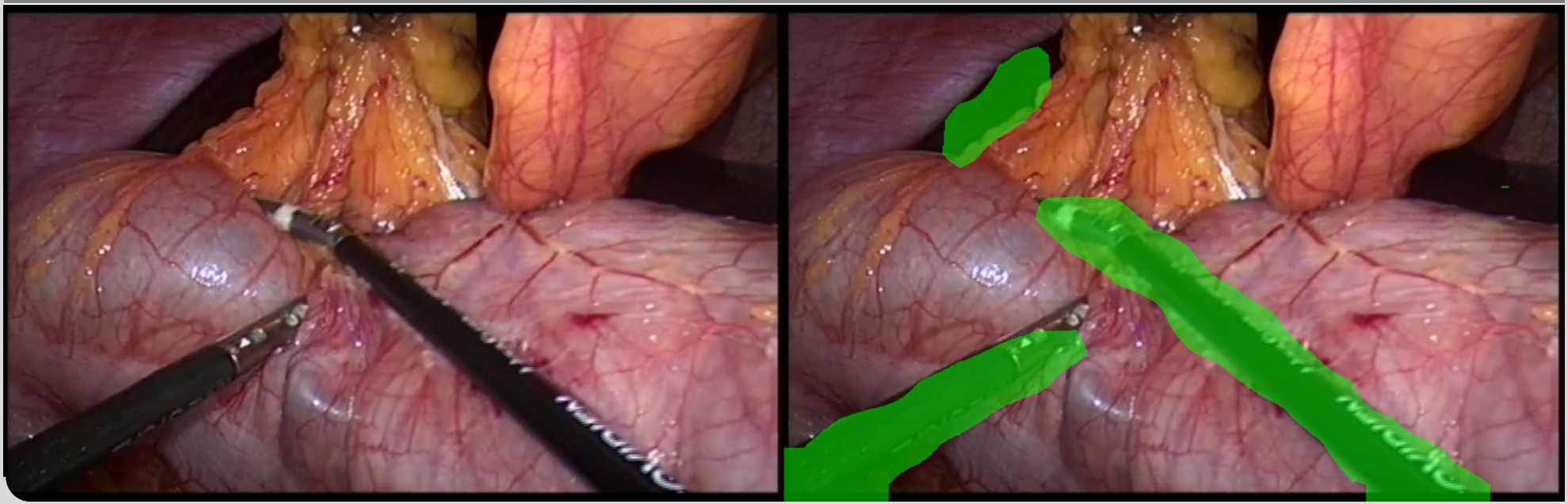
Karlsruher Institut für Technologie

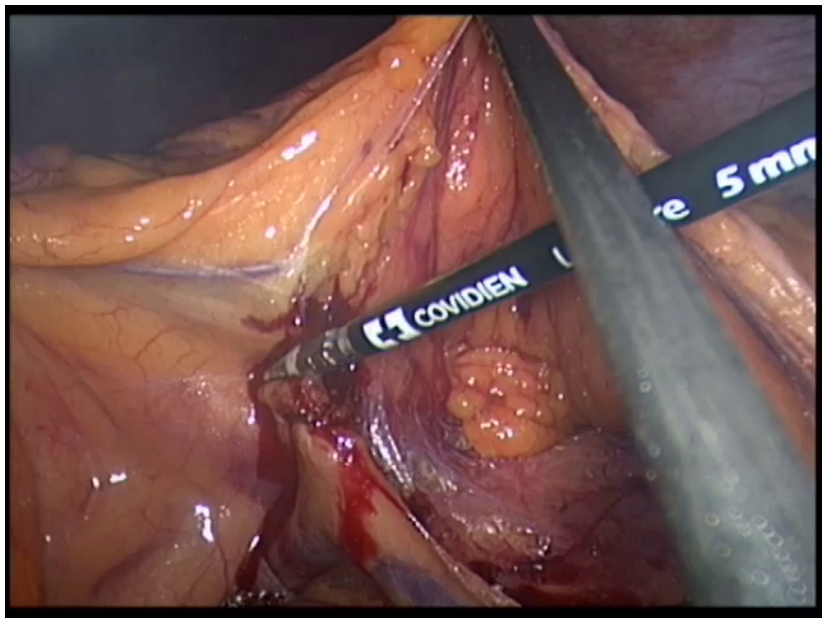
Semantische Segmentierung mit Deep Learning

Marvin Teichman und Martin Thoma | 19. Juli 2016

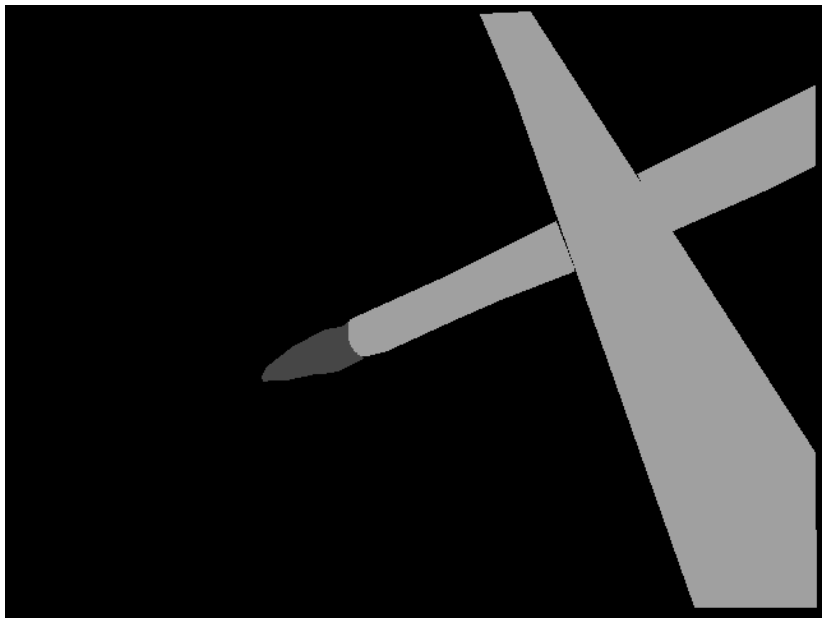
SEMINAR INFORMATIK



Problemstellung



Endoskop-Bild (input)
RGB-Bilder mit 640 px × 480 px als Eingabe.



Segmentation mask (output)
3-Farben Bilder mit 640 px × 480 px als Segmentierungsmaske

Die Daten stammen aus Endoscopic Sub-Challenge „Instrument Segmentation and Tracking“.

Taxonomie nach Thoma 2016: A survey of semantic segmenation

- Klassen: (1) Medizinisches Instrument (2) Nicht „medizinisches Instrument“ / Hintergrund
- Jeder Pixel gehört zu genau einer Klasse.
- Daten: vgl. vorherige Folie
- Keine Tiefeninformationen; keine Stereo-Bilder; 2D
- Passive, automatische Segmentierung

Nur Farbinformationen

Die Farbe (3 Features, RGB) eines Pixels gibt einen Hinweis darauf, ob es sich um ein medizinisches Instrument handelt. Dies wird als Baseline zum Vergleich mit anderen Techniken verwendet.

Model	Accuracy	Precision	Recall
Baseline	92.88 %	76.13 %	32.94 %

Zur Interpretation:

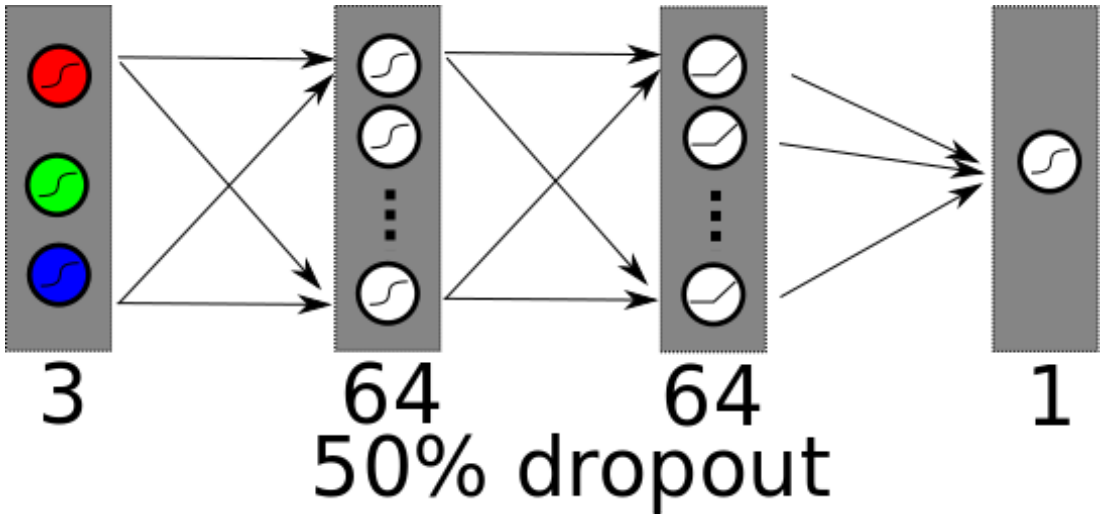
- 90.5 % sind nicht „medizinisches Instrument“
- 9.5 % sind medizinisches Instrument

$\Rightarrow$ 90.5 % Accuracy sind trivial zu erreichen.

Nur Farbinformationen

Die Farbe (3 Features, RGB) eines Pixels gibt einen Hinweis darauf, ob es sich um ein medizinisches Instrument handelt. Dies wird als Baseline zum Vergleich mit anderen Techniken verwendet.

Model	Accuracy	Precision	Recall
Baseline	92.88 %	76.13 %	32.94 %



Kontext + Morphologische Operationen

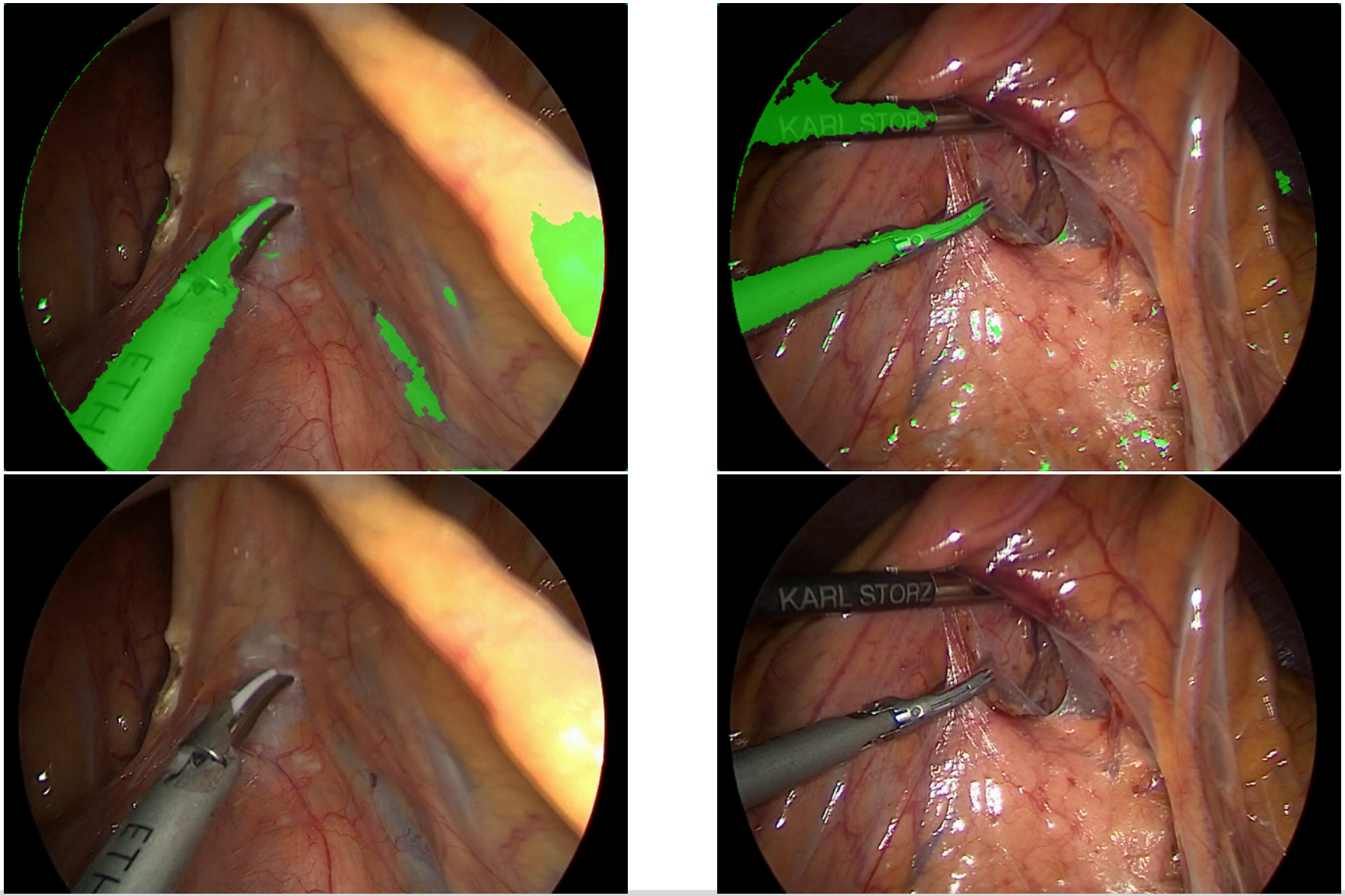
Ideen:

- Die Farbinformationen der umgebenden Pixel macht das Modell robuster.
- Morphologische Operationen (öffnen, schließen) machen die lokalen Segmentierungsergebnisse global konsistenter.

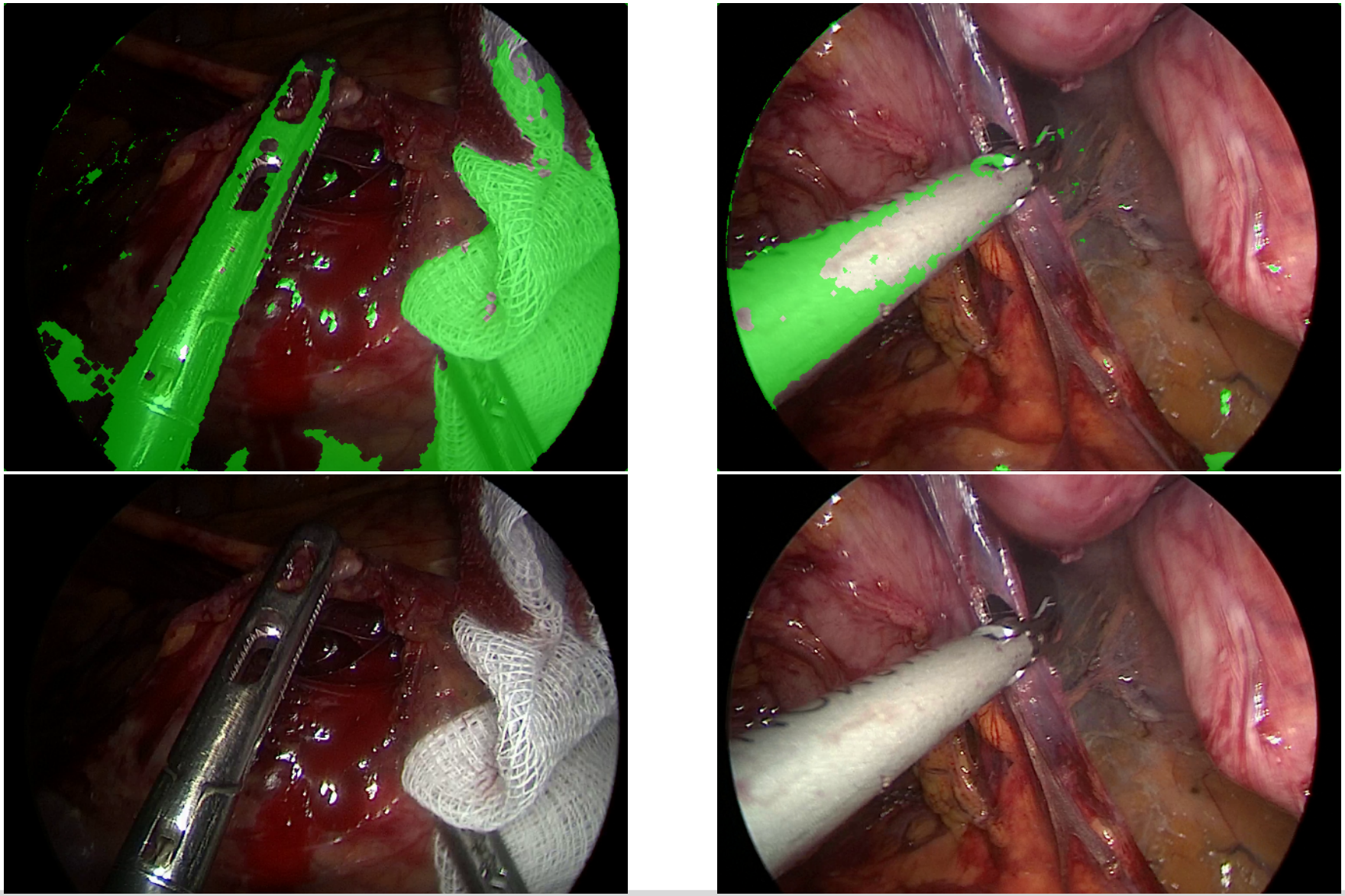
Model	Accuracy	Precision	Recall
Baseline	92.88 %	76.13 %	32.94 %
	(+8 %)	(+4 %)	(+11 %)
Local model + opening	93.42 %	76.98 %	40.65 %

Zur Interpretation: +8 % bedeutet, dass 8 % des Weges von der Baseline zum perfekten Ergebnis zurückgelegt wurden. Also $\frac{93.42-92.88}{100-92.88} \approx 8$.

Ergebnisse Local model + opening



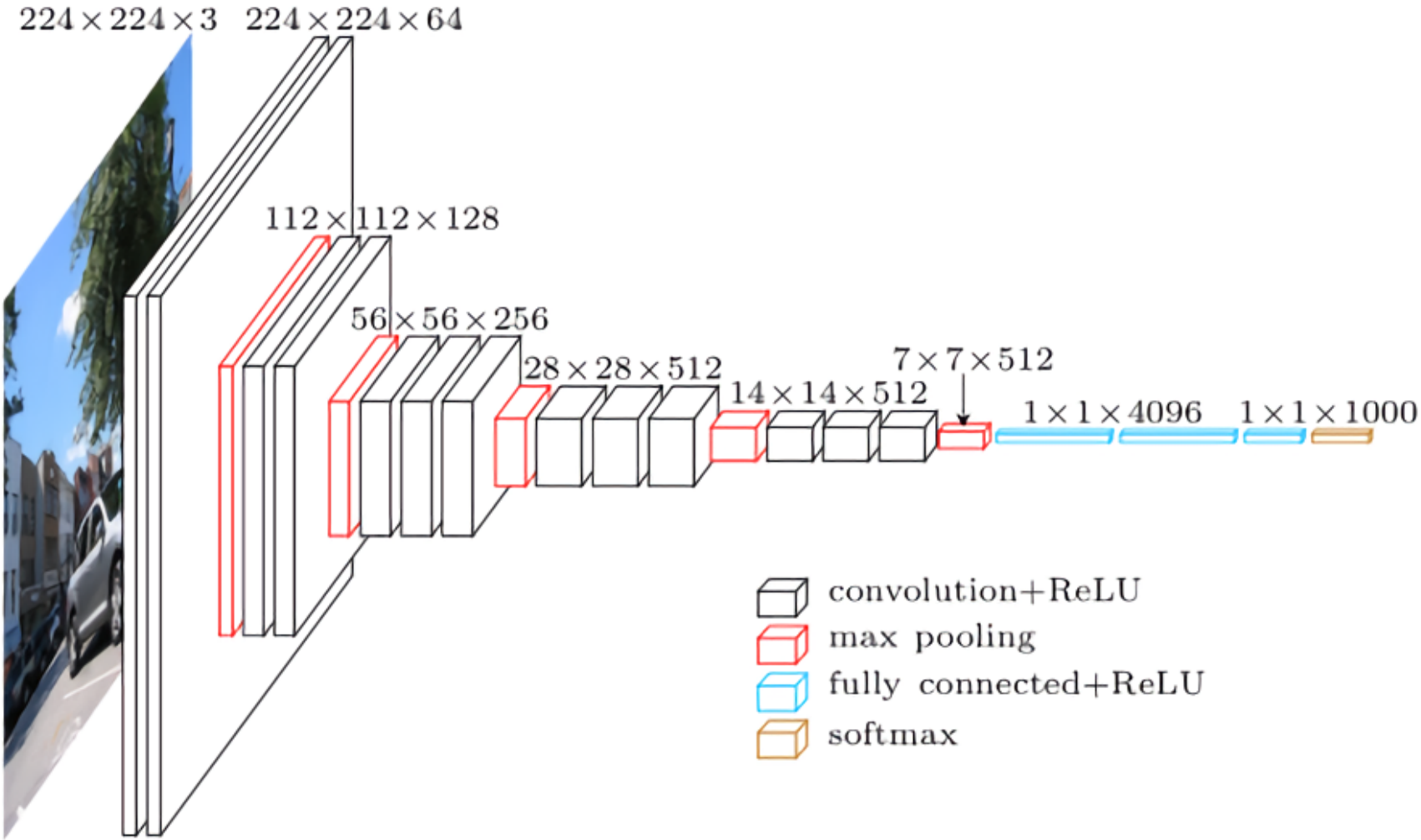
Ergebnisse Local model + opening



Fully Convolutional Networks (FCNs)

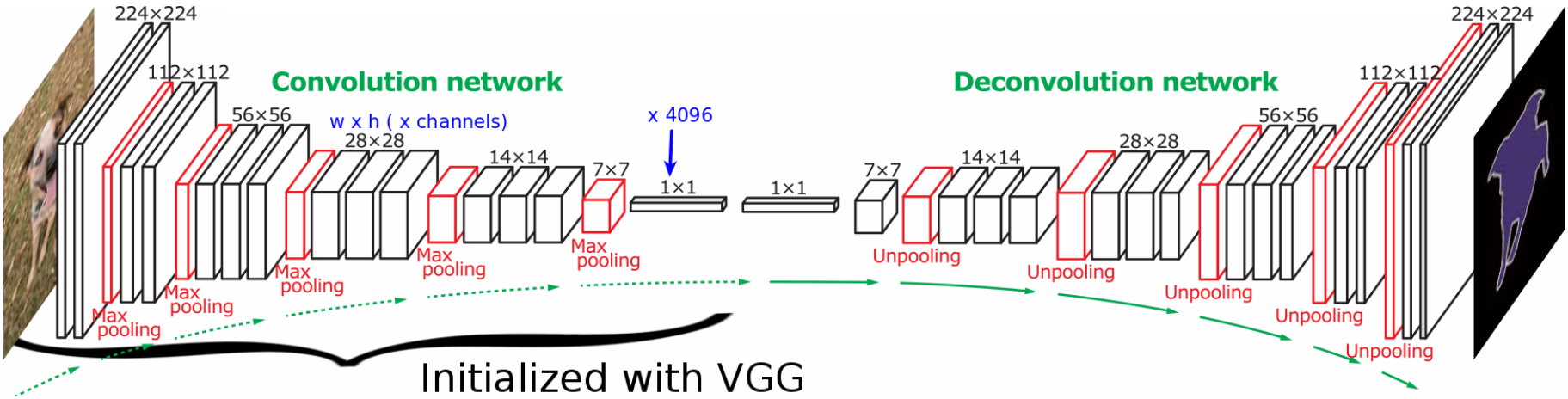
- Long et. al: Fully Convolutional Networks for Semantic Segmentation. 2014.
- Ideen:
  - Trainiere Netzwerk net auf $w \times h$ Bildern zur Klassifikation.
  - Interpretiere net als nicht-lineare Filter (CNN)
  - Wende Filter auf Bilder beliebiger Größe (min $w \times h$) mit stride $s \in \mathbb{N}_{\geq 1}$
  - Falls $s > 1$, dann mache upsampling.

FCN Klassifikationsnetz



VGG 16 Architektur

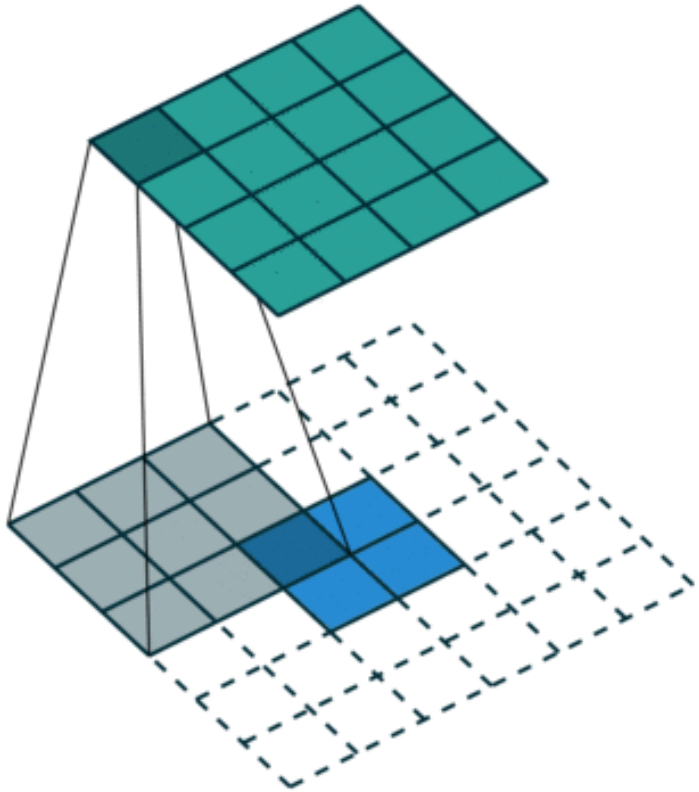
FCN Insgesamt



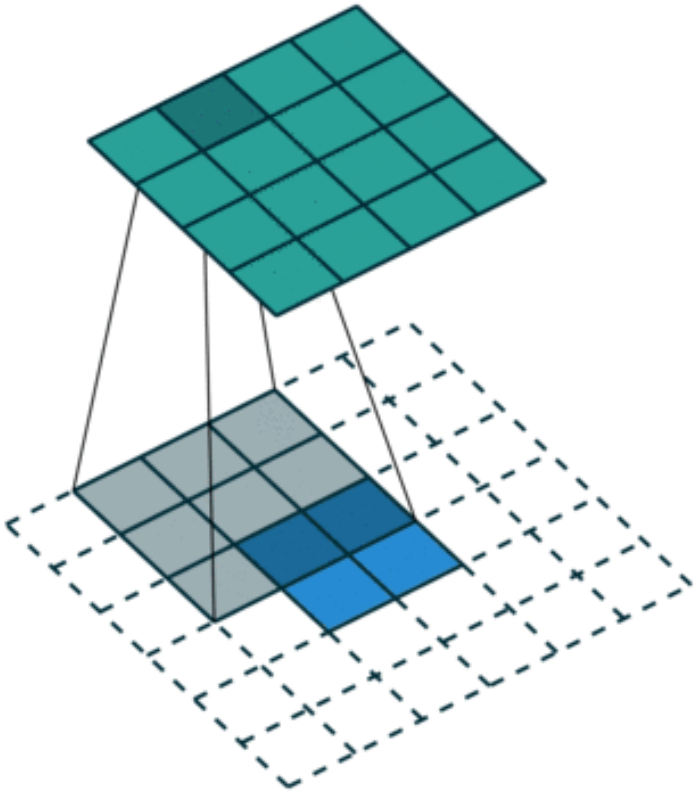
Upsampling

Sollte besser *transposed convolutional layer* genannt werden. Wird manchmal auch *deconvolutional layer* genannt.
Die folgenden Bilder stammen von dem Github user vdumoulin, github.com/vdumoulin/conv_arithmetic.

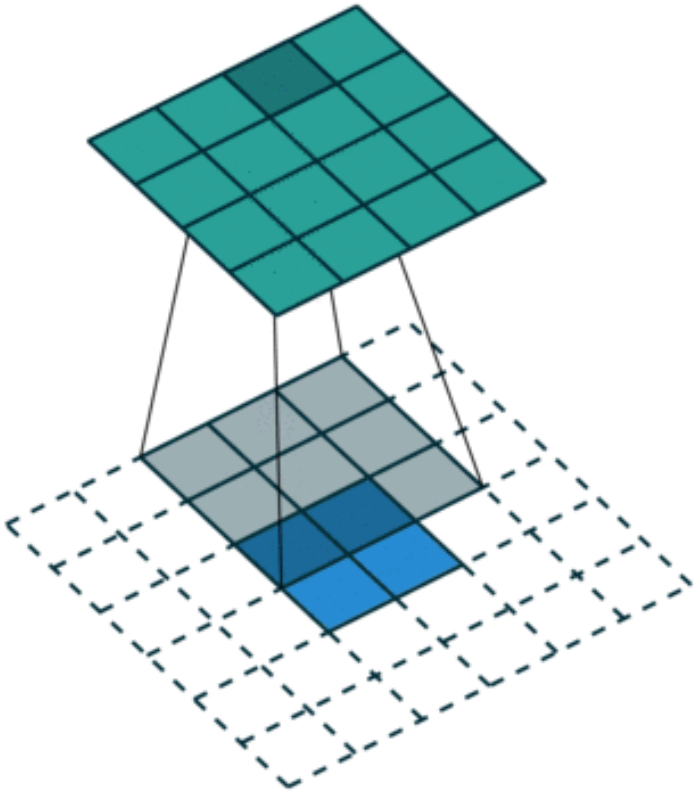
Upsampling 1: Padding außen



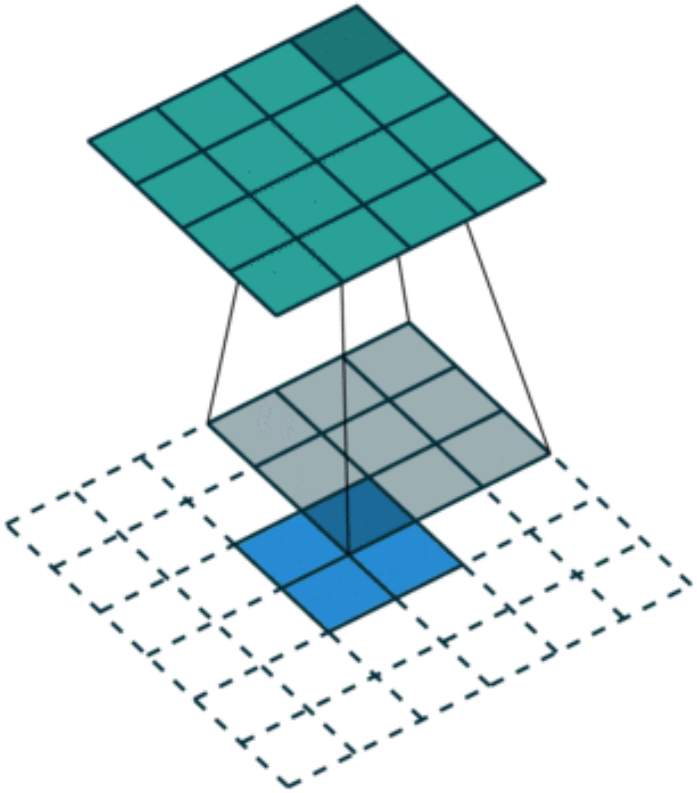
Upsampling 1: Padding außen



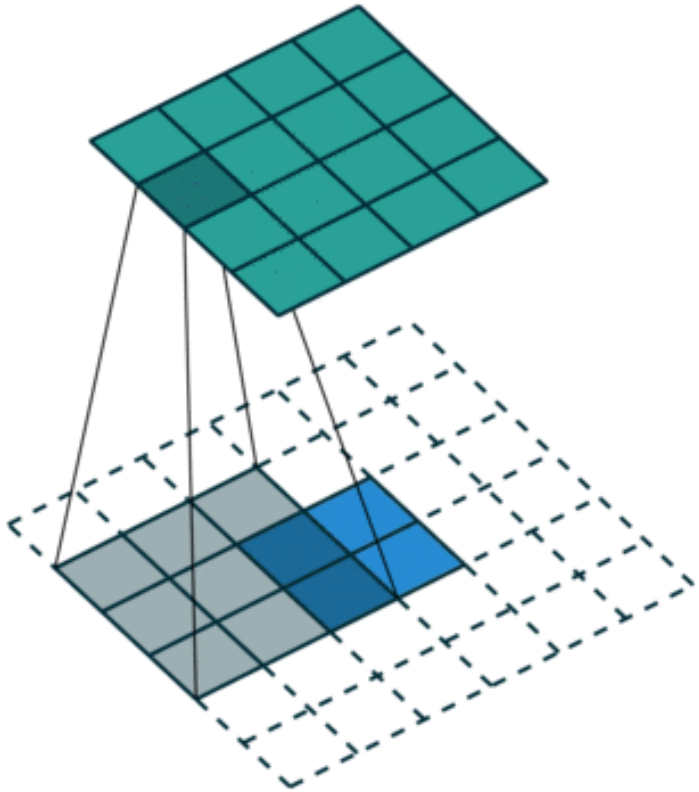
Upsampling 1: Padding außen



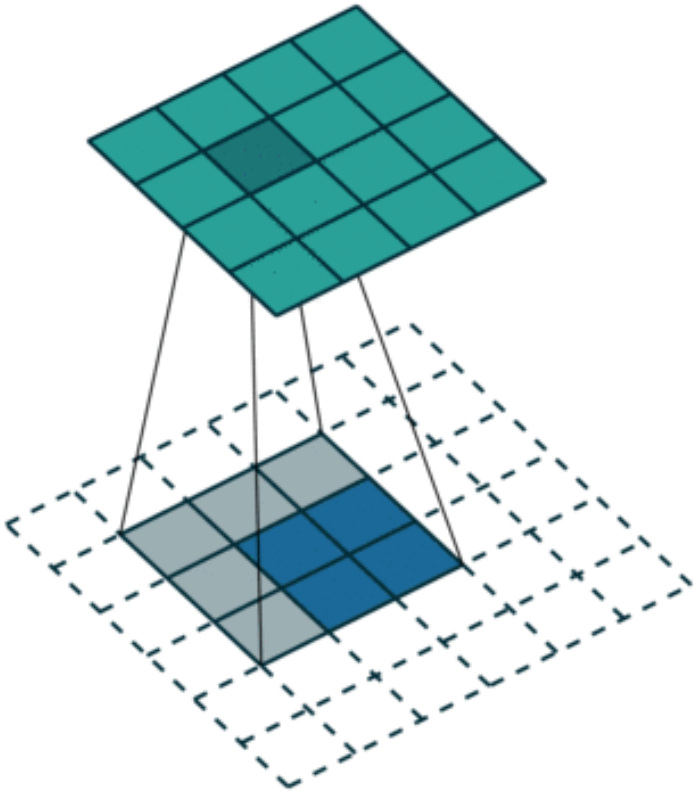
Upsampling 1: Padding außen



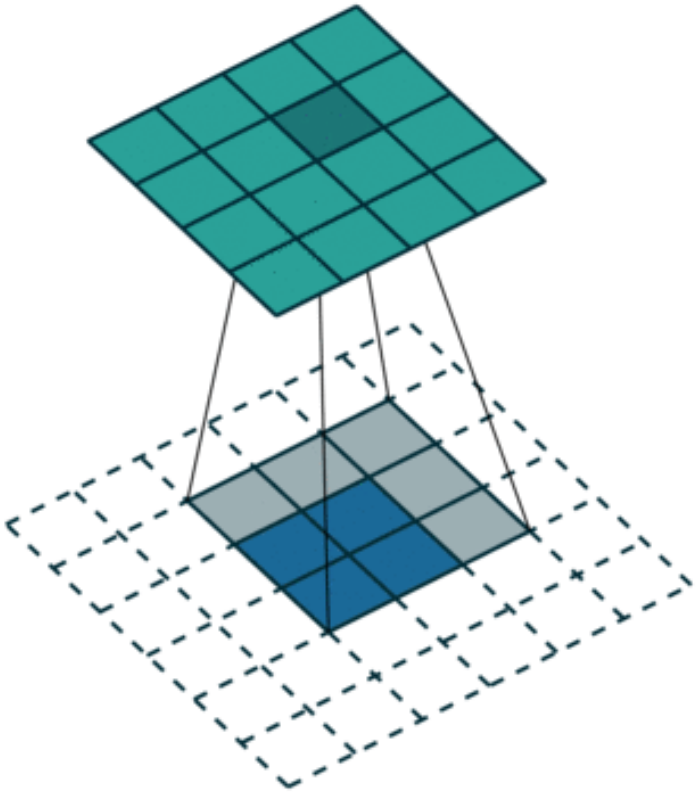
Upsampling 1: Padding außen



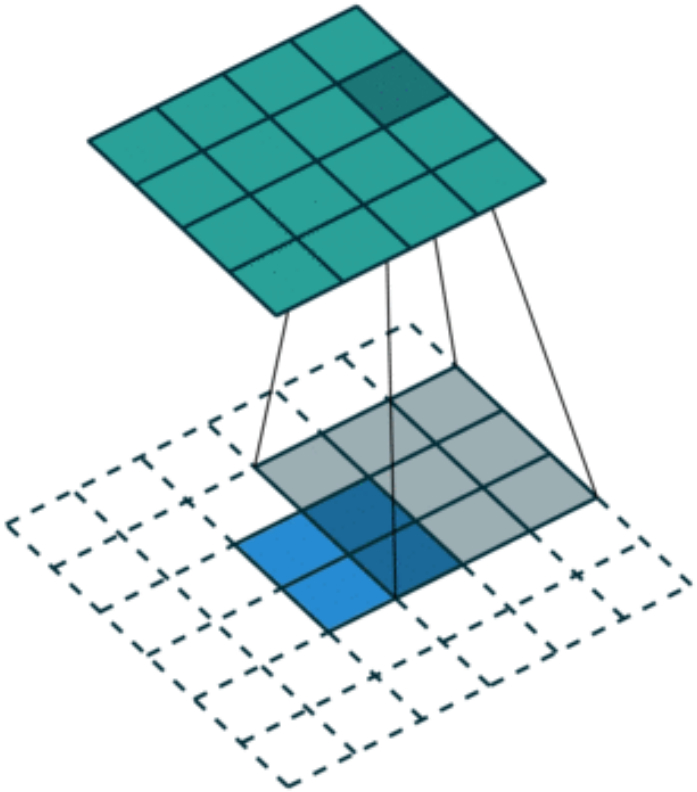
Upsampling 1: Padding außen



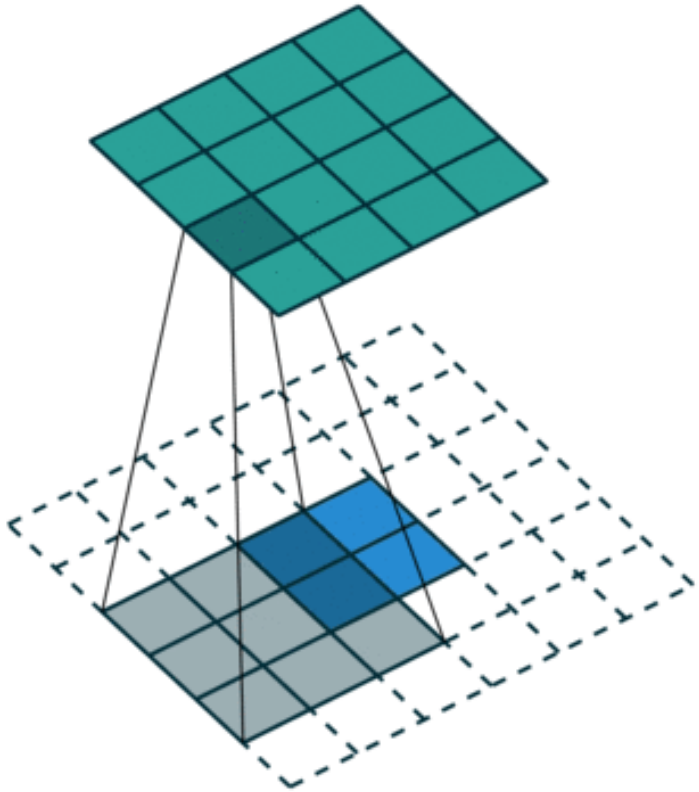
Upsampling 1: Padding außen



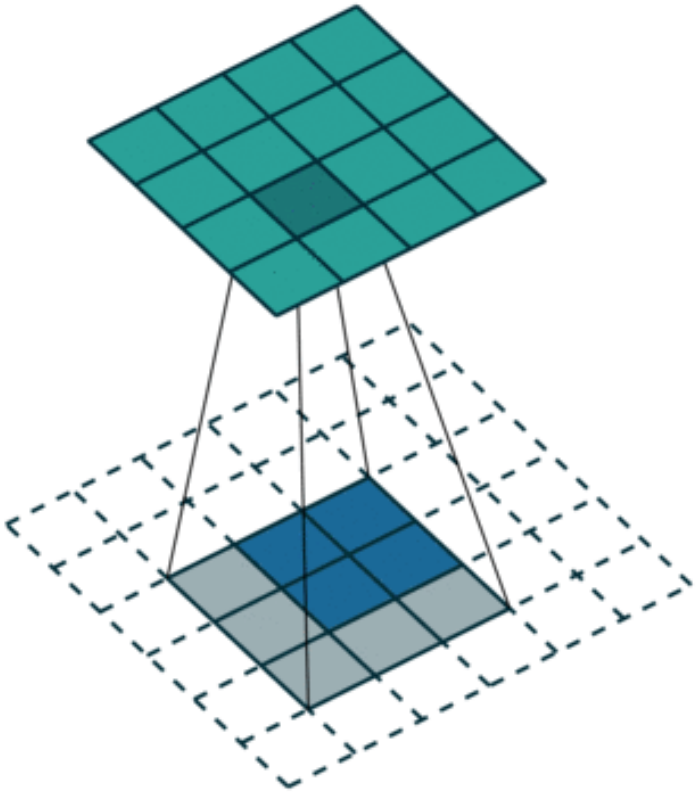
Upsampling 1: Padding außen



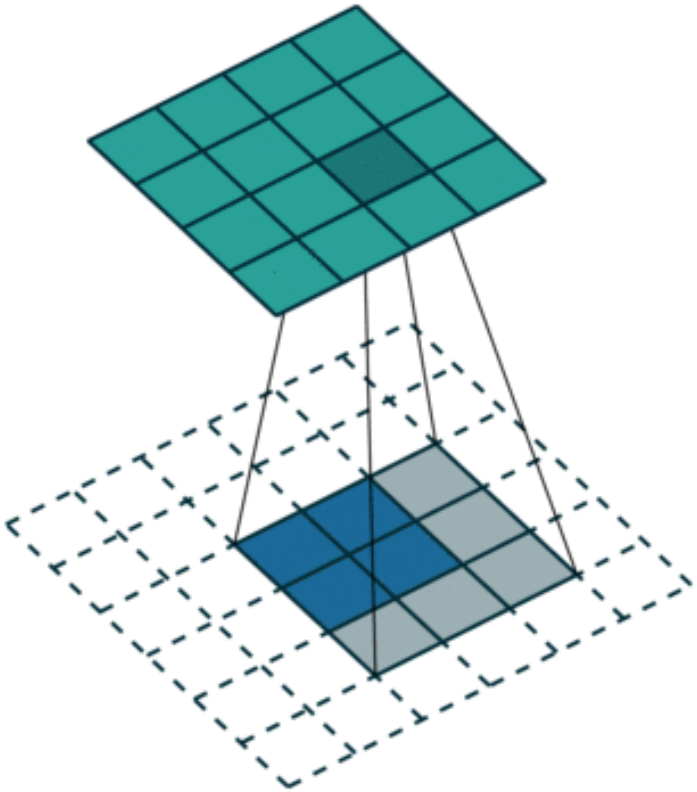
Upsampling 1: Padding außen



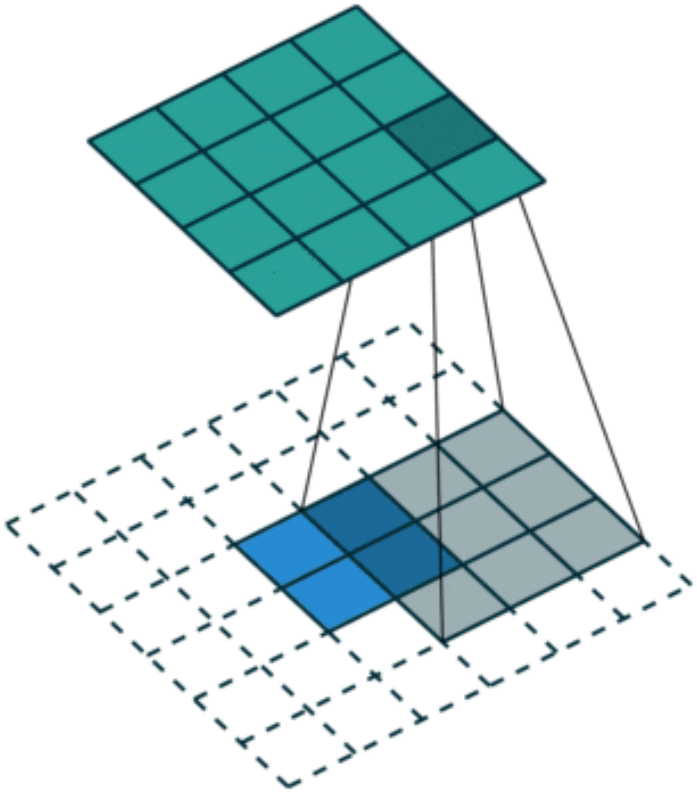
Upsampling 1: Padding außen



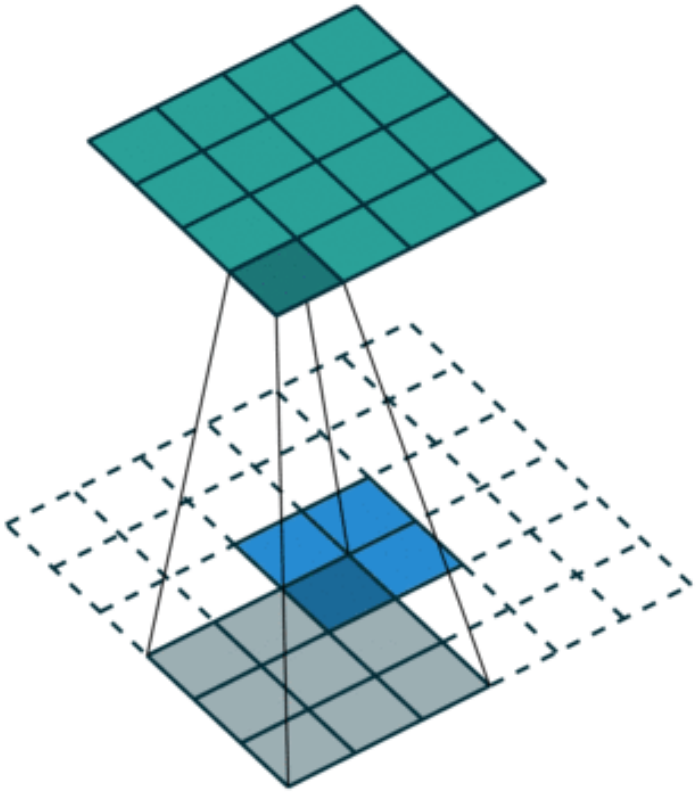
Upsampling 1: Padding außen



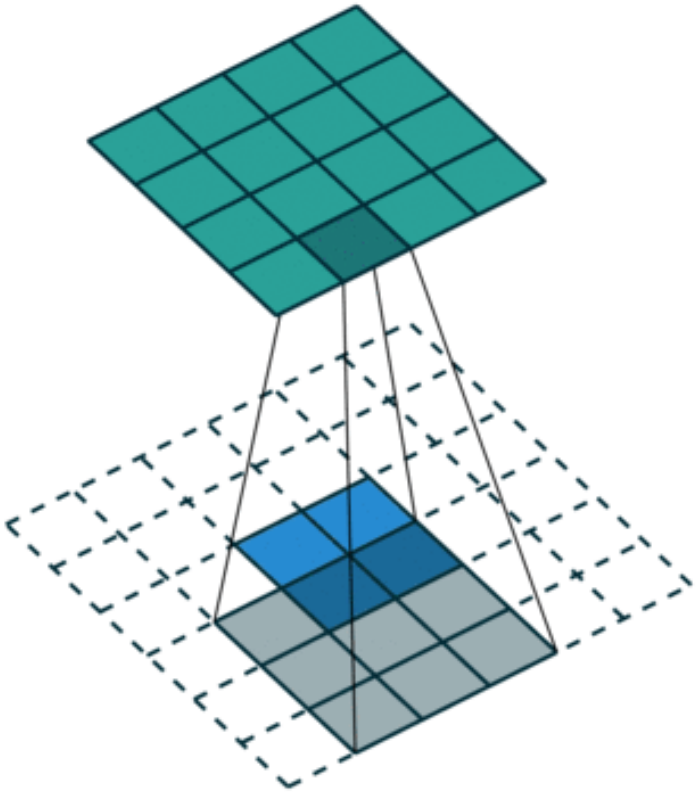
Upsampling 1: Padding außen



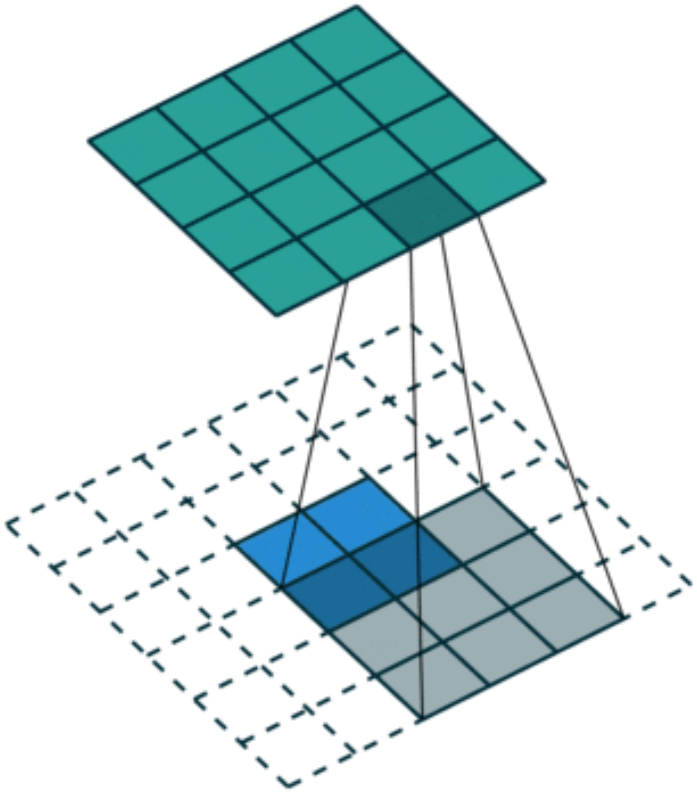
Upsampling 1: Padding außen



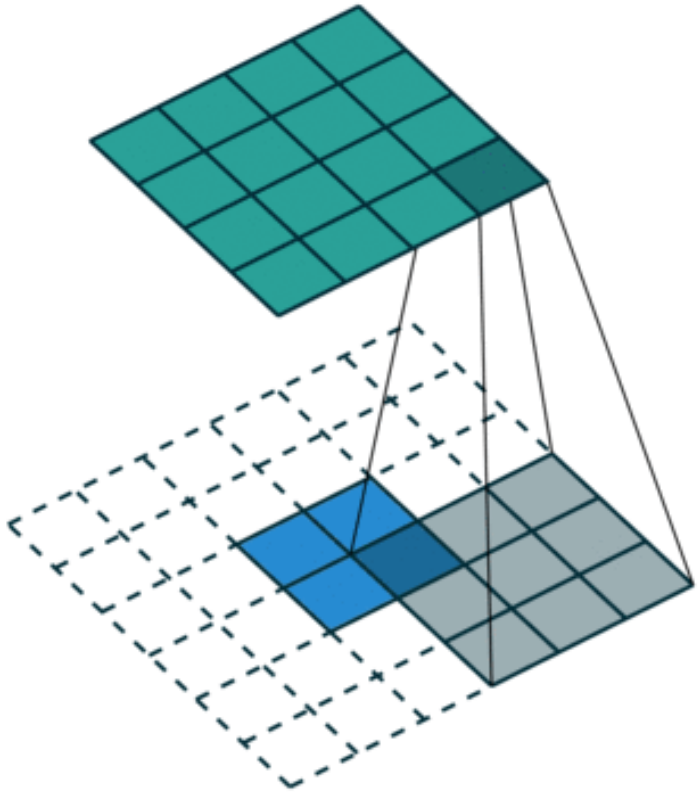
Upsampling 1: Padding außen



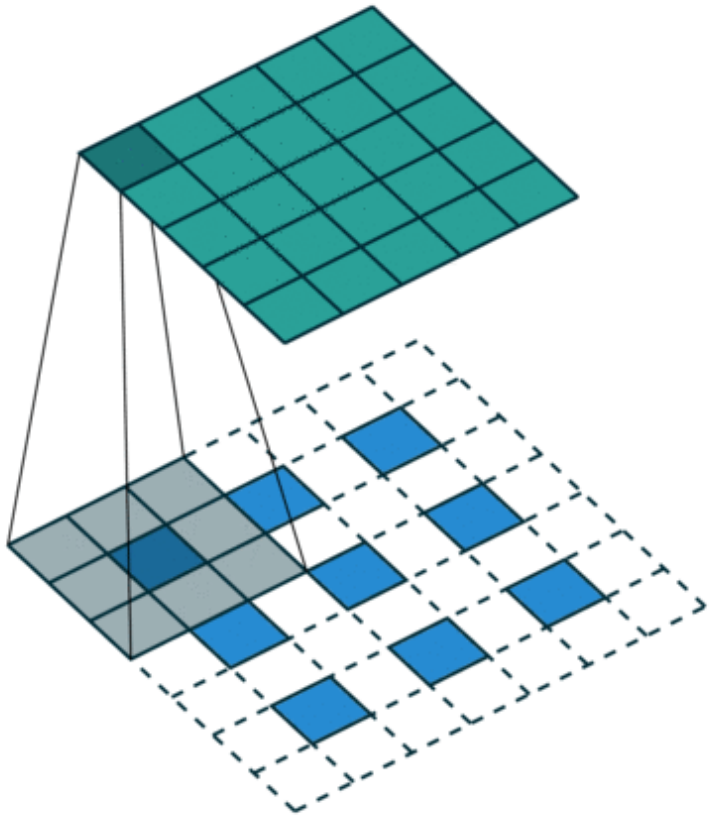
Upsampling 1: Padding außen



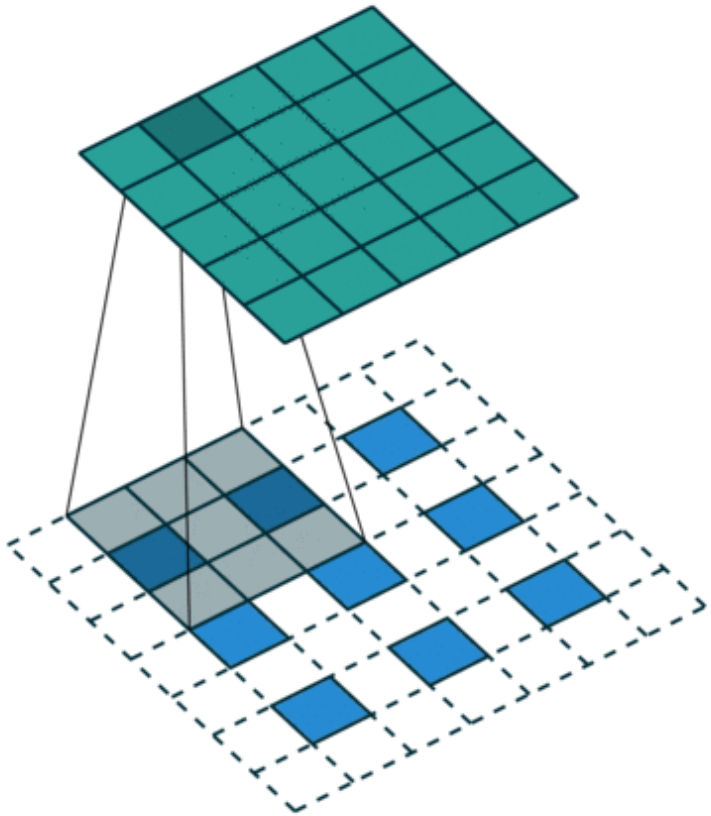
Upsampling 1: Padding außen



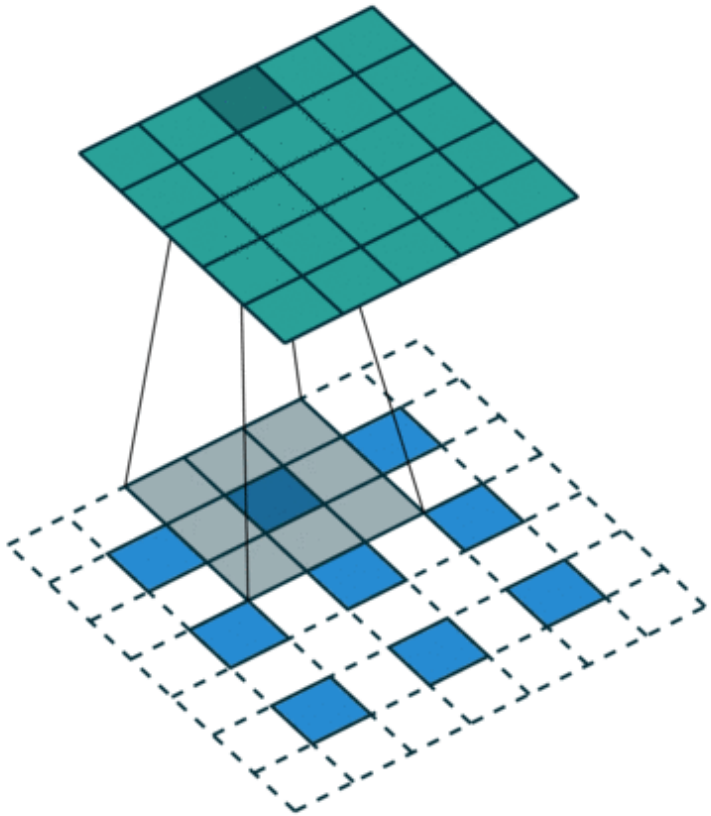
Upsampling 2: Padding um Pixel



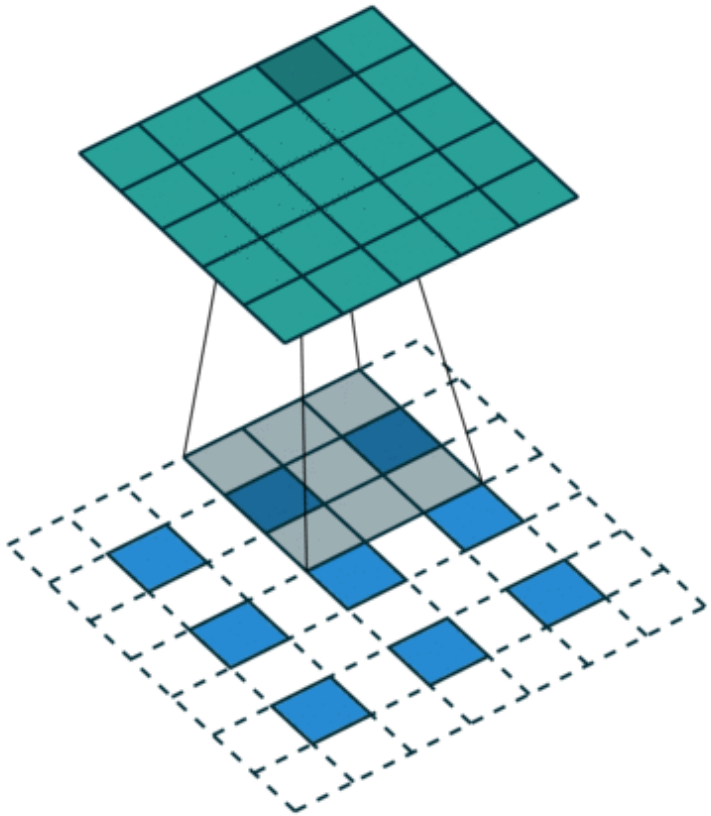
Upsampling 2: Padding um Pixel



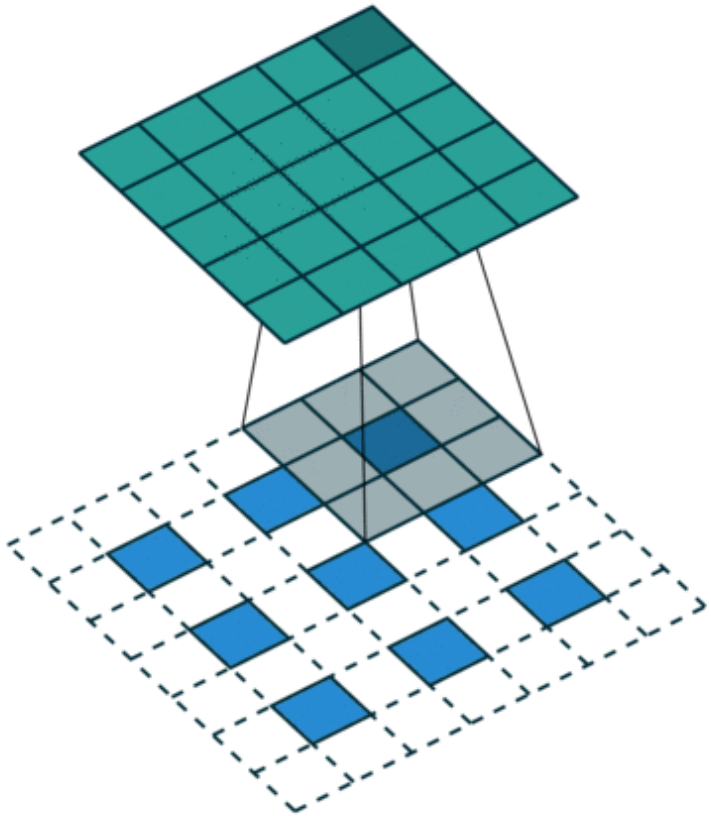
Upsampling 2: Padding um Pixel



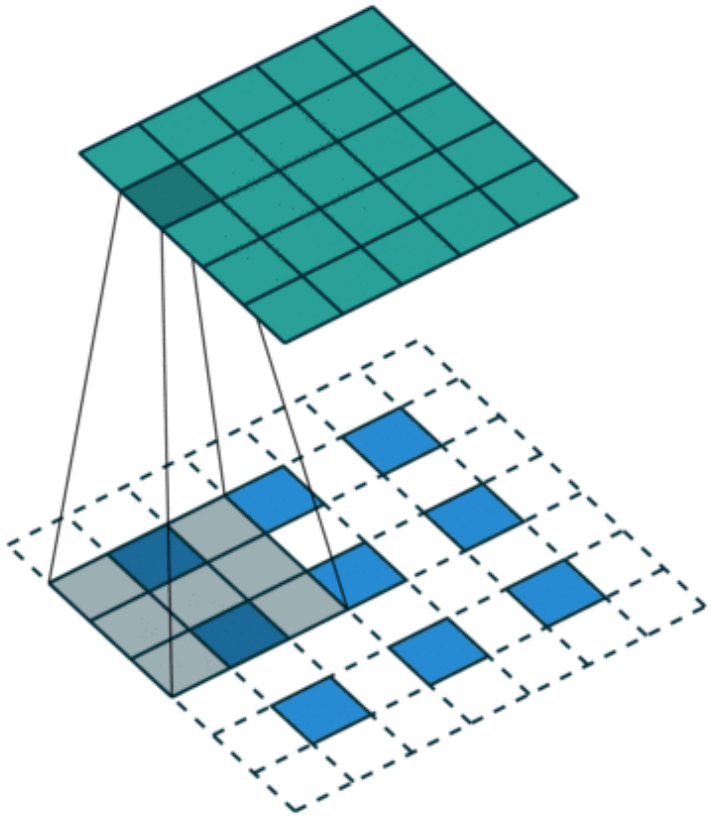
Upsampling 2: Padding um Pixel



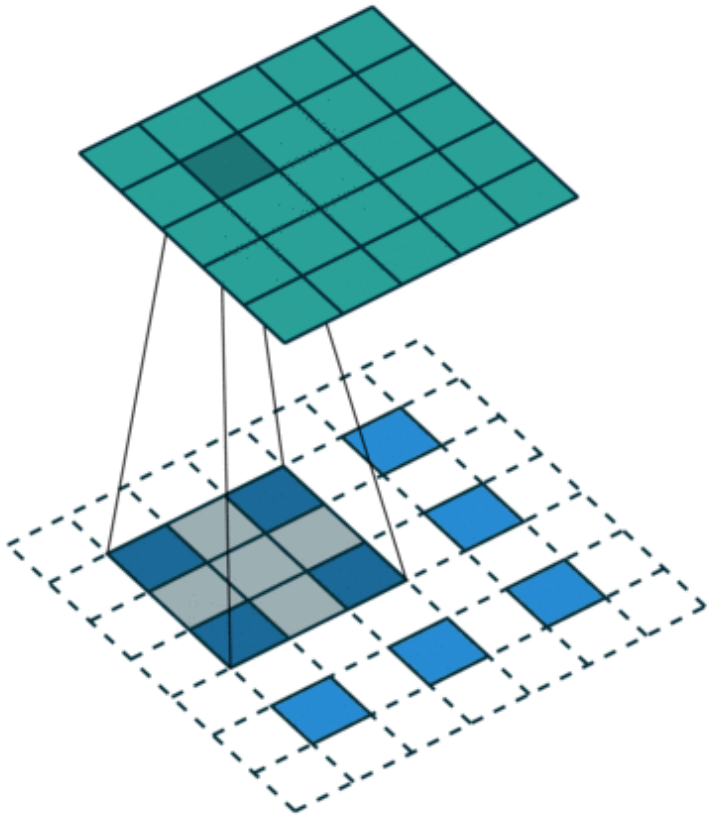
Upsampling 2: Padding um Pixel



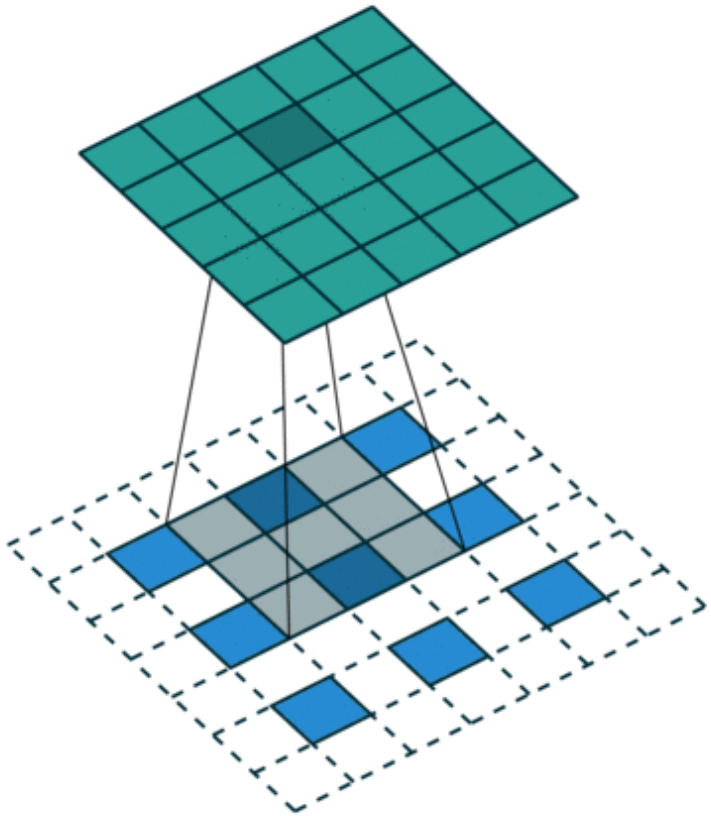
Upsampling 2: Padding um Pixel



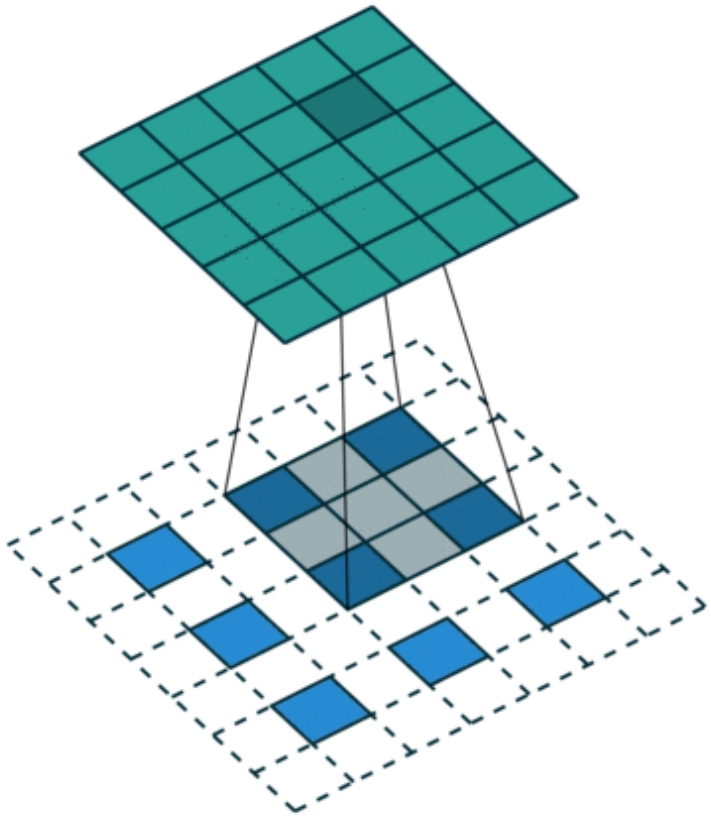
Upsampling 2: Padding um Pixel



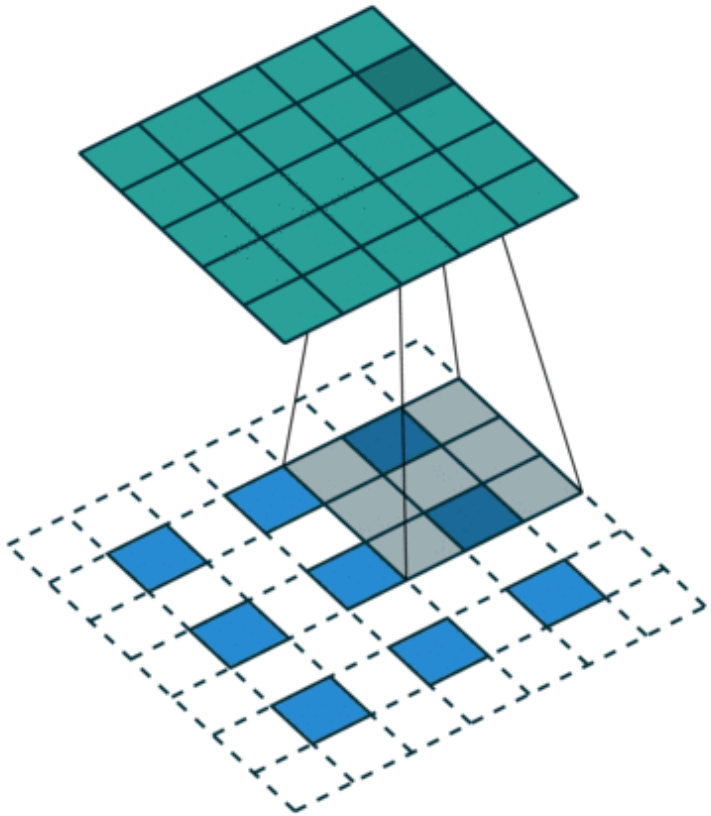
Upsampling 2: Padding um Pixel



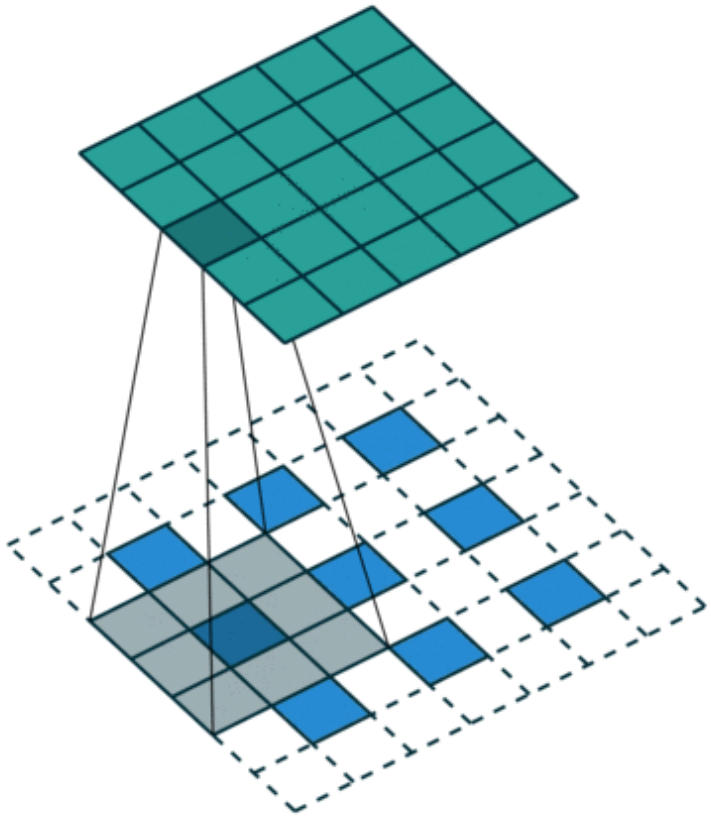
Upsampling 2: Padding um Pixel



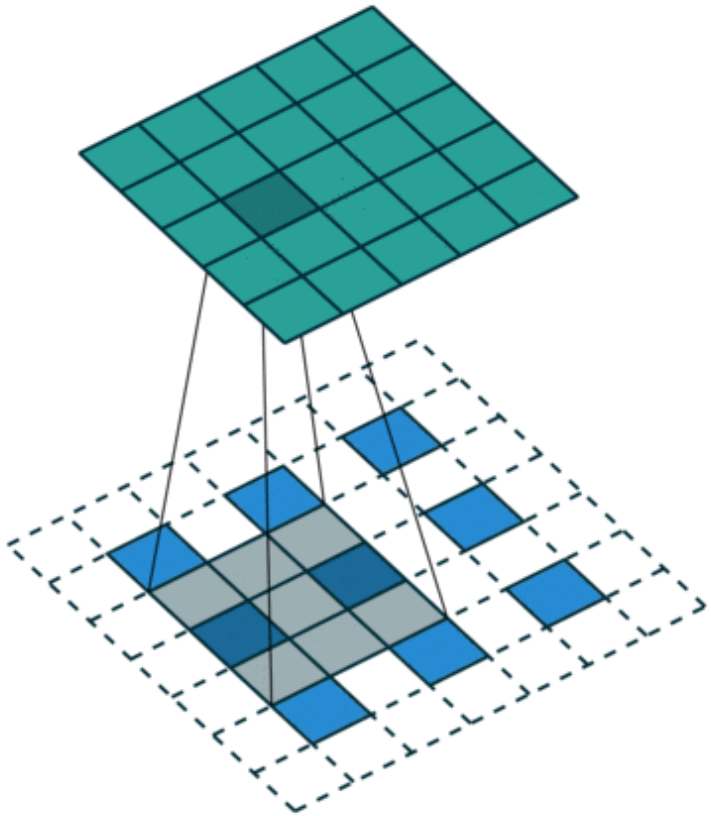
Upsampling 2: Padding um Pixel



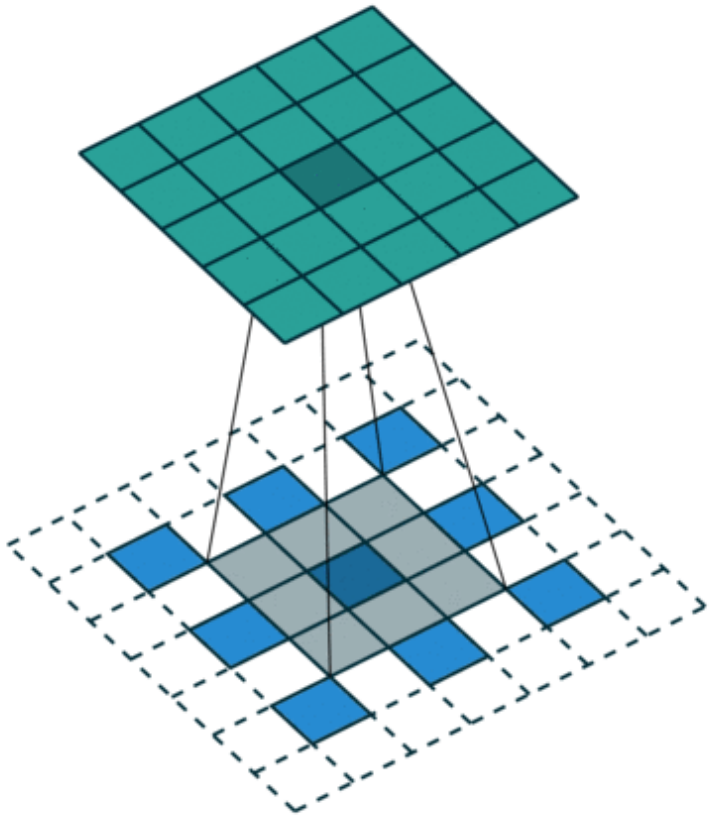
Upsampling 2: Padding um Pixel



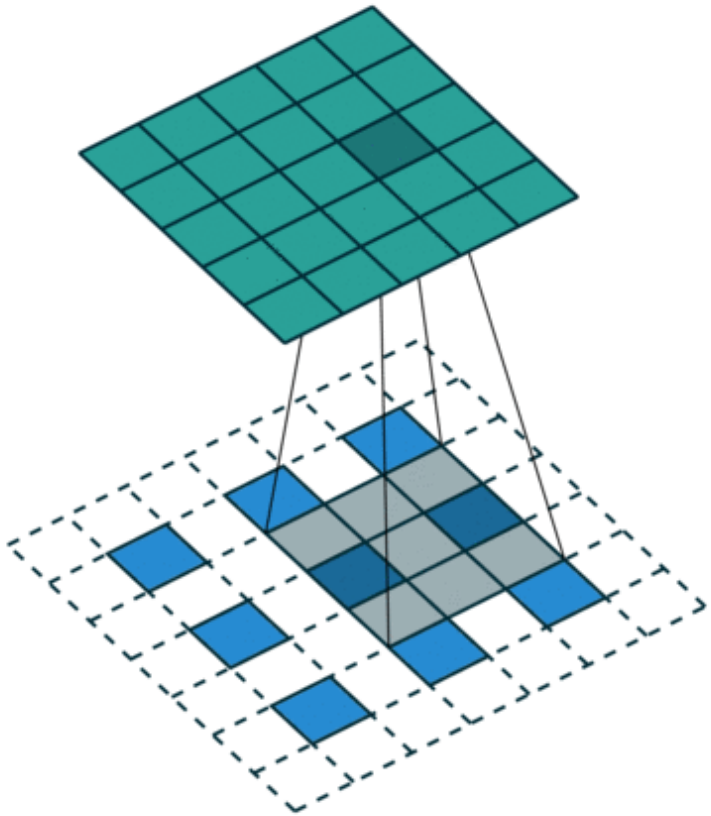
Upsampling 2: Padding um Pixel



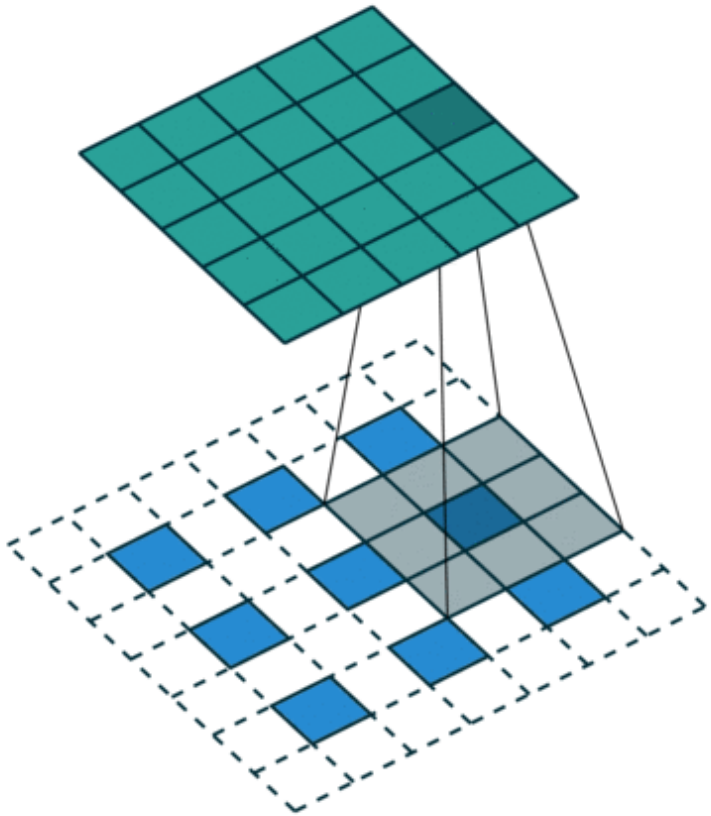
Upsampling 2: Padding um Pixel



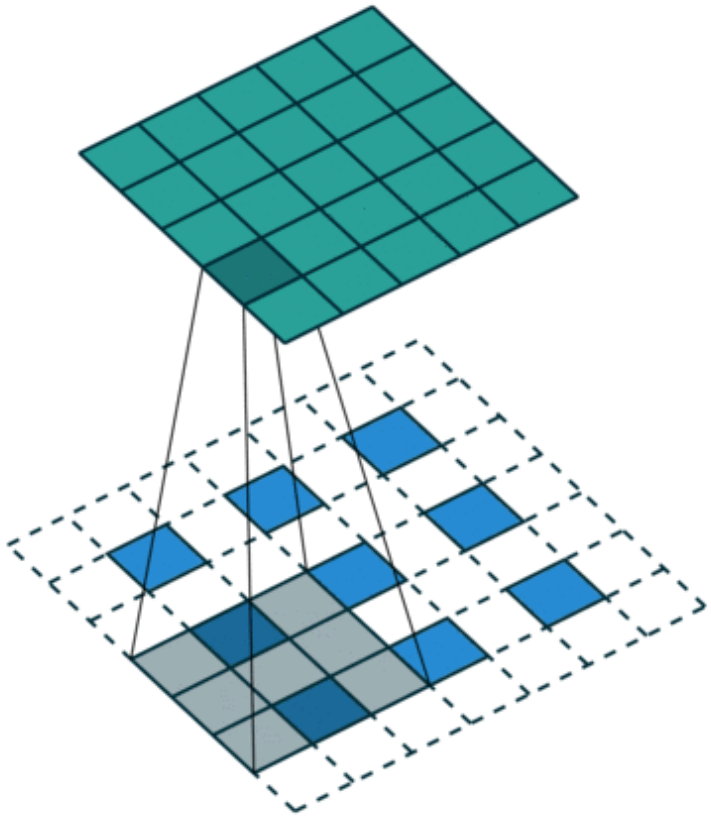
Upsampling 2: Padding um Pixel



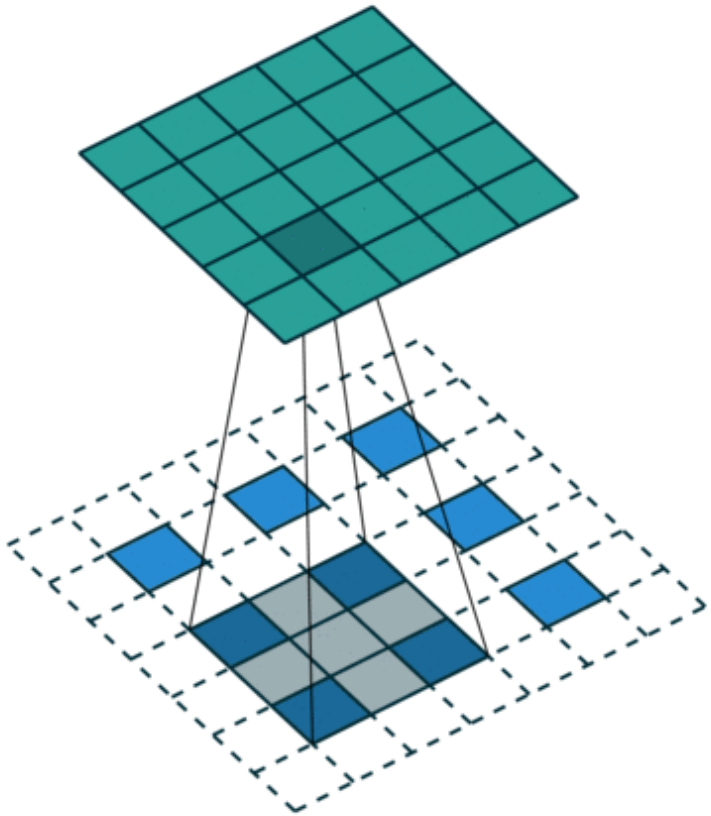
Upsampling 2: Padding um Pixel



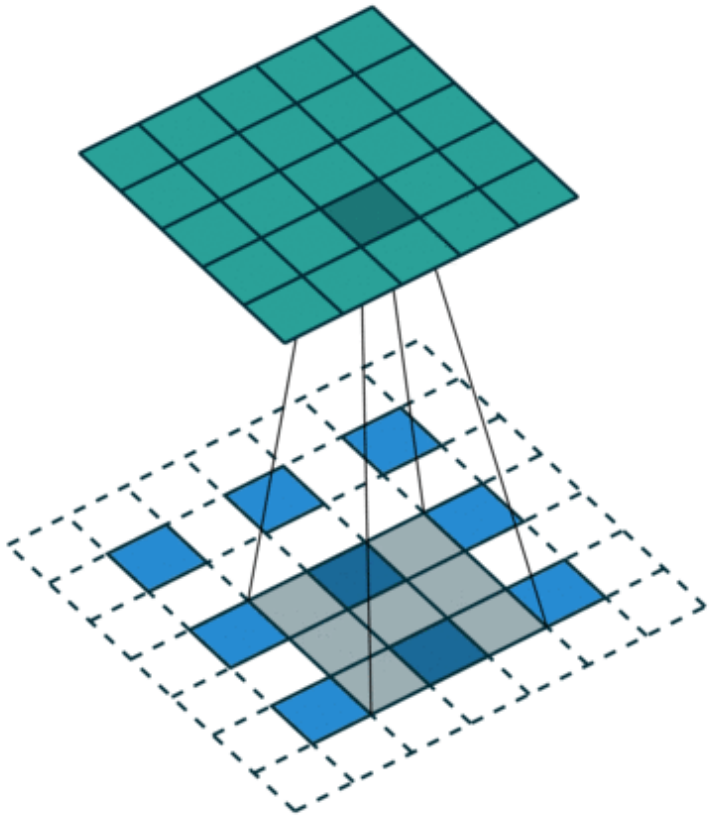
Upsampling 2: Padding um Pixel



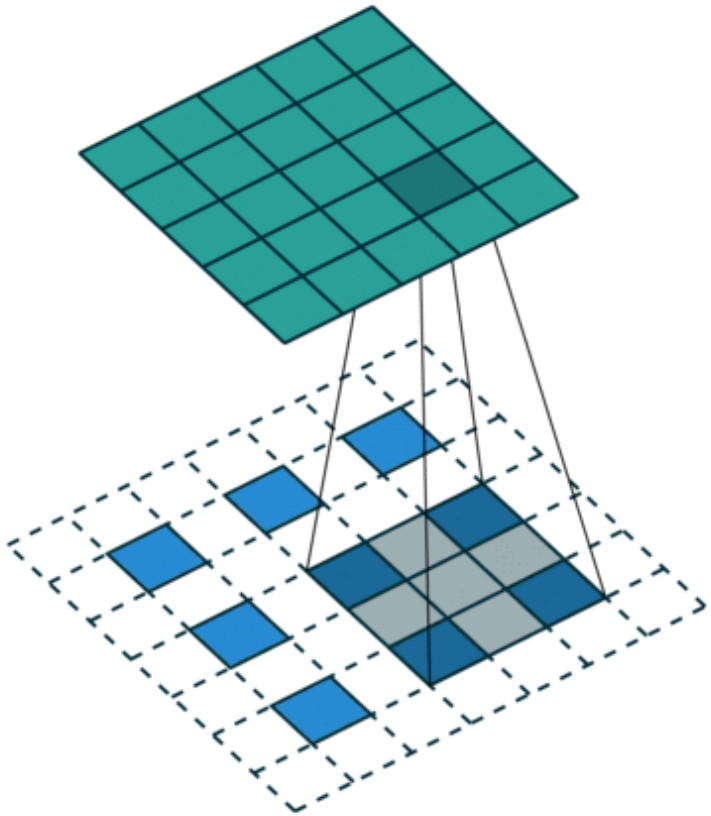
Upsampling 2: Padding um Pixel



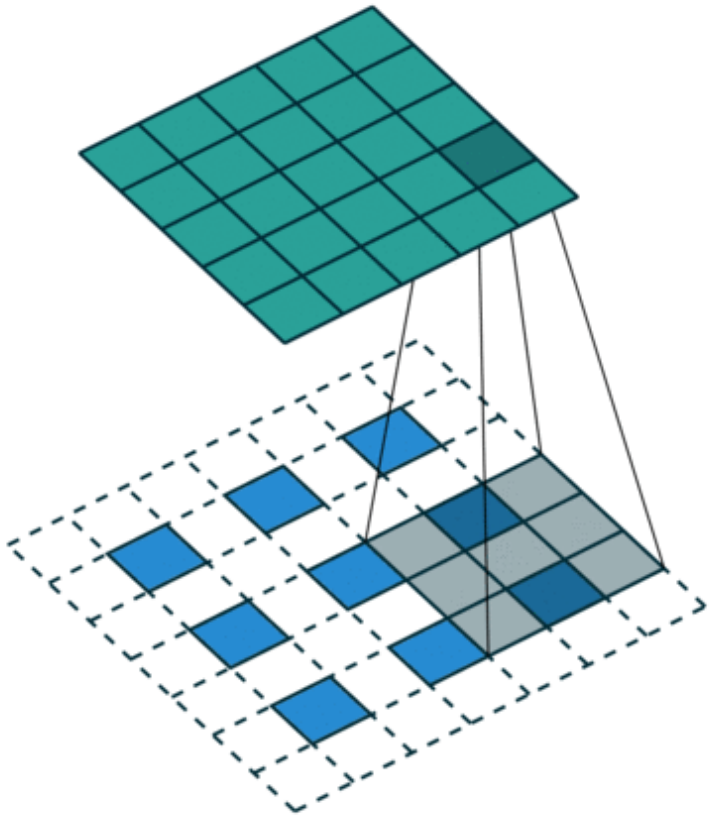
Upsampling 2: Padding um Pixel



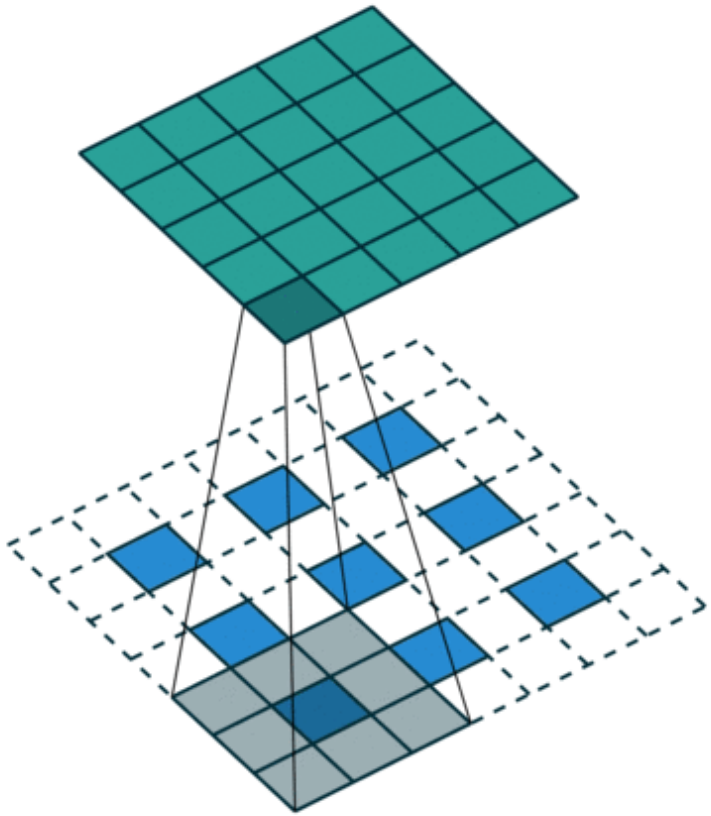
Upsampling 2: Padding um Pixel



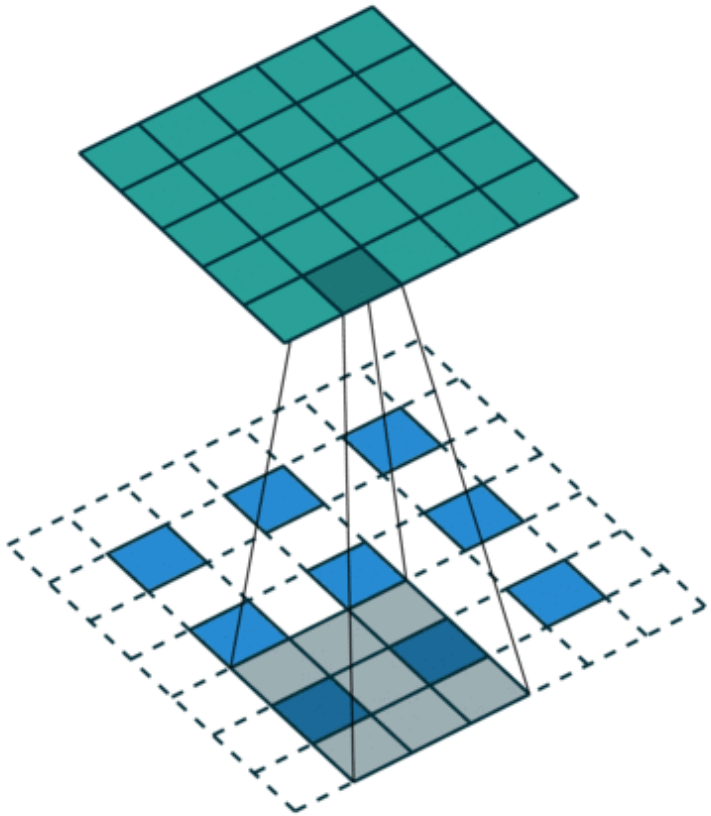
Upsampling 2: Padding um Pixel



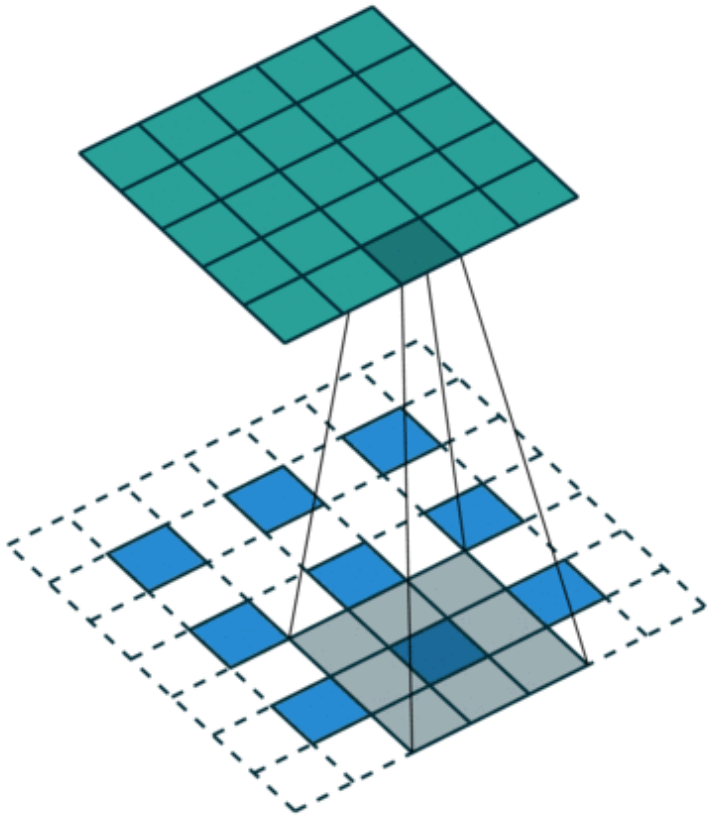
Upsampling 2: Padding um Pixel



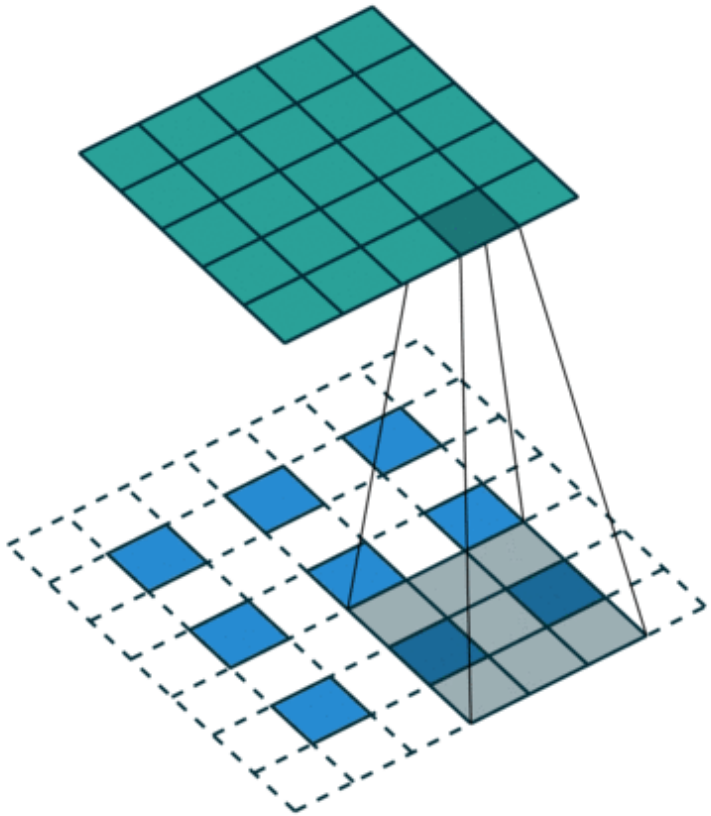
Upsampling 2: Padding um Pixel



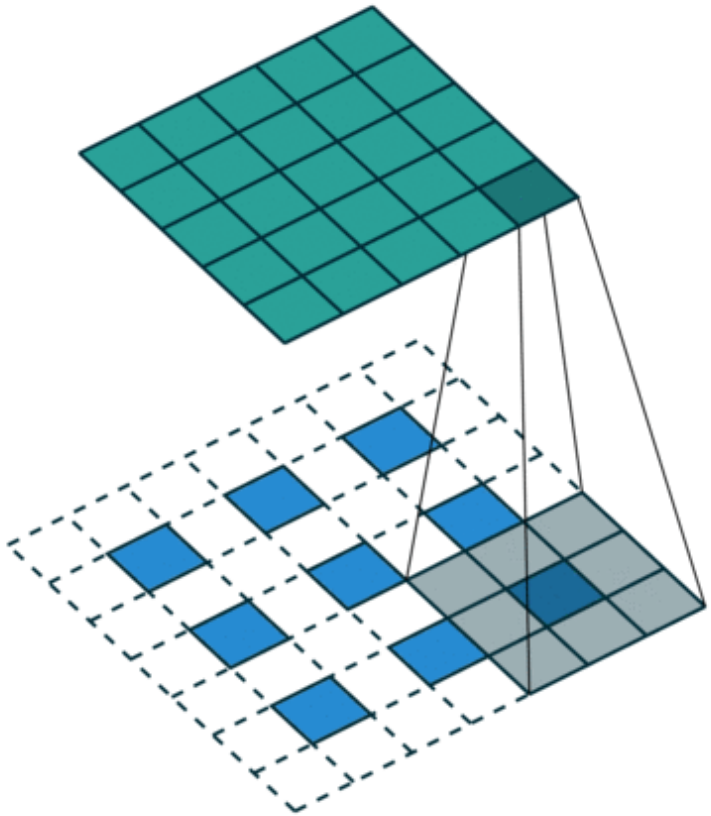
Upsampling 2: Padding um Pixel



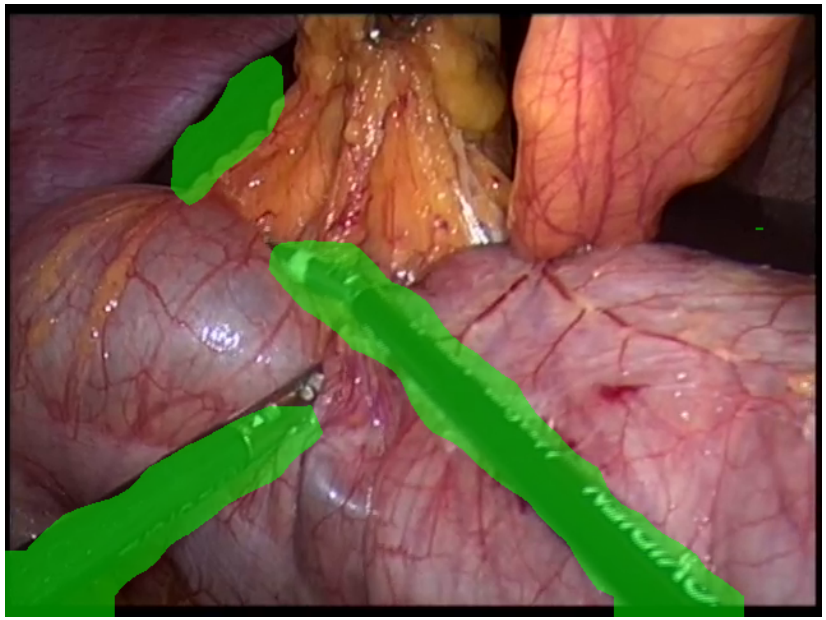
Upsampling 2: Padding um Pixel

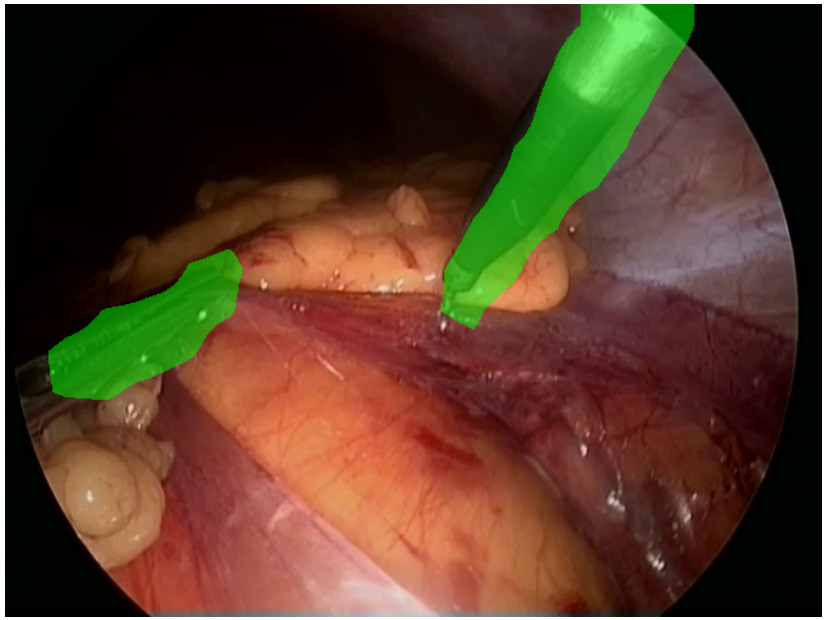


Upsampling 2: Padding um Pixel



FCN Ergebnisse





Model	Accuracy	Precision	Recall
Baseline	92.88 %	76.13 %	32.94 %
	(+8 %)	(+4 %)	(+11 %)
Local model + opening	93.42 %	76.98 %	40.65 %
	(+62 %)	(+41 %)	(+77 %)
FCN	97.32 %	85.87 %	84.90 %

FCN Ergebnisse

Model	Accuracy	Precision	Recall
Baseline	92.88 %	76.13 %	32.94 %
	(+8 %)	(+4 %)	(+11 %)
Local model + opening	93.42 %	76.98 %	40.65 %
	(+62 %)	(+41 %)	(+77 %)
FCN	**97.32** %	**85.87** %	84.90 %
KIT	93 %	69 %	46 %
Lausanne	96.5 %	76 %	**93** %

TensorVision

- Toolkit für Semantische Segmentierung mit Neuronalen Netzen
- Alpha-Version
- github.com/TensorVision/TensorVision

TensorVision Konfigurationsdateien

```
1  {
2    "model": {
3      "name": "model-301"
4    },
5
6    "model_nr": 1,
7
8    "solver": {
9        "epochs": 1,
10       "samples_per_epoch": 49152000,
11       "batch_size": 512
12   },
13
```

TensorVision Konfigurationsdateien

```
18   "data": {
19     "train": "../DATA/trainfiles.json",
20     "test": "../DATA/testfiles.json"
21   },
22
23   "classes": [
24     {"name": "background",
25      "colors": ["#000000"],
26      "output": "#ff000000",
27      "weight": 1},
28     {"name": "instrument",
29      "colors": ["#464646", "#a0a0a0", "#ffffff"],
30      "output": "#00ff007f",
31      "weight": 2}
32   ]
```

Ausblick

- Einfluss von Data Augmentation
- Pixel+Umgebung mit Data Augmentation
- FCN: Peak Memory Usage und Zeit für die Segmentierung
- Alternativen zu VGG 16
- Conditional Random Fields (CRFs)

Danke!

Gibt es Fragen?

Bildquellen

- VGG 16 architecture: D. Frossard. *VGG in TensorFlow*. 2016.
- FCN: H. Noh, S. Hong, B. Han. *Learning Deconvolution Network for Semantic Segmentation*. 2015.

Literatur

- M. Thoma: *A Survey of Semantic Segmentation*. 2016. arxiv.org/abs/1602.06541v2
- Jonathan Long, Evan Shelhamer, Trevor Darrell: *Fully Convolutional Networks for Semantic Segmentation*. 2014. https://arxiv.org/abs/1411.4038

Folien, LaTeXund Material

Der Foliensatz sowie die LaTeXund Ti*k*Z-Quellen sind unter github.com/TensorVision/MediSeg/tree/master/AP8

Kurz-URL: tinyurl.com/medsim-segmentation