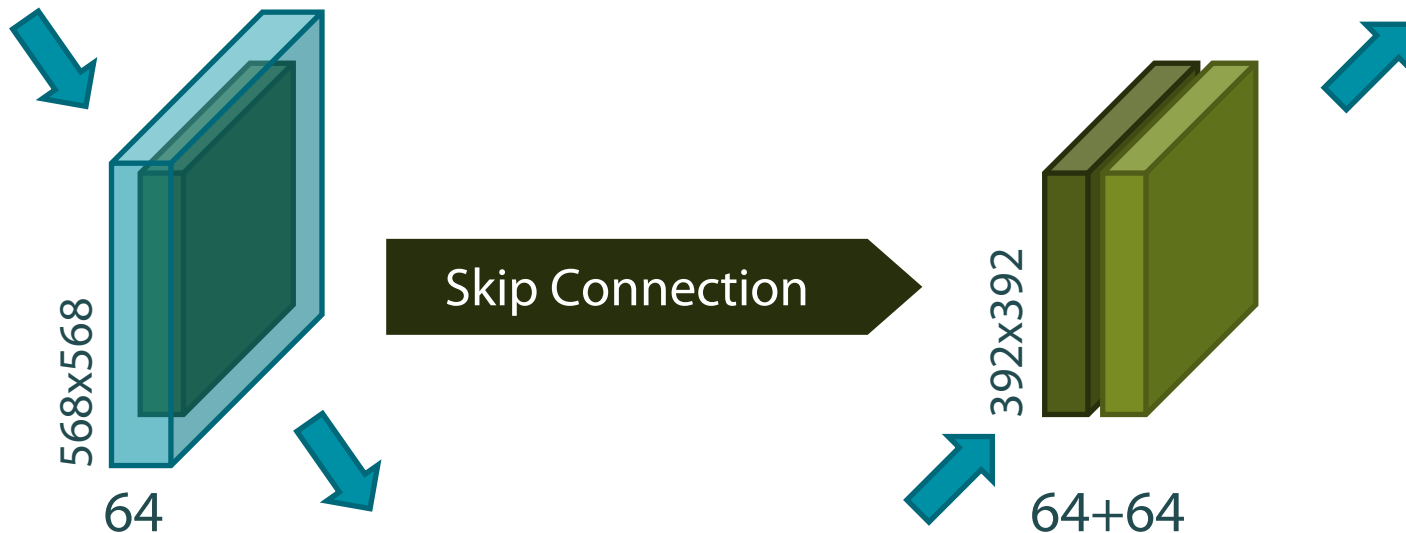


Skip Connections



- Informationen über die Positionierung der Objekte gehen im kontrahierenden Pfad sukzessive verloren bzw. werden sukzessive durch das Max-Pooling vergrößert.
- Skip Connections vom KP zum EP führen diese Information der Rekonstruktion direkt zu.

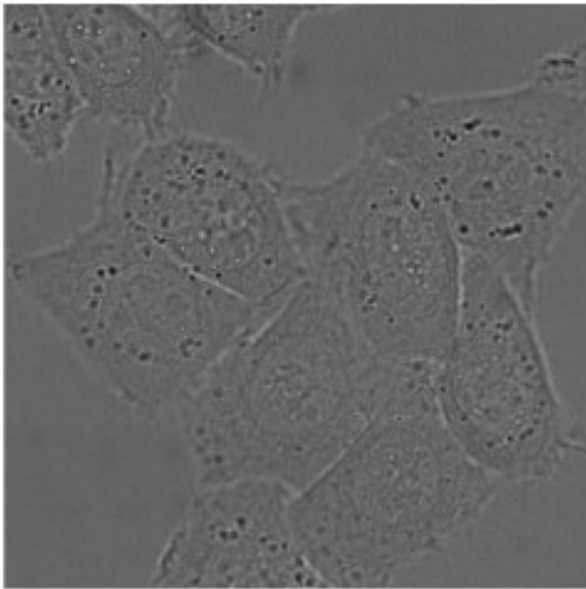
Skip Connections



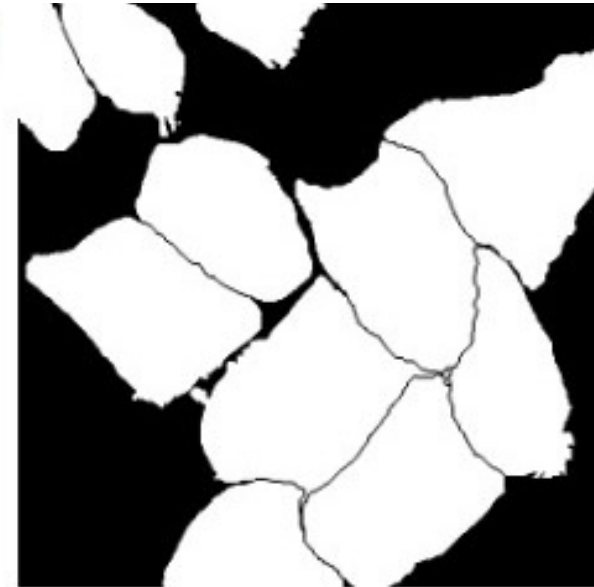
- Die übertragenen Feature Maps des KP werden entsprechend der Bildgröße des EP zugeschnitten.
 - Zentriert auf die Bildmitte wird der Bildrand abgeschnitten.
 - Beispiel: von 568×568 werden diese auf 392×392 reduziert.
- Im EP werden die aus den verschiedenen Channels erhaltenen Feature Maps konkateniert.
 - Beispiel: 64 Feature Maps aus dem KP und 64 aus der darunter liegenden Stufe des EP werden zusammen als Input verwendet, $64 + 64 = 128$

U-Net: Anwendung Zellsegmentierung

- Das U-Net wurde ursprünglich zur Segmentierung von Zellbildern eingesetzt.
- Beispielbild:

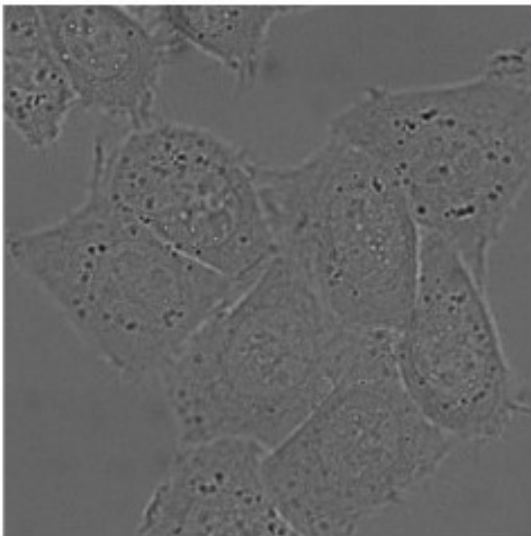


Eingabebild: Mikroskopisches Zellbild

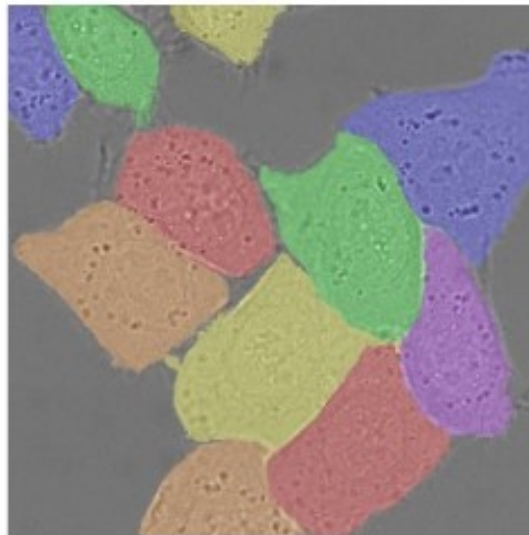


Ausgabebild: Binäre Segmentierung

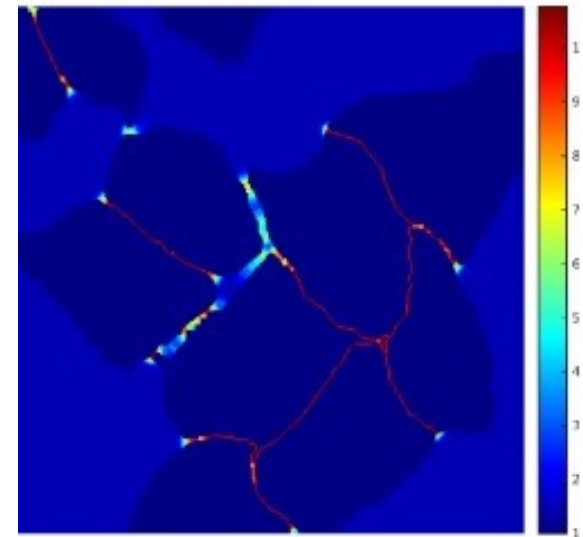
Trainingsdaten



Eingabebild



Ground Truth



Distanzkarte w der
Zellenabstände

Fehlerfunktion beim Training

- Als Fehlerfunktion (Loss) wird nach Anwendung der Softmax-Funktion der Cross Entropie Loss verwendet:

$$E = \sum_x w(x) \log(p_{l(x)}(x))$$

- l ist die Ground Truth Klasse der Pixelposition x
- w ist die Distanzkarte, die eingeführt wurde, um die Segmentierung im Grenzbereich der Zellen zu verbessern.
- Trainingsziel: Minimierung der Fehlerfunktion E durch den Backpropagation-Algorithmus (Gradientenabstieg)

Fehlerfunktion beim Training

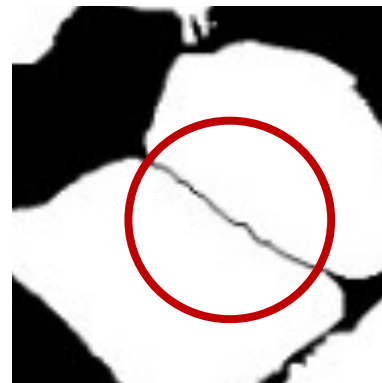
- Als Fehlerfunktion (Loss) wird nach Anwendung der Softmax-Funktion der Cross Entropie Loss verwendet:

$$E = \sum_x w(x) \log(p_{l(x)}(x))$$

- l ist die Ground Truth Klasse der Pixelposition x
- w ist die Distanzkarte, die eingeführt wurde, um die Segmentierung im Grenzbereich der Zellen zu verbessern.
- Trainingsziel: Minimierung der Fehlerfunktion E durch den Backpropagation-Algorithmus (Gradientenabstieg)

Distanzkarte

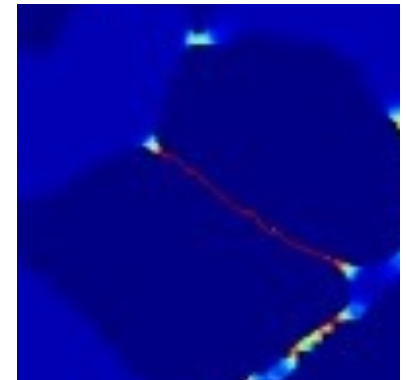
- Viele der zu segmentierenden Objekte bzw. Zellen teilen sich Objektgrenzen.
- Die Distanzkarte w wird während des Trainings als Gewichtung in der zu minimierenden Fehlerfunktion E eingesetzt.
- Die Gewichtung mit den Distanzen sorgt für einen erhöhten Fokus auf die präzise Segmentierung der Grenzpixel.



Distanzkarte

$$w(\mathbf{x}) = w_k(\mathbf{x}) + w_0 * \exp\left(-\frac{(d_1(\mathbf{x}) + d_2(\mathbf{x}))^2}{2\sigma^2}\right)$$

- w_k ist eine Gewichtung der Klassenhäufigkeit (hier: 2 Klassen)
- d_1 Distanz zur Grenze der nächsten Zelle
- d_2 Distanz zur Grenze der zweitnächsten Zelle
- Die Autoren verwenden:
 - $w_0 = 10$
 - $\sigma = 5$



Fehlerfunktion im Training

- Als Fehlerfunktion (Loss) wird nach Anwendung der Softmax-Funktion der Cross Entropy Loss verwendet:

$$E = \sum_x w(x) \log(p_{l(x)}(\mathbf{x}))$$

- l ist die Ground Truth Klasse der Pixelposition \mathbf{x}
- w ist die Distanzkarte, die eingeführt wurde, um die Segmentierung im Grenzbereich der Zellen zu verbessern.
- Training: Minimierung der Fehlerfunktion E durch den Backpropagation-Algorithmus (Gradientenabstieg)

Softmax-Funktion

- Die Softmax-Funktion transformiert die neuronalen Aktivierungen in eine kategoriale Wahrscheinlichkeitsverteilung.

$$p_k(\mathbf{x}) = \exp(a_k(\mathbf{x})) / \sum_{k=1}^K \exp(a_k(\mathbf{x}))$$

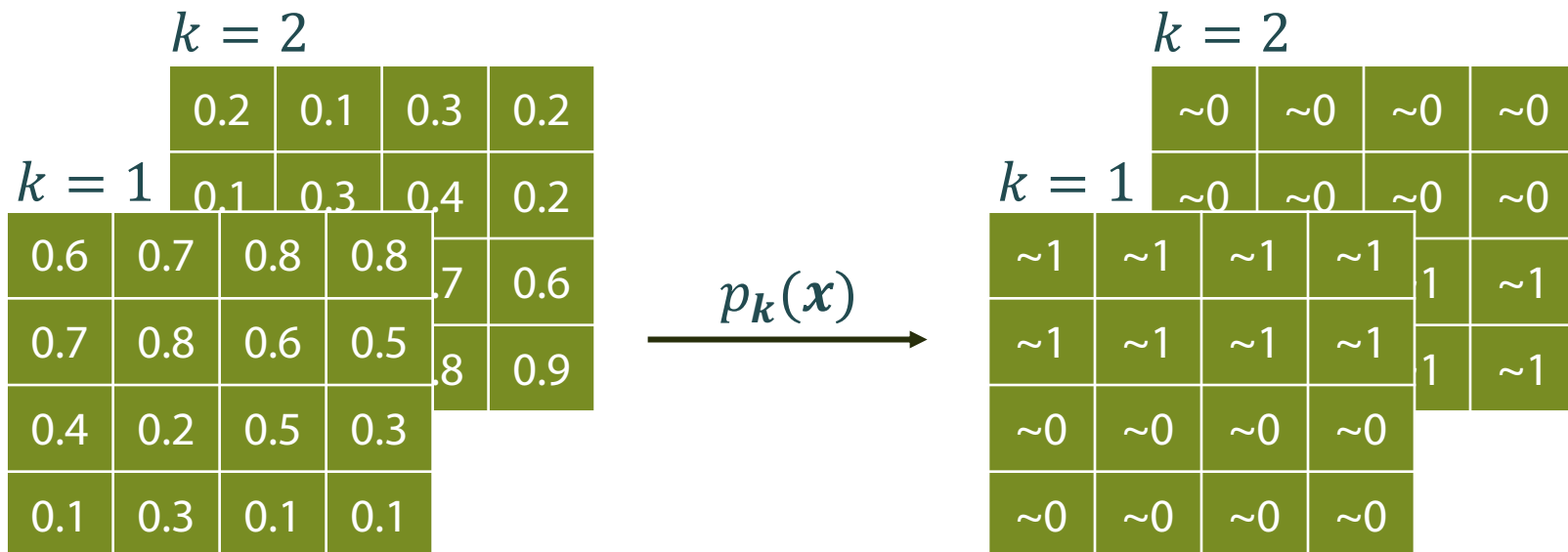
$p_k(\mathbf{x})$ summiert sich für alle K Klassen zu 1.

$a_k(\mathbf{x})$ ist die Aktivierung der Pixelposition \mathbf{x} des Channels der Klasse k

- $p_k(\mathbf{x})$ gibt die Wahrscheinlichkeit an, dass der Input an der Pixelposition \mathbf{x} der Kategorie/Klasse k zugeordnet wird.
- Effekt:** Nach Anwendung der Softmax-Funktion auf die Aktivierungen sind die **Unterschiede zwischen den Klassen deutlicher**.
- Bem.: Softmax-Funktion wird auch als normalisierte Exponentialfunktion bezeichnet.

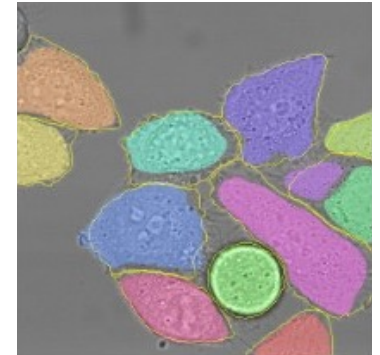
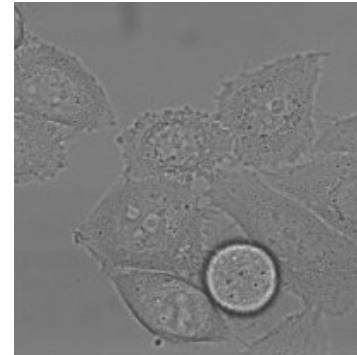
Fehlerfunktion im Training: Beispiel

- Für die betrachtete Zellsegmentierung gibt das U-Net 2 Channels aus:
 - $k=1$: Hintergrund
 - $k=2$: Zellen

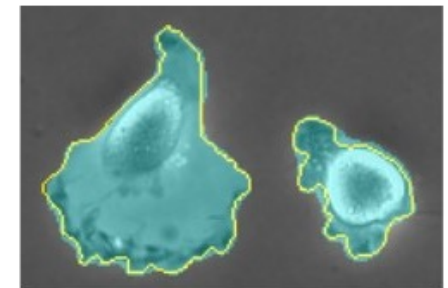
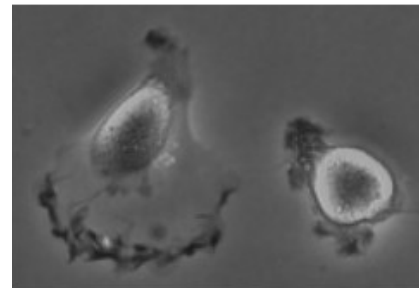


Ergebnisse

- Dataset: „DIC-HeLa“
 - 1. Platz, U-Net: 77.5% IOU
 - 2. Platz: 46% IOU

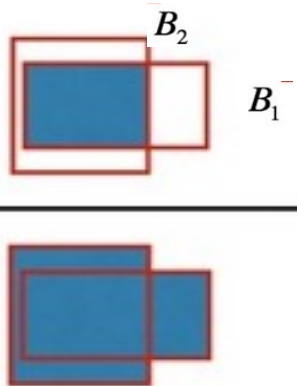


- Dataset: „PhC-U373“
 - 1. Platz, U-Net: 92% IOU
 - 2. Platz: 83% IOU



Qualitätsmaß: Intersection over Union (IoU)

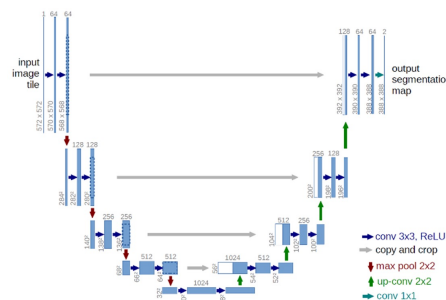
- Auch als **Jaccard Koeffizient** bekannt
- Bewertet die Qualität einer Segmentierung durch Betrachtung der Überdeckungen der erhaltenen Segmentierung mit der „wahren“ Segmentierung.
 - IoU ist gleich 1, wenn eine perfekte Überlappung und somit eine ideale Segmentierung erreicht wurde.
 - IoU ist gleich 0, wenn die Segmentierungsbereiche nicht überlappen.

$$IoU = \frac{B_1 \cap B_2}{B_1 \cup B_2} =$$


- Quotient der Schnittmenge (\cap) und der Vereinigung (\cup) der aktuellen Segmentierung B_1 und der „wahren“ Segmentierung B_2

Zusammenfassung

- Das U-Net setzt sich aus zwei Komponenten zusammen:
 - Kontrahierender Pfad: Erlernst die wesentlichen Objektmerkmale und den Kontext des Bildobjektes
 - Expandierender Pfad: Rekonstruiert die räumliche Position und verwendet eine Kombination der Merkmalsinformation und der räumlichen Information durch Upscaling und Konkatination mit hochaufgelösten Merkmalen des kontrahierenden Pfads über Skip-Connections
- Das U-Net ist eines der erfolgreichsten neuronalen Deep Learning Netze in der Medizin.

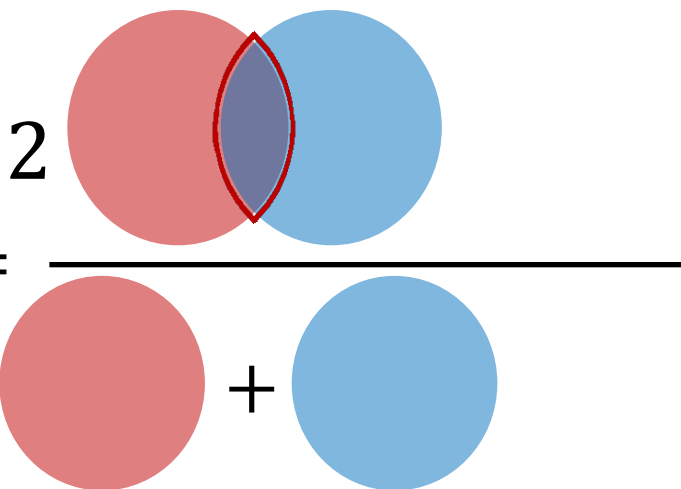


Weitere Loss-Funktionen zum Training von Segmentierungsnetzwerken

- Dice Loss
- Generalisierter Dice Loss

Dice-Koeffizient

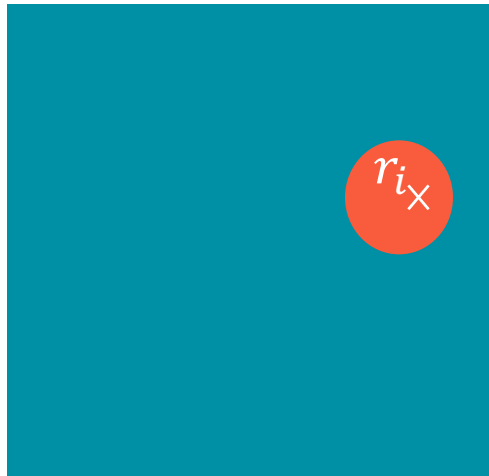
- Bewertet die Qualität einer Segmentierung durch Betrachtung der Überdeckungen der erhaltenen Segmentierung mit der „wahren“ Segmentierung.
 - Der Dice-Koeffizient ist gleich 1, wenn eine perfekte Überlappung und somit eine ideale Segmentierung erreicht wurde.
 - Der Dice-Koeffizient ist gleich 0, wenn die Segmentierungsbereiche nicht überlappen.

$$Dice = \frac{2 |A \cap B|}{|A| + |B|} = \frac{2 \cdot \text{Überlappung}}{\text{A} + \text{B}}$$


- Idee: Benutzung des Dice-Koeffizienten zur Definition einer Fehlerfunktion E, **Dice Loss** genannt, beim Training von neuronalen Segmentierungsnetzen

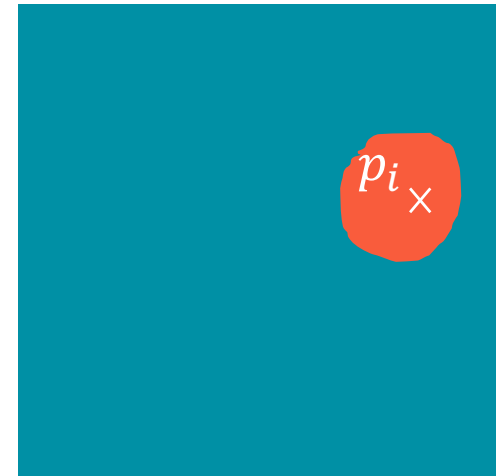
DICE Loss: Notationen

Grundwahrheit



r_n : Label der Referenzsegmentierung des
Bildobjektes (ground truth)
Objeklabel: $r_n = 1$
Hintergrundlabel: $r_n = 0$

Prädiktion durch das Netz



p_n : Vorhergesagte Wahrscheinlichkeit des Auftretens des
Objeklabels bzw. Aktivierung des Ausgabeneurons der Objektklasse

 $(1 - p_n)$: Vorhergesagte Wahrscheinlichkeit
des Auftretens des Hintergrundlabels bzw. Aktivierung des
Ausgabeneurons der Hintergrundklasse

2 Klassen (Objekt ■, Hintergrund ■)

Dice Loss in neuronalen Netzen

$$DL_2 = 1 - \frac{\sum_{n=1}^N p_n r_n + \epsilon}{\sum_{n=1}^N p_n + r_n + \epsilon} - \frac{\sum_{n=1}^N (1 - p_n)(1 - r_n) + \epsilon}{\sum_{n=1}^N 2 - p_n - r_n + \epsilon}$$

- Dice Loss (DL) für $k=2$ Klassen
- Der Dice Loss ist umgekehrt proportional zum Dice-Koeffizienten.
 - D.h. je größer der Dice-Koeffizient, desto kleiner ist der Dice Loss (und umgekehrt)
- **DL_2** wird während des Trainings sukzessive minimiert.
 - Dice Loss wird während des Trainings umso kleiner, je stärker die wahre Segmentierung mit Bildregionen mit hohen Aktivierungswerten für die Ausgabeneuronen der Objektklasse überlappt (und umgekehrt).
- Beim DL liegt eine negative Korrelation mit der Größe der Segmentierung vor.
 - d.h. **DL** ist bei kleinen Objekten tendenziell größer als bei großen Objekten
 - Daher sind sehr kleine Objekte unter Verwendung des **DL** schwierig zu segmentieren.

Generalisierter Dice Loss

- Verwendung für mehr als 2 Klassen möglich
- Für den Generalized *DL* (*GDL*) wird im Vorfeld für jedes Label bzw. Klassen eine Gewichtung berechnet.
- Das Gewicht w_l korrigiert der Beitrag jedes Labels durch den Kehrwert seines Anteils am Bild.

$$w_l = \frac{1}{(\sum_{n=1}^N r_{ln})^2}$$

Generalisierter Dice Loss

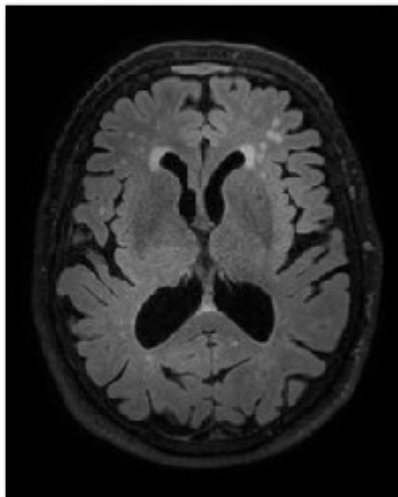
- l : Anzahl der Klassen bzw. Labels (hier 2)
- r_{ln} : Labelwert der Grundwahrheit
- p_{ln} : Wahrscheinlichkeit der Prädiktion durch das Netz von Klasse l

$$GDL = 1 - 2 \frac{\sum_{l=1}^2 w_l \sum_n r_{ln} p_{ln}}{\sum_{l=1}^2 w_l \sum_n r_{ln} + p_{ln}}$$

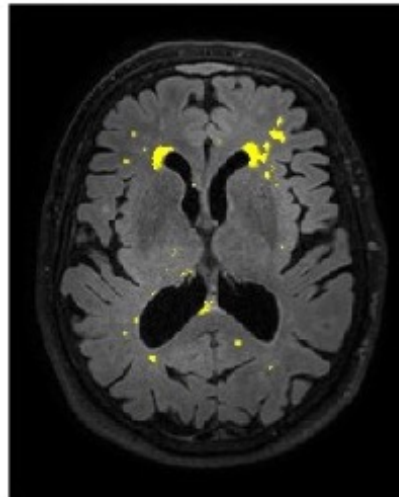
- Durch den GDL ist es möglich, auch kleine Bildstrukturen mit geringem Anteil an der Gesamtpixelmenge zu segmentieren.

Segmentierungsergebnisse unter Verwendung verschiedener Loss-Funktionen

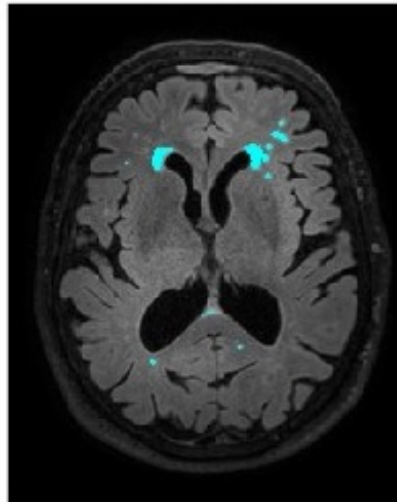
FLAIR



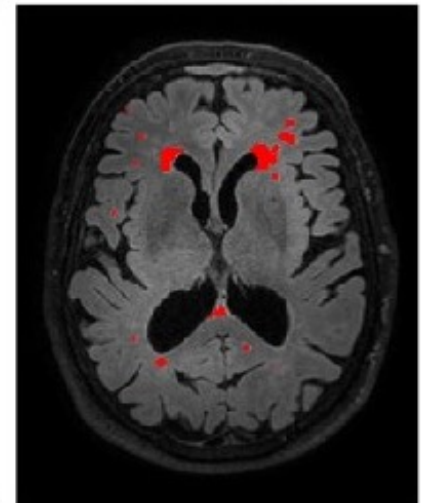
Gold Standard



DL_2



GDL_v



Vom U-Net zum nnUnet

- In Praxis ist nicht nur Netz-Architektur entscheidend, sondern auch:
 - Vorverarbeitung der Daten (u.a. Normalisierung, Resampling)
 - Datenaugmentierung
 - Hyperparameter (u.a. Lernrate, Batch-Größe)
 - Aufteilung in Trainings-, Validierungsdaten
 - Loss-Funktionen
 - Hardware
- nnUnet („no new u-net“) führt automatisch Augmentierungen, Vorverarbeitungen durch und bestimmt optimale Hyperparameter
- Das erspart das manuelle Optimieren
- Aktueller Stand der Technik für (DL-basierte) Segmentierung