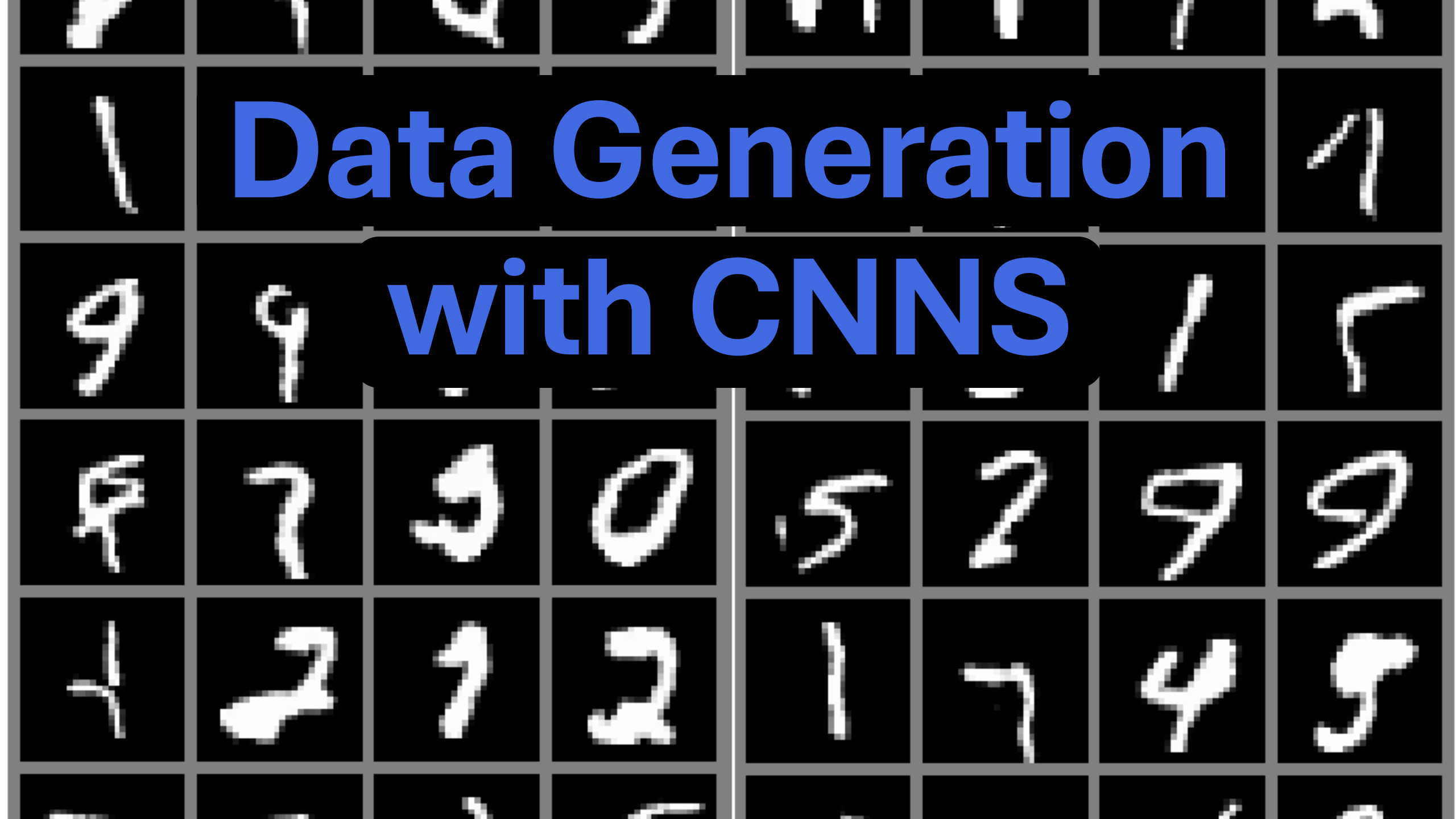
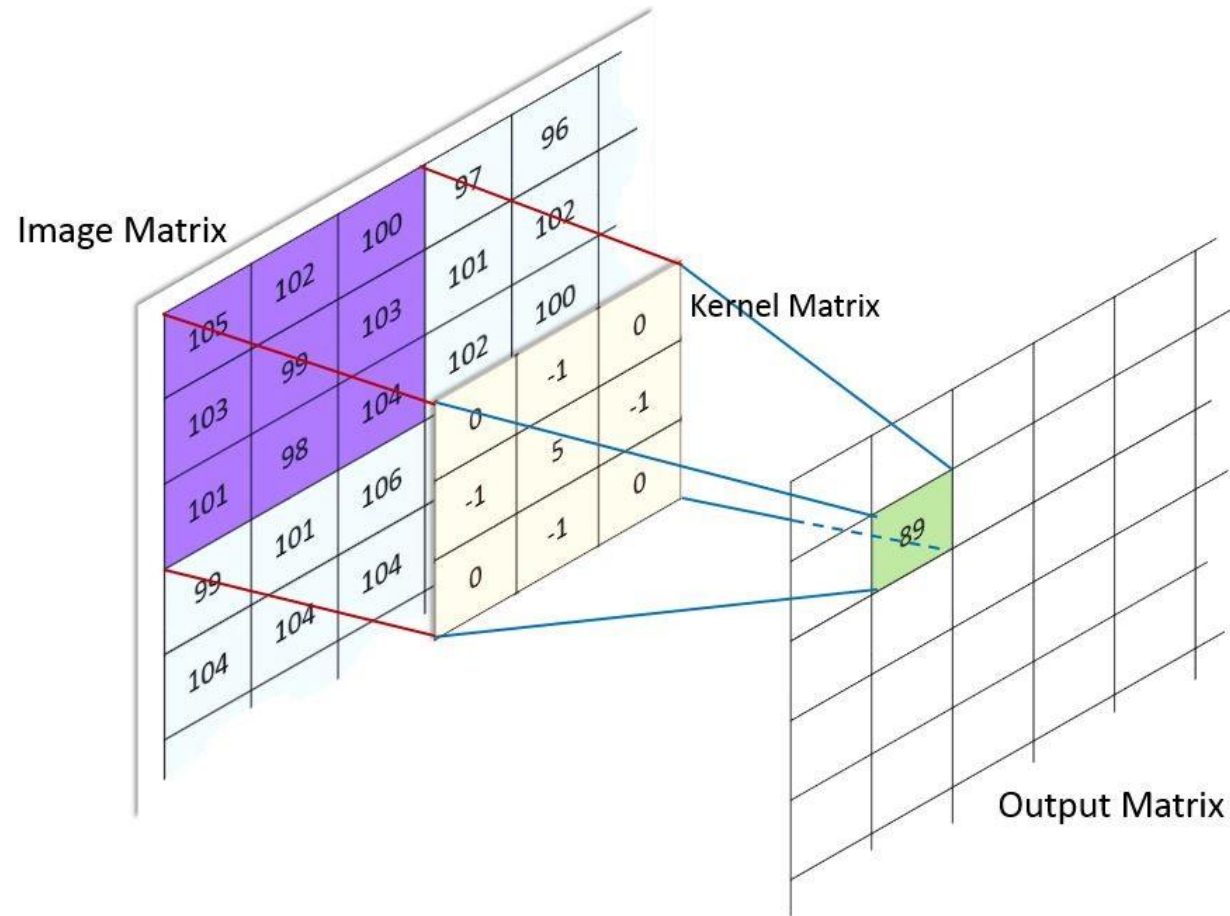


The background of the image is a 10x10 grid of handwritten digits from the MNIST dataset. The digits are white on a black background and are arranged in a grid. The central text is overlaid on the grid. The text is in a bold, blue, sans-serif font. The first line of text is "Data Generation" and the second line is "with CNNs".

Data Generation with CNNs

Convolutions



$$\begin{aligned} &105 * 0 + 102 * -1 + 100 * 0 + \\ &103 * -1 + 99 * 5 + 103 * -1 + \\ &101 * 0 + 98 * -1 + 104 * 0 = 89 \end{aligned}$$

Padding

0	0	0	0	0	0	0	0
0	3	3	4	4	7	0	0
0	9	7	6	5	8	2	0
0	6	5	5	6	9	2	0
0	7	1	3	2	7	8	0
0	0	3	7	1	8	3	0
0	4	0	4	3	2	2	0
0	0	0	0	0	0	0	0

$6 \times 6 \rightarrow 8 \times 8$

<https://datahacker.rs/what-is-padding-cnn/>

*

1	0	-1
1	0	-1
1	0	-1

3×3

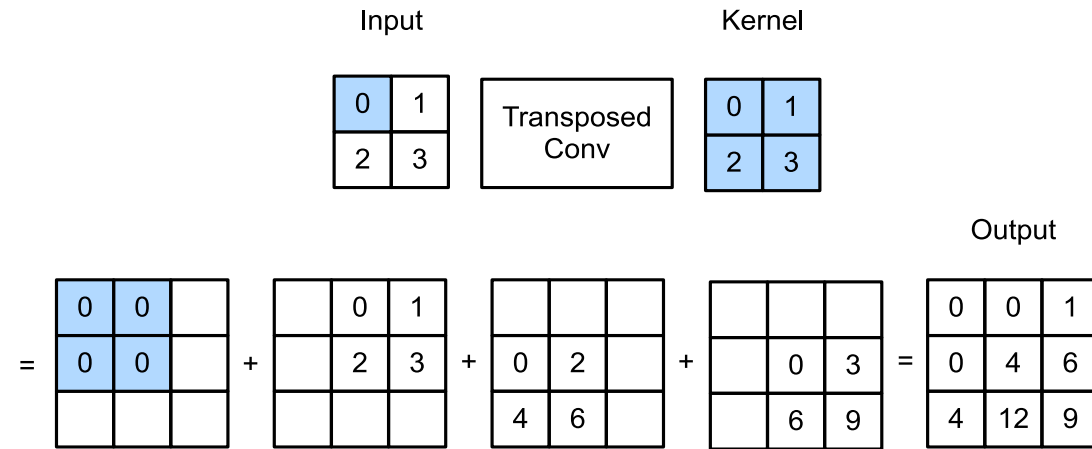
=

-10	-13	1			
-9	3	0			

6×6

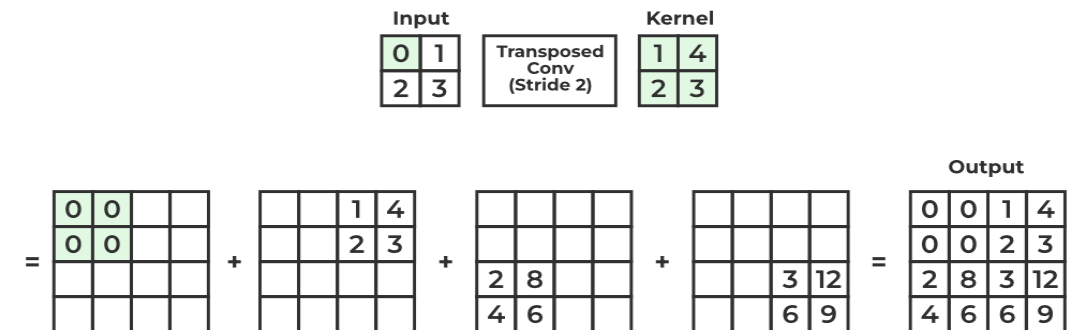
- Filter passt nicht auf jedes Element -> kleinerer Output
- Padding erweitert Input, um Form zu bewahren

Transposed Convolution



$$\text{Output} = (\text{Input} - 1) * \text{Stride} + \text{Kernel} - 2 * \text{Padding}$$

https://d2l.ai/chapter_computer-vision/transposed-conv.html

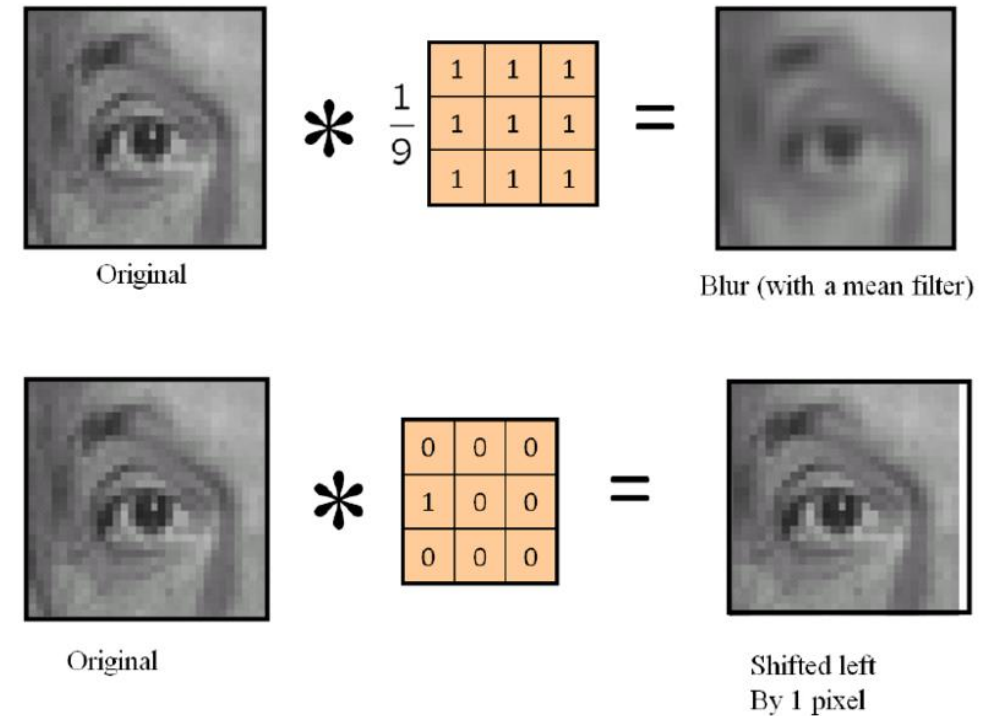


<https://www.geeksforgeeks.org/machine-learning/what-is-transposed-convolutional-layer/>

Verschiedene Convolutions

<i>Original</i>	<i>Gaussian Blur</i>	<i>Sharpen</i>	<i>Edge Detection</i>
$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$
			

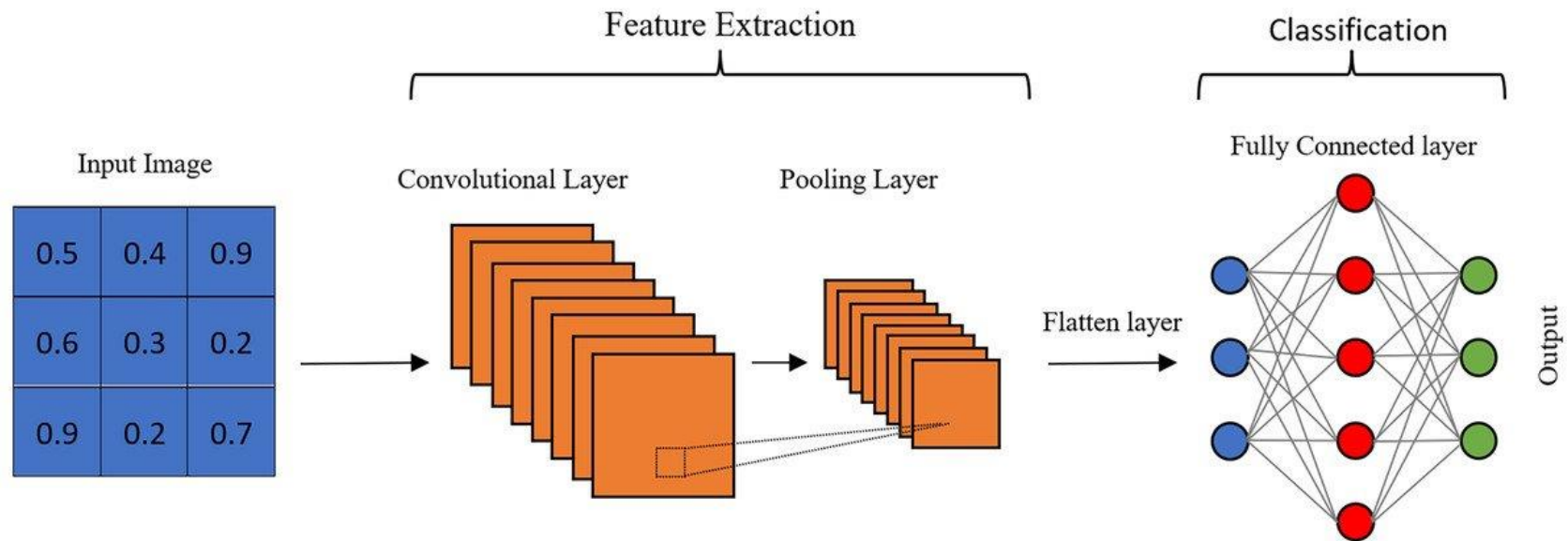
<https://medium.com/@abhishekjainindore24/all-about-convolutions-kernels-features-in-cnn-c656616390a1>



<https://blog.naver.com/framkang/220561249726>

Convolutional Neural Network

- Convolutions erkennen lokale Zusammenhänge und Strukturen
- Kernel durch Backpropagation angepasst



Sichtweite erhöhen

Erste Schicht

2	1	1	0	1
2	0	2	1	1
0	1	0	2	0
0	2	2	0	2
2	1	2	0	2

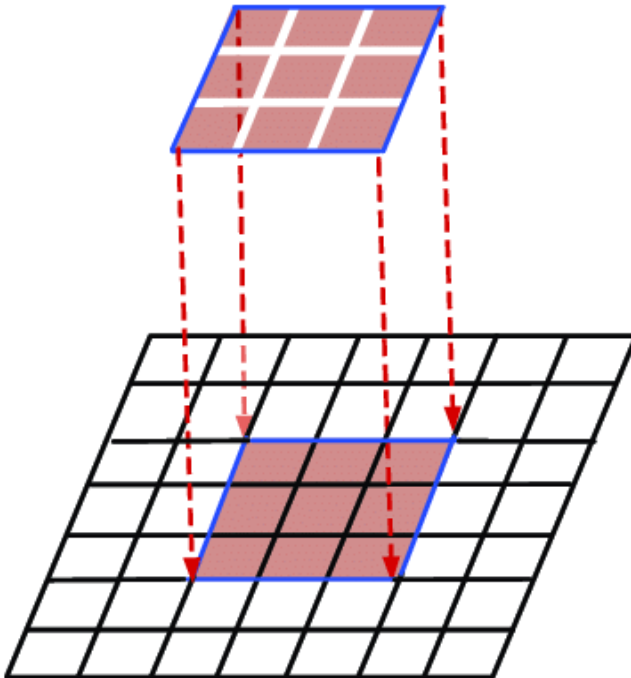
Zweite Schicht

2	1	1	0	1
2	0	2	1	1
0	1	0	2	0
0	2	2	0	2
2	1	2	0	2

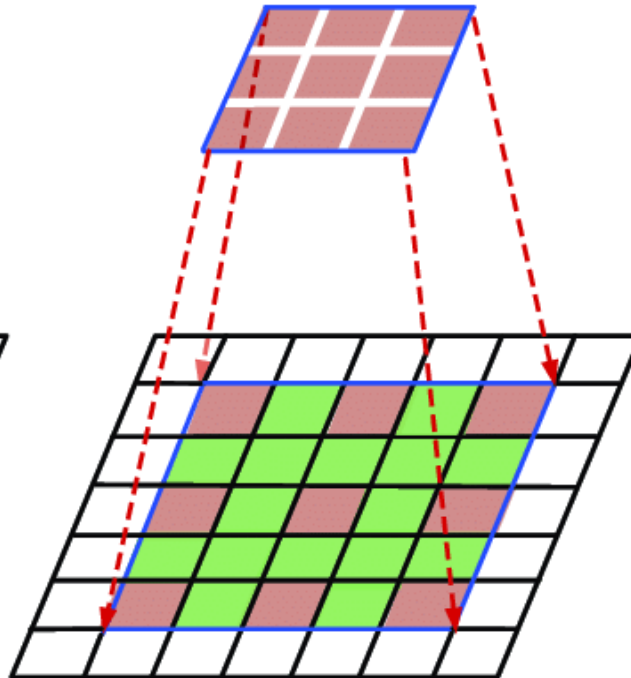
- Sichtweite ist der Bereich, in dem Informationen eines Elements die Berechnung eines anderen Elements beeinflussen können
- Höhere Sichtweite bedeutet, größere und komplexere strukturelle Zusammenhänge werden erfasst
- Sichtweite eines Elements erhöht sich durch:
 - Größe des Kernels
 - Mehrere convolutions. "3x3" -> "5x5"
 - Dilation

Dilation

Normal Convolution
 3×3

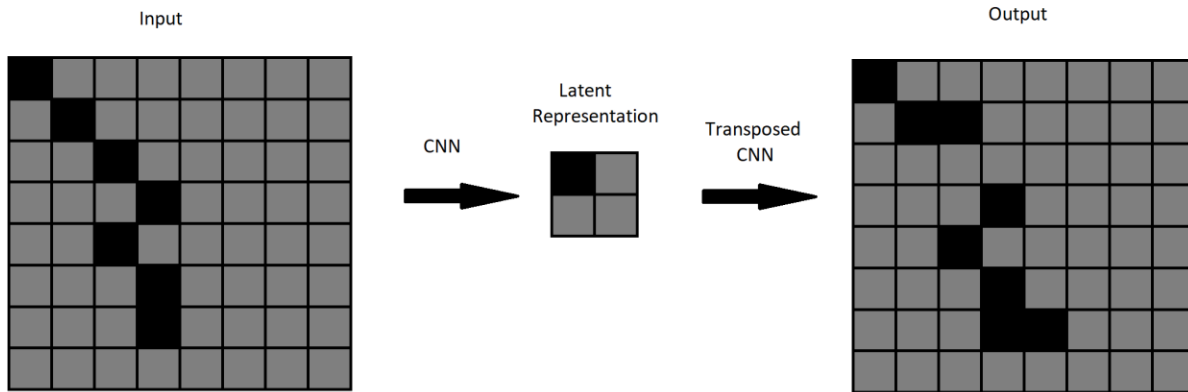


Dilated Convolution
 $3 \times 3, d=2$

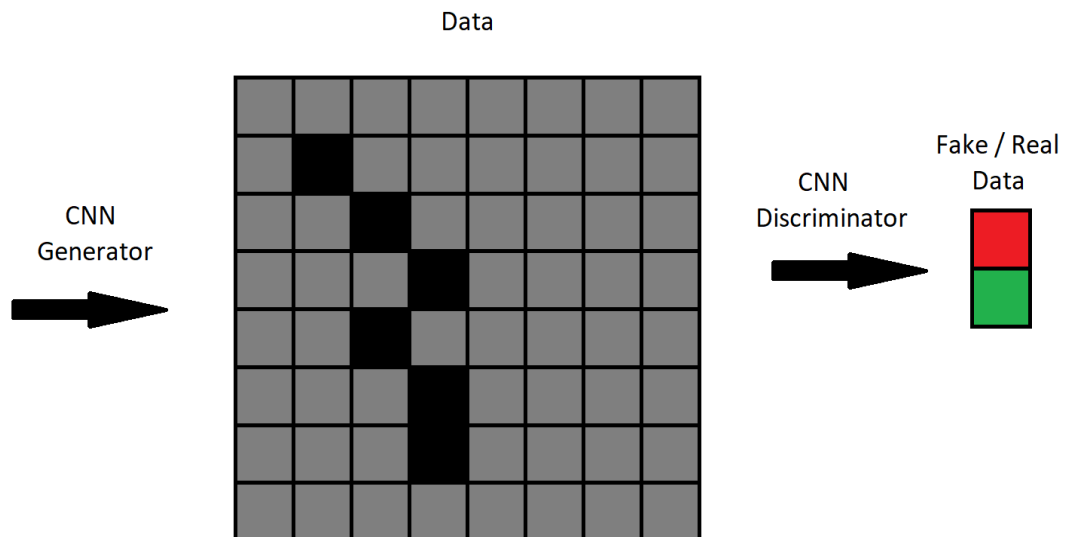


- Dilation erweitert Kernel durch Lücken
- Gleiche Parameteranzahl
- Zu viel Dilation kann zu schwächerer lokaler Erkennung führen

Generative CNN Models



CNNs erstellen latenten Vektor und rekonstruieren daraus Daten



CNNs erstellen Daten und entscheiden über ihre Echtheit

Autoregressive Image Generation

$$P(x_1, x_2, \dots, x_{n^2}) = \prod_{i=1}^{n^2} P(x_i \mid x_1, x_2, \dots, x_{i-1})$$

Wahrscheinlichkeit eines Pixel
basiert auf seinen Vorgängern

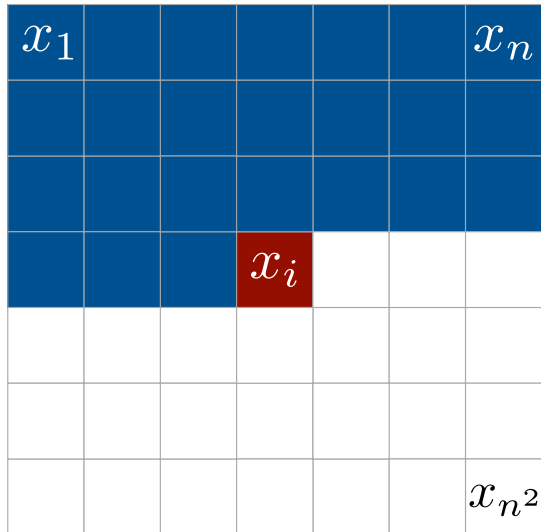


figure credit - [Aaron van den Oord et al.](#)

Maske für Kernel

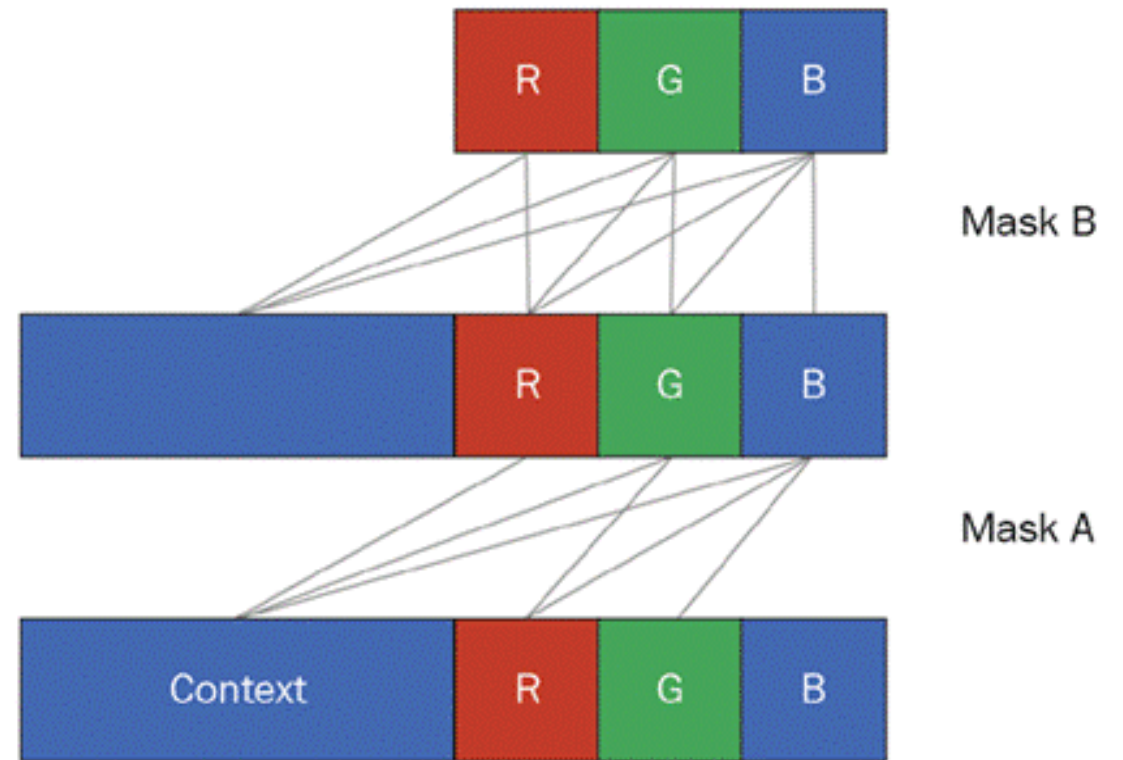
1	1	1	1	1
1	1	1	1	1
1	1	0	0	0
0	0	0	0	0
0	0	0	0	0

figure credit - [Aaron van den Oord et al.](#)

Channel Mask

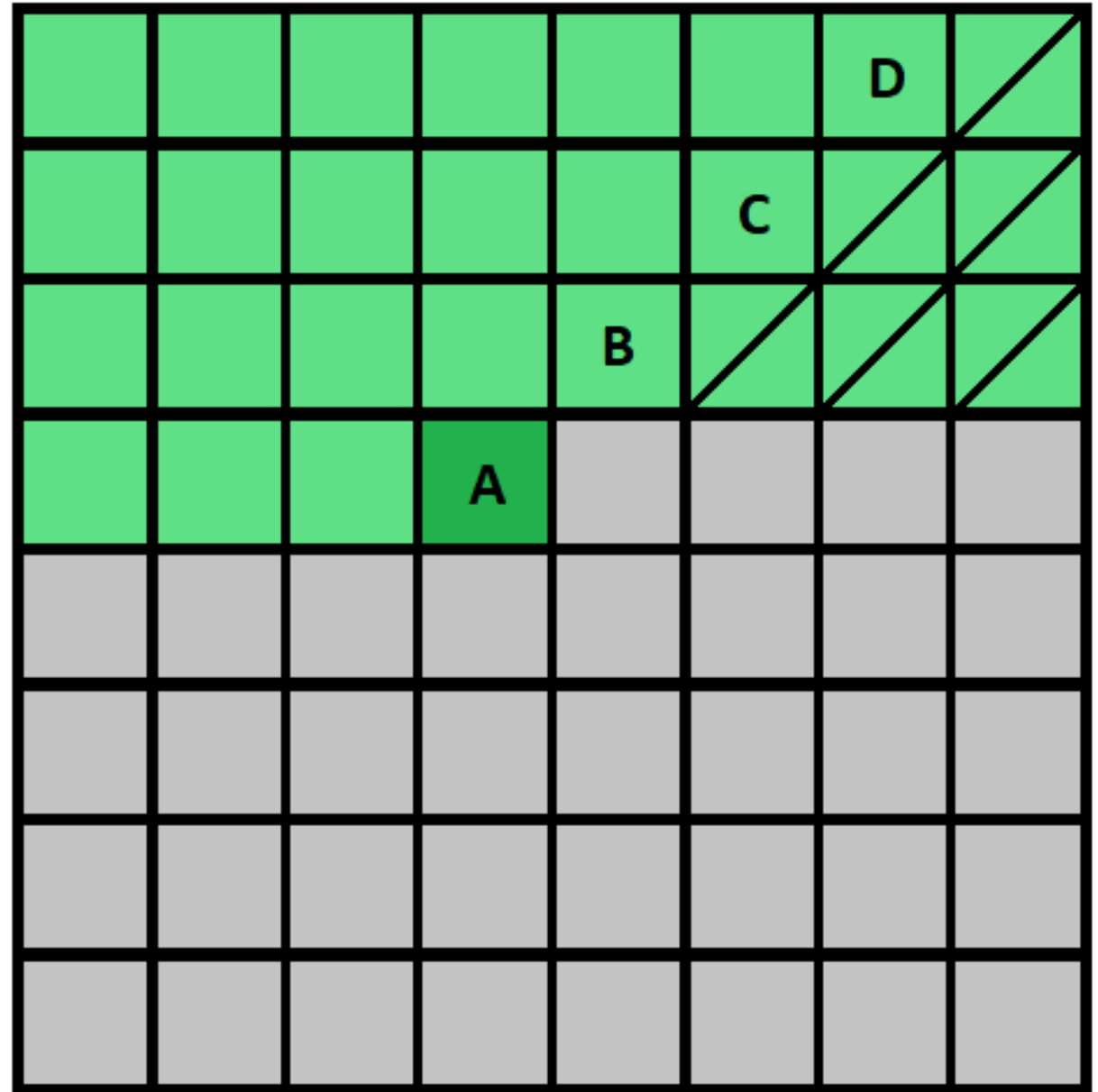
- Autoregressive Generation in Colour Channels

- (C) $\rightarrow$ R (C, R) $\rightarrow$ R
- (C, R) $\rightarrow$ G (C, R, G) $\rightarrow$ G
- (C, R, G) $\rightarrow$ B (C, R, G, B) $\rightarrow$ B

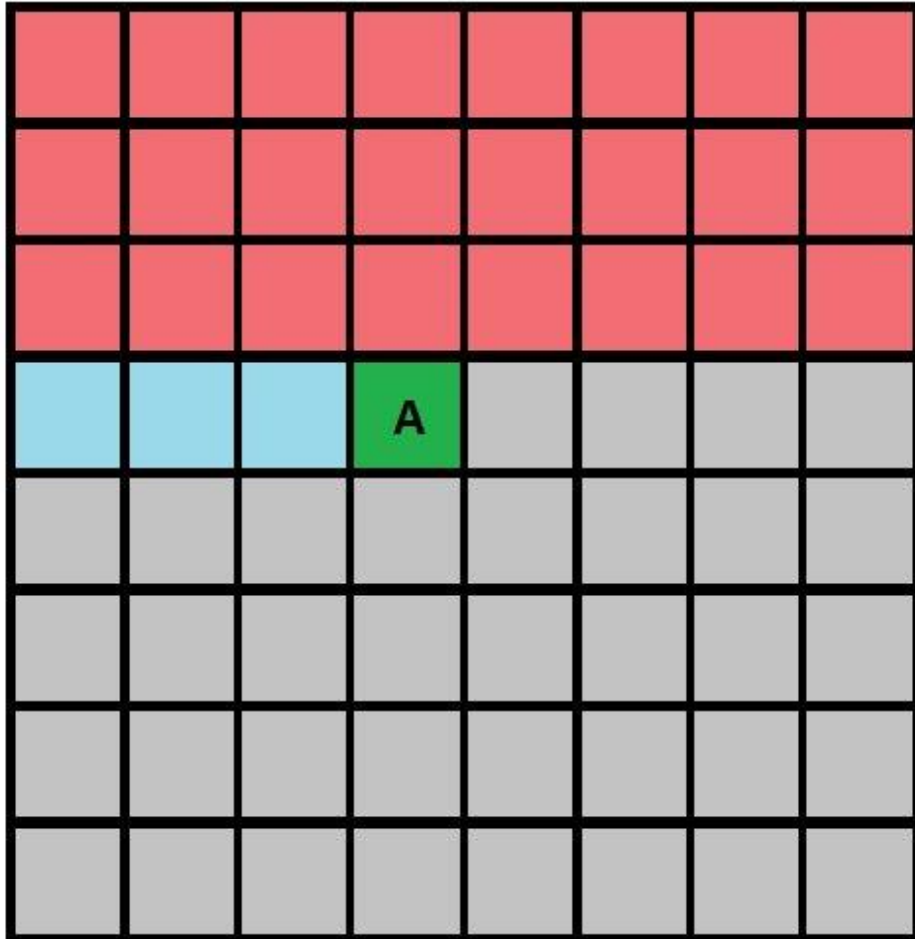


Blind Spot

- A erhält Vorgänger über B, C, D, ...
- B blendet rechte Pixel aus
- A kann nicht weiter rechts sehen, als B

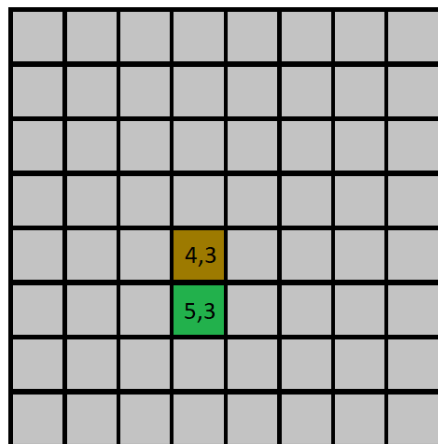


Blind Spot

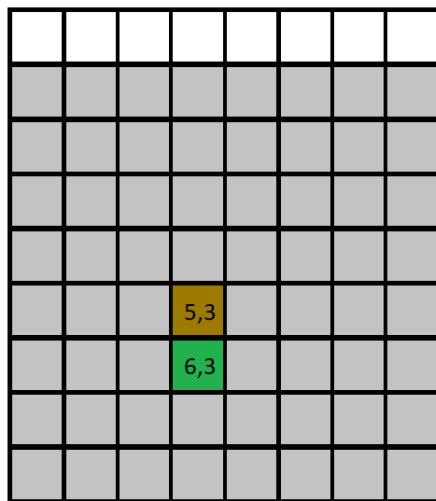


- Zwei getrennte Convolution Stacks
- Rot: Vertical Stack
 - Schaut sich alle Elemente über A an
- Blau: Horizontal Stack
 - Schaut sich alle Elemente in gleicher Reihe rechts an
- Informationen des Vertical Stack werden Horizontal Stack zugeführt

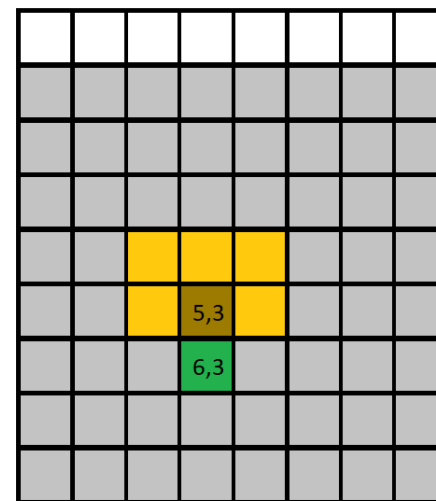
Vertical Stack



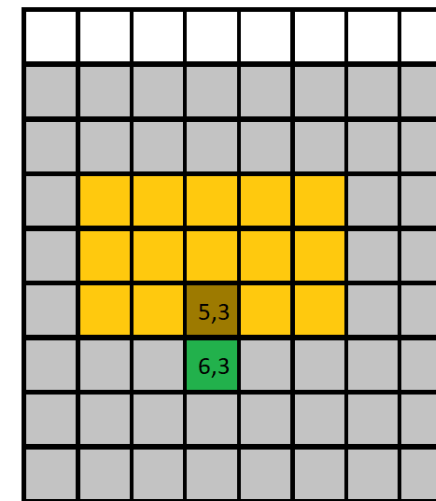
Padding



vertical masked 3x3 Conv
1. Layer



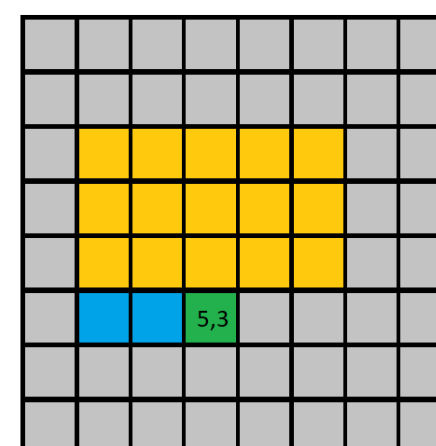
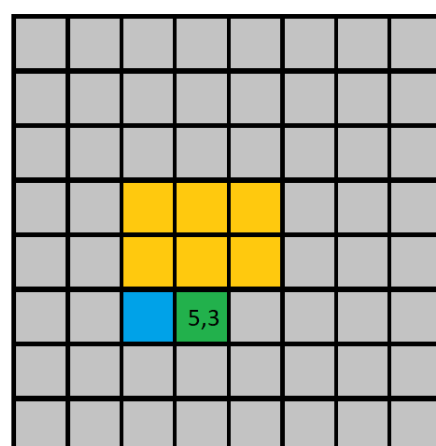
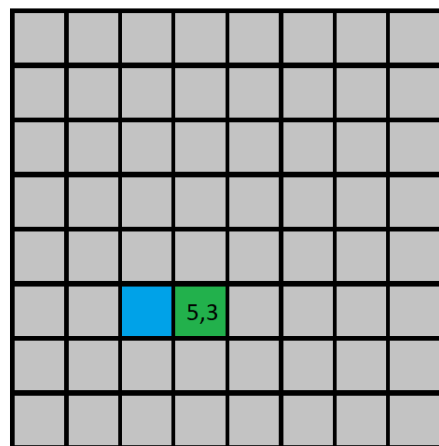
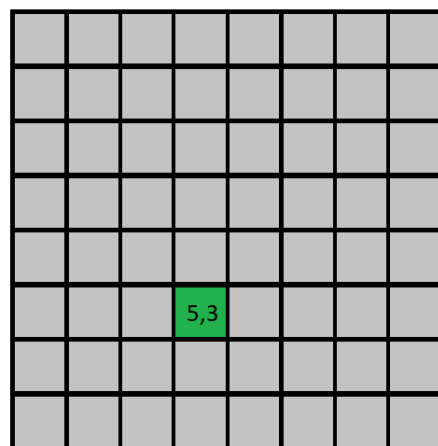
vertical masked 3x3 Conv
2. Layer



1x1 Convolutuon
Addition von Vertical 5,3 and Horizontal 5,3



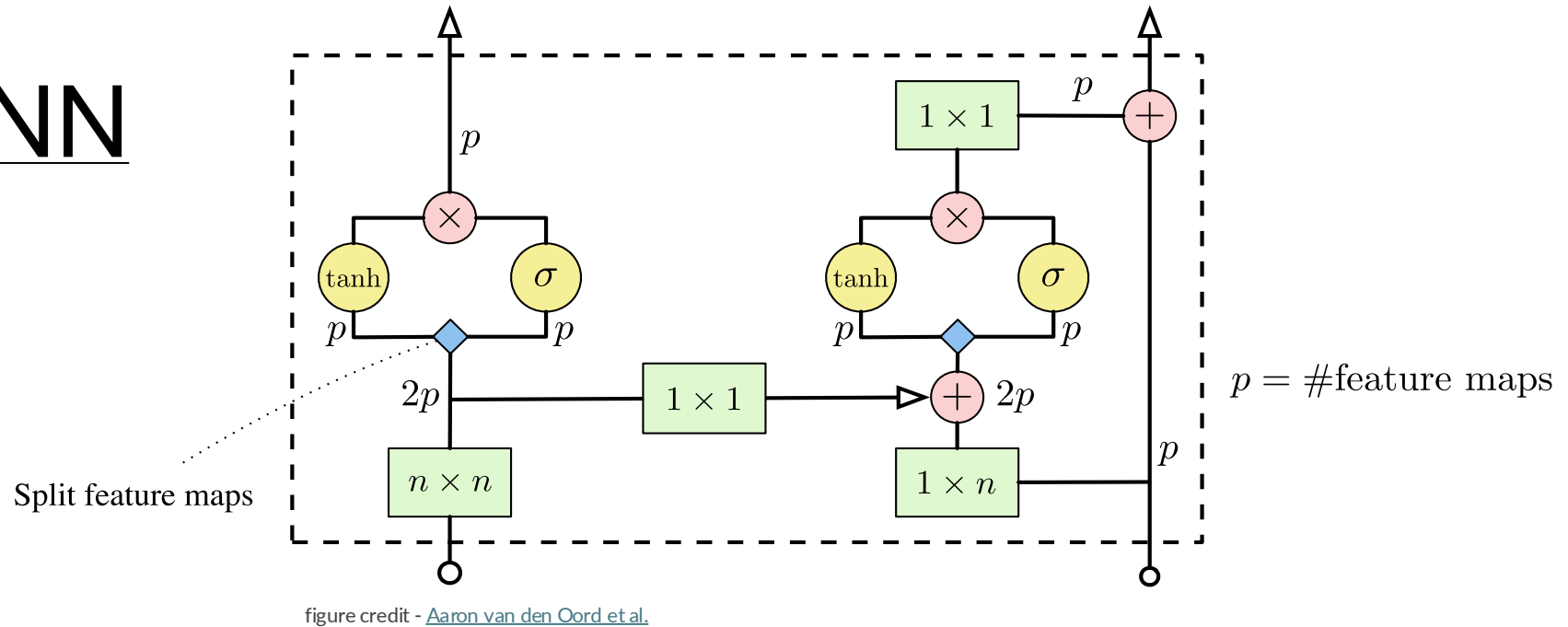
Horizontal Stack



horizontal masked 3x3 Conv
1. Layer

horizontal masked 3x3 Conv
2. Layer

Gated PixelCNN

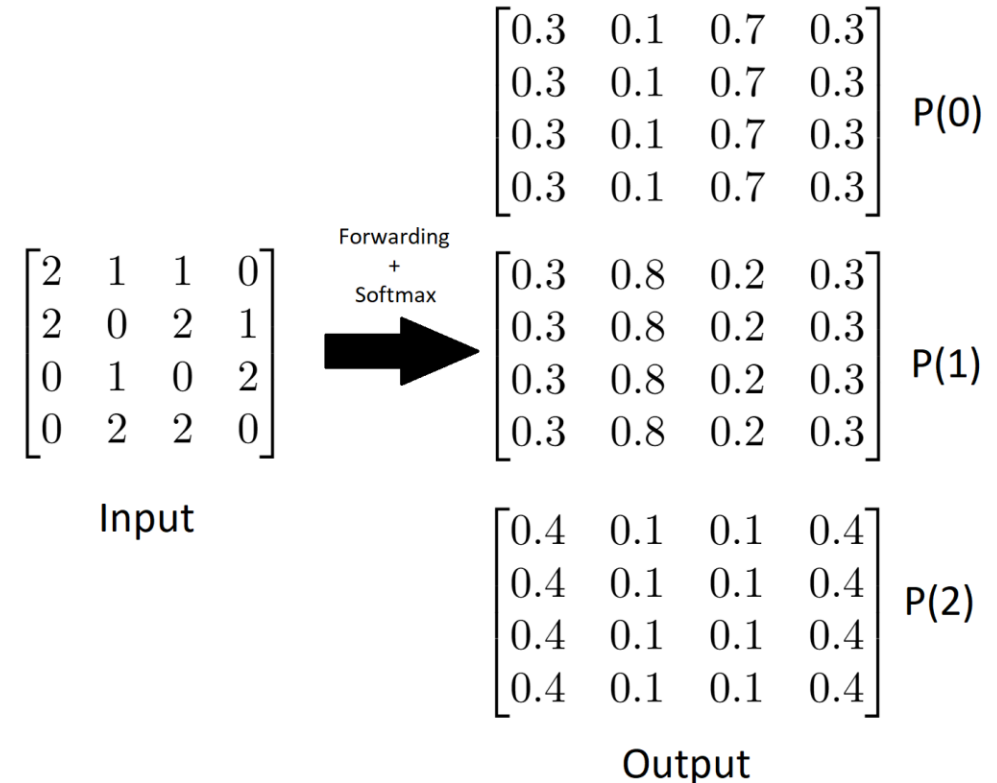


- Tanh gibt Signal aus $(-1, 1)$
- Sigmoid gibt Gating Wert aus $(0, 1)$
- Ist LSTM Gating-Struktur nachempfunden
- Keine Speicherzelle, weniger Kontrolle
- Residual Connection im Horizontal Stack
- Ist Hyper-Parameter

$$Y = \tanh(W * x) \odot \text{sigmoid}(W * x)$$

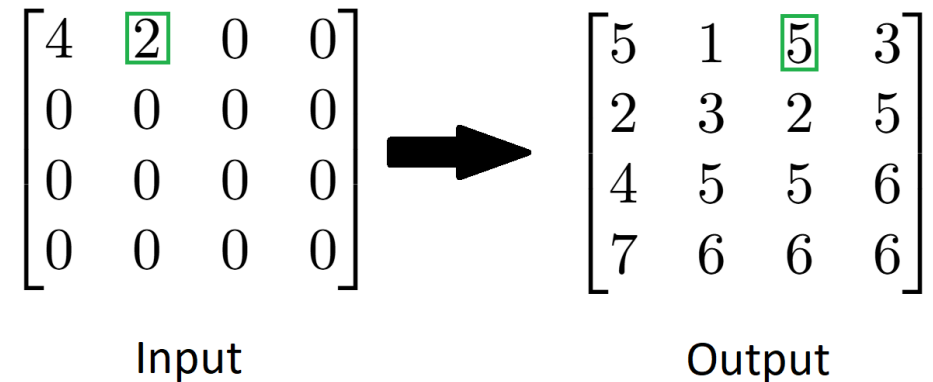
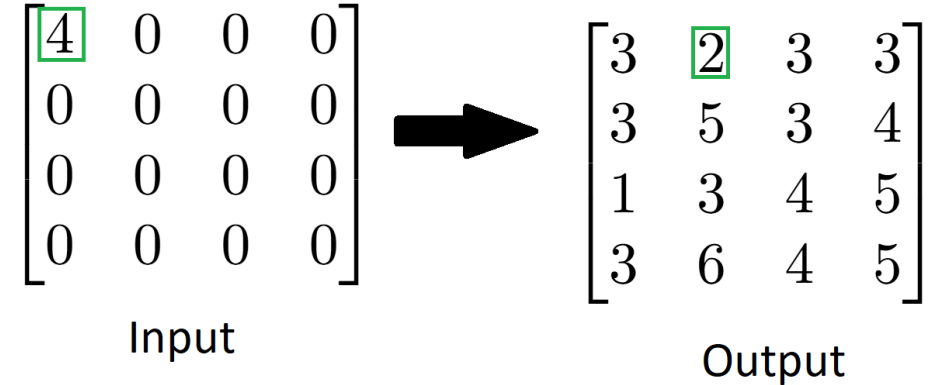
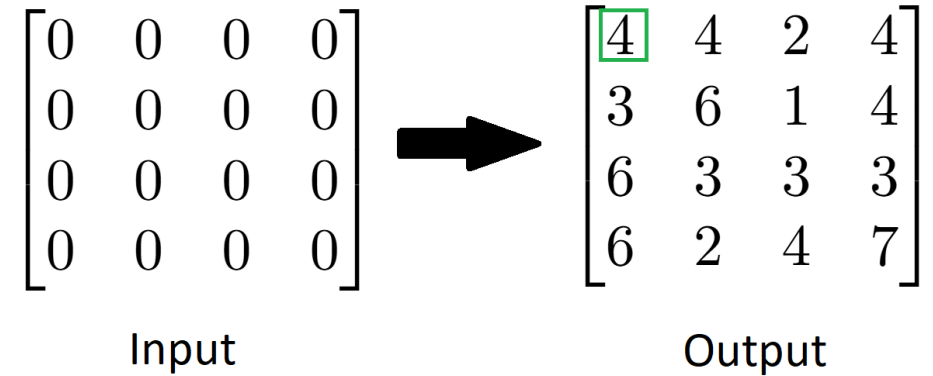
Training

- Input ist Bild aus Trainingsdaten
- Alle Vorgänger aller Pixel vorhanden
-> Paralleles Training
- Output ist diskrete Wahrscheinlichkeitsverteilung aller möglichen Farbwerte jedes Pixels mittels Softmax
[Classes, Channel, Height, Width]
- Loss wird berechnet als Negative Log-Likelihood von input und output



Sampling

- Ein Durchlauf pro Pixel * Channel
- Ein 64 * 64 RGB Bild braucht 12.288 Durchläufe
- Input wird für Effizienz abgeschnitten



Conditional PixelCNN

Latent Vector $\mathbf{h}$

$$p(\mathbf{x}|\mathbf{h}) = \prod_{i=1}^{n^2} p(x_i|x_1, \dots, x_{i-1}, \mathbf{h})$$

- Vector $\mathbf{h}$ wird als Bias auf den Input addiert
- $\mathbf{V}$ ist Term, der auf $\mathbf{h}$ basiert
- Nur Information über Was generiert werden soll
nicht Wo

$$Y = \tanh(W * x + \mathbf{V} * \mathbf{h}) \odot \text{sigmoid}(W * x + \mathbf{V} * \mathbf{h})$$

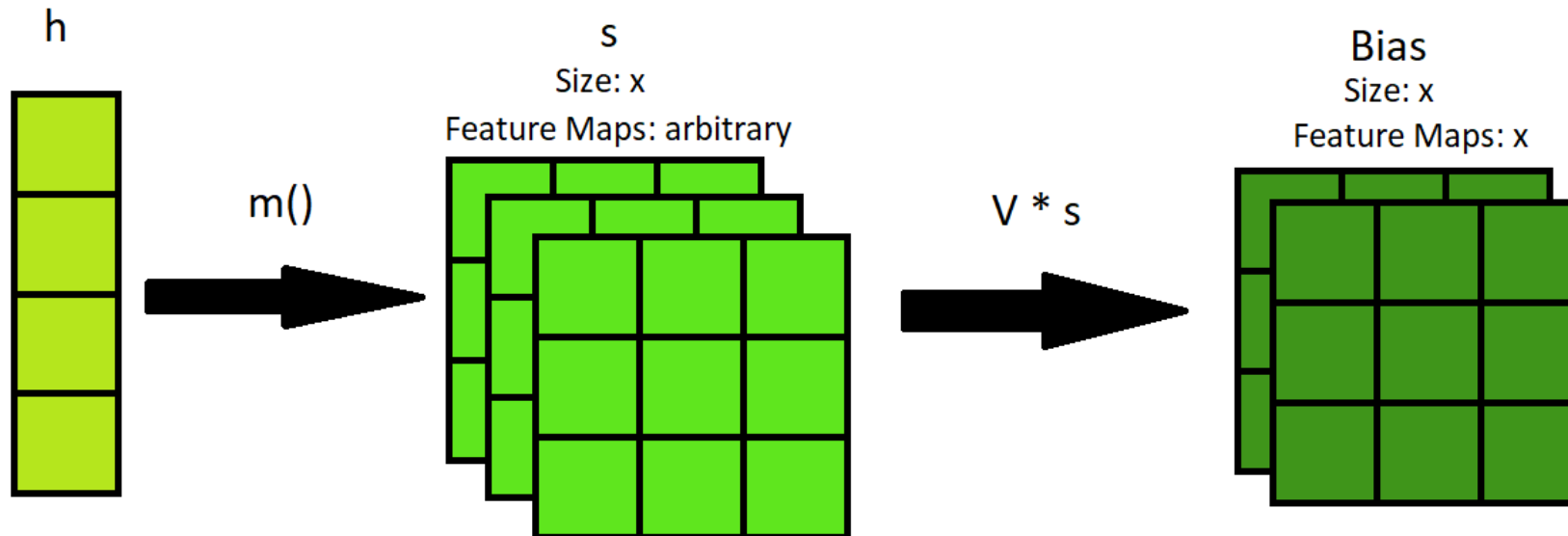
Gated Block ohne Condition

$$Y = \tanh(W * x) \odot \text{sigmoid}(W * x)$$

Conditional PixelCNN

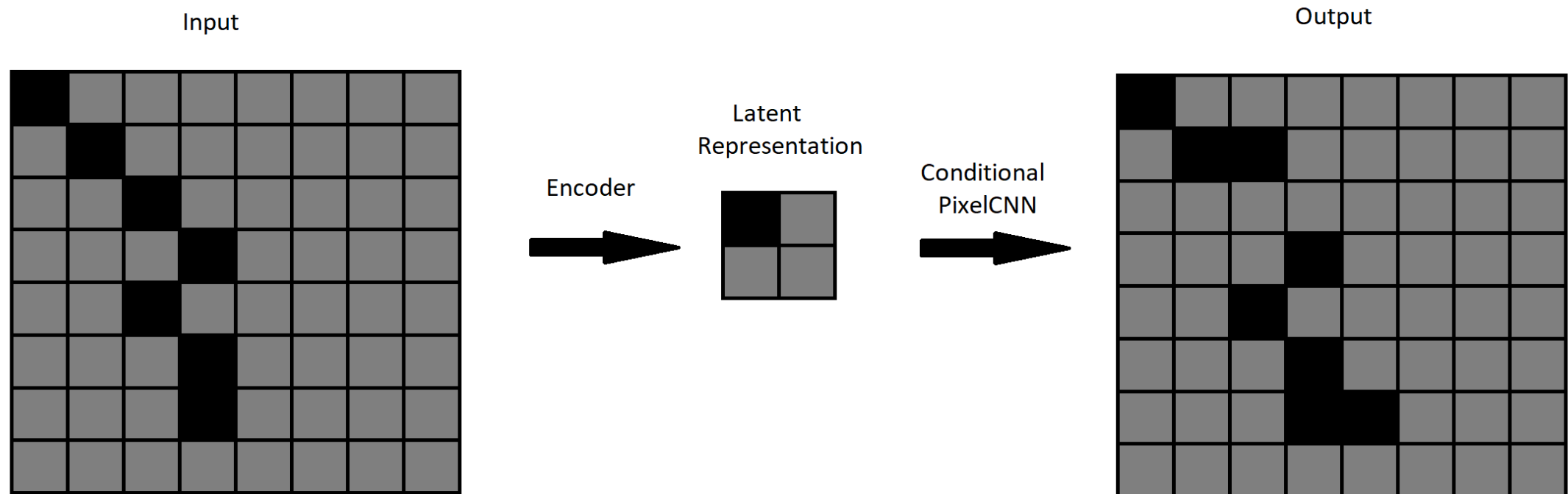
$$Y = \tanh(W * x + \mathbf{V} * \mathbf{s}) \odot \text{sigmoid}(W * x + \mathbf{V} * \mathbf{s})$$

- h enthält räumliche Informationen
- $s = m(h)$
- $m()$: Deconvolutional Neural Network (transposed Convolutions)
- $\mathbf{V} * s$: unmasked 1 x 1 Convolution



Conditional PixelCNN in Auto-Encoder

- PixelCNN wird als Decoder benutzt
- Latent Vector wird als Condition benutzt
- Encoder erarbeitet Globale Struktur, PixelCNN lokale Struktur



Tests: PixelCNN

Paper: "Conditional Image Generation with PixelCNN Decoders"

Model	NLL Test (Train)	
Uniform Distribution: [30]	8.00	
Multivariate Gaussian: [30]	4.70	
NICE: [4]	4.48	
Deep Diffusion: [24]	4.20	■ CIFAR-10
DRAW: [9]	4.13	■ Bits/Dim
Deep GMMs: [31, 29]	4.00	■ 2016
Conv DRAW: [8]	3.58 (3.57)	
RIDE: [26, 30]	3.47	
PixelCNN: [30]	3.14 (3.08)	
PixelRNN: [30]	3.00 (2.93)	
Gated PixelCNN:	3.03 (2.90)	

Tests: PixelVAE

- Paper: "PixelVAE: A Latent Variable Model for Natural Images"
- MNIST
- 2016

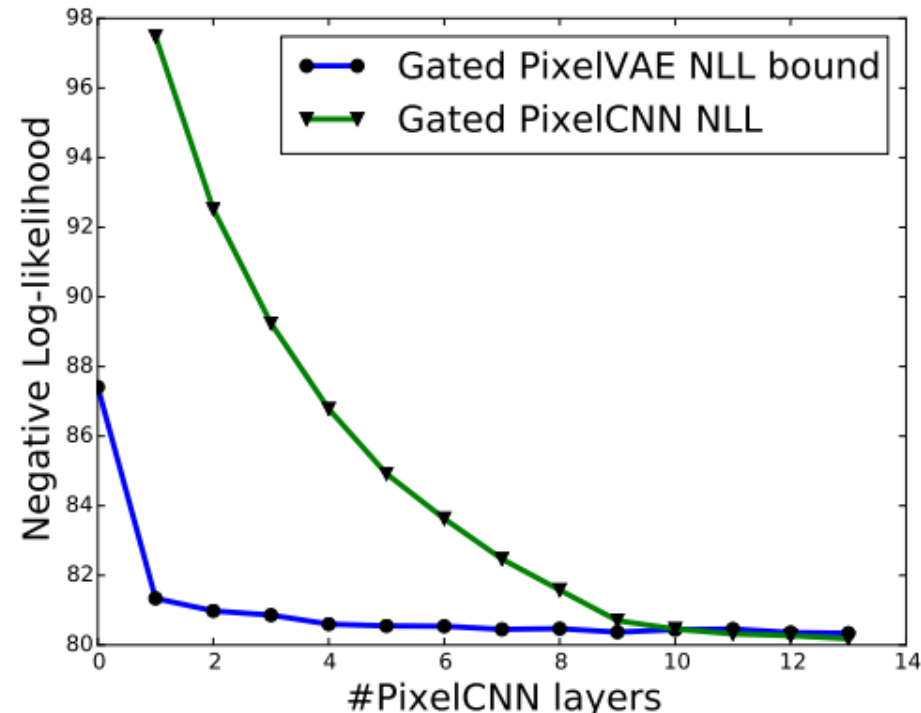


figure credit - [Gulrajani et al.](#)

Model	NLL Test
DRAW (Gregor et al., 2016)	≤ 80.97
Discrete VAE (Rolfe, 2016)	$= 81.01$
IAF VAE (Kingma et al., 2016)	≈ 79.88
PixelCNN (van den Oord et al., 2016a)	$= 81.30$
PixelRNN (van den Oord et al., 2016a)	$= 79.20$
Convolutional VAE	≤ 87.41
PixelVAE	≤ 80.64
Gated PixelCNN (our implementation)	$= 80.10$
Gated PixelVAE	$\approx 79.48 (\leq 80.02)$
Gated PixelVAE without upsampling	$\approx \mathbf{79.02} (\leq 79.66)$

figure credit - [Gulrajani et al.](#)

PixelCNN++

$$X = \mu + s * \log\left(\frac{y}{1-y}\right)$$

$$p(y|w, \mu, s) = \sum_{i=1}^k w_i \left[\sigma\left(\frac{y + 0.5 - \mu_i}{s_i}\right) - \sigma\left(\frac{y - 0.5 - \mu_i}{s_i}\right) \right]$$

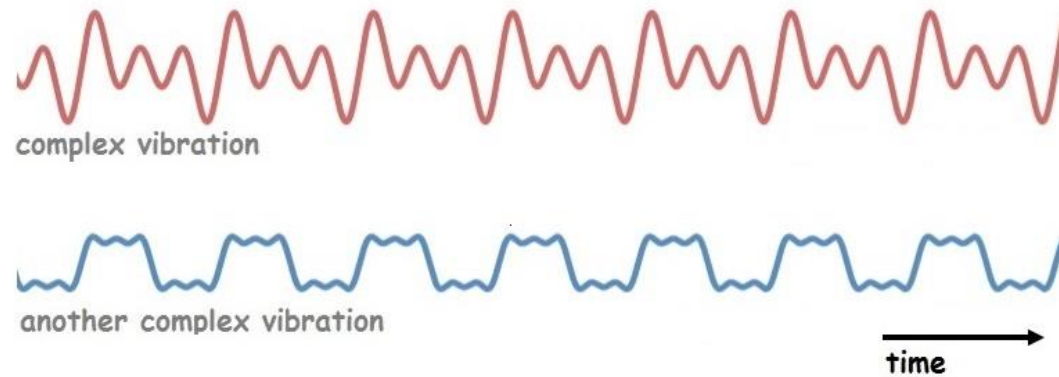
- Discretized Logistic Mixture Likelihood anstatt Softmax
 - Es gibt k Verteilungsfunktionen
 - Je Funktion Gewicht w, Streuung s und Mittel μ
 - Wahrscheinlichkeit von Wert y ist Bereich $[y-0.5, y+0.5]$
 - Erfasst Distanz von Werten (124 näher an 125 als 15)
- Weitere Änderungen

PixelCNN (van den Oord et al., 2016b)	3.14
VAE with IAF (Kingma et al., 2016)	3.11
Gated PixelCNN (van den Oord et al., 2016c)	3.03
PixelRNN (van den Oord et al., 2016b)	3.00
PixelCNN++	2.92

- Paper: "PixelCNN++"
- CIFAR-10
- 2017
- Bits/Dim

Sound Generation: WaveNet

1-Dimensionale Strukturen



<https://pressbooks.pub/sound/chapter/time-domain-graphs-rewrite/>

Autoregressive Generierung

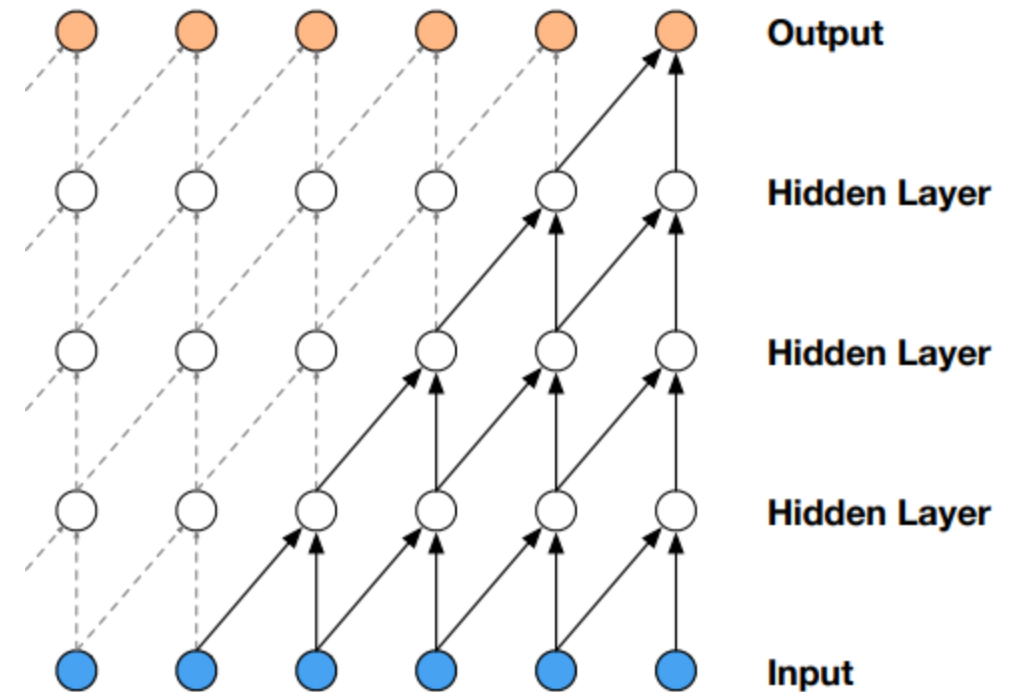
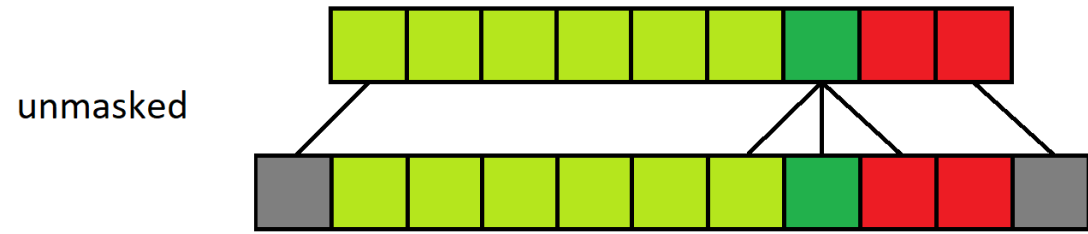


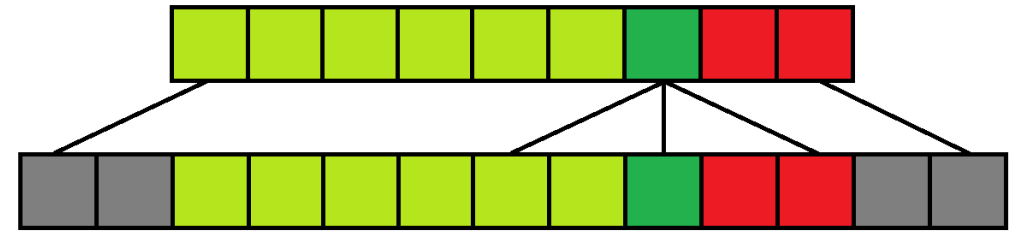
figure credit - [Aaron van den Oord et al.](#)

Shifted Causal Convolution

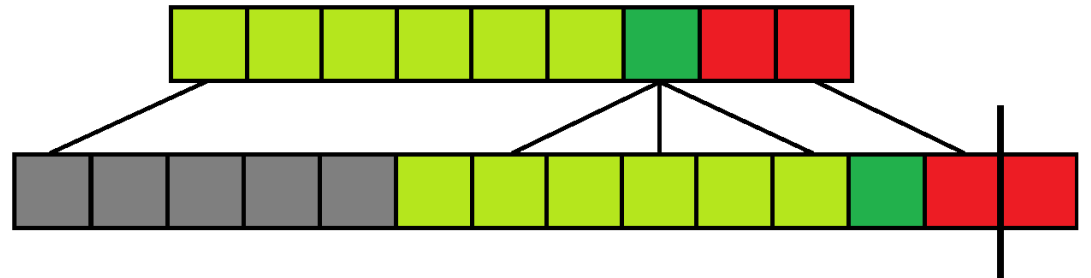
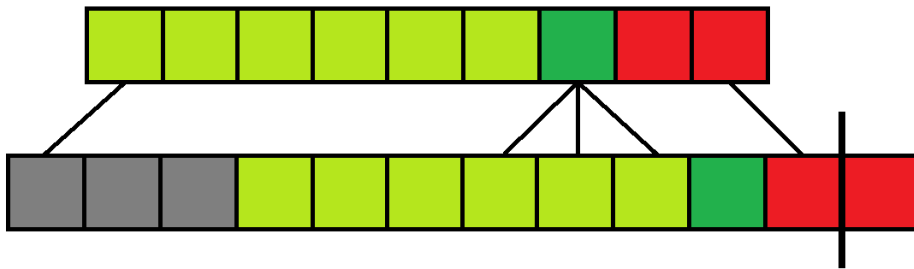
Dilation = 1



Dilation = 2



masked



Gated Block und Conditioning

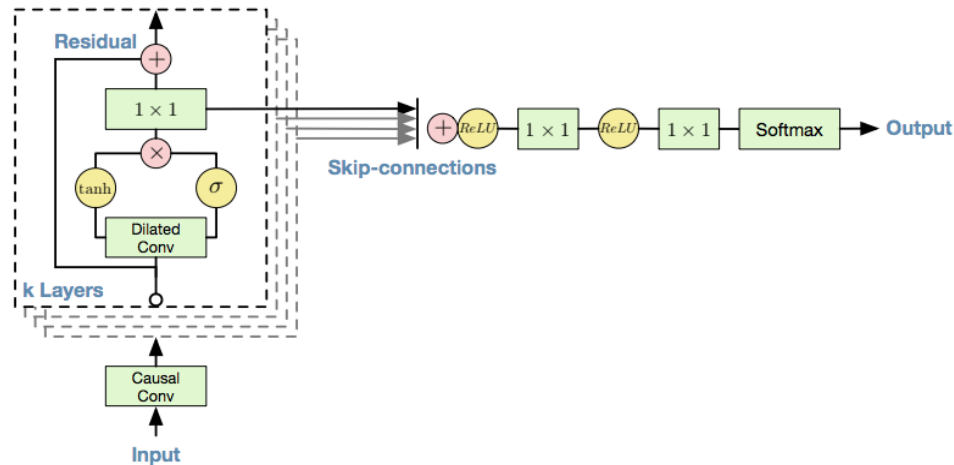


figure credit - [Aaron van den Oord et al.](#)

zeitunabhängig

$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x} + V_{f,k}^T \mathbf{h}) \odot \sigma(W_{g,k} * \mathbf{x} + V_{g,k}^T \mathbf{h})$$

zeitabhängig

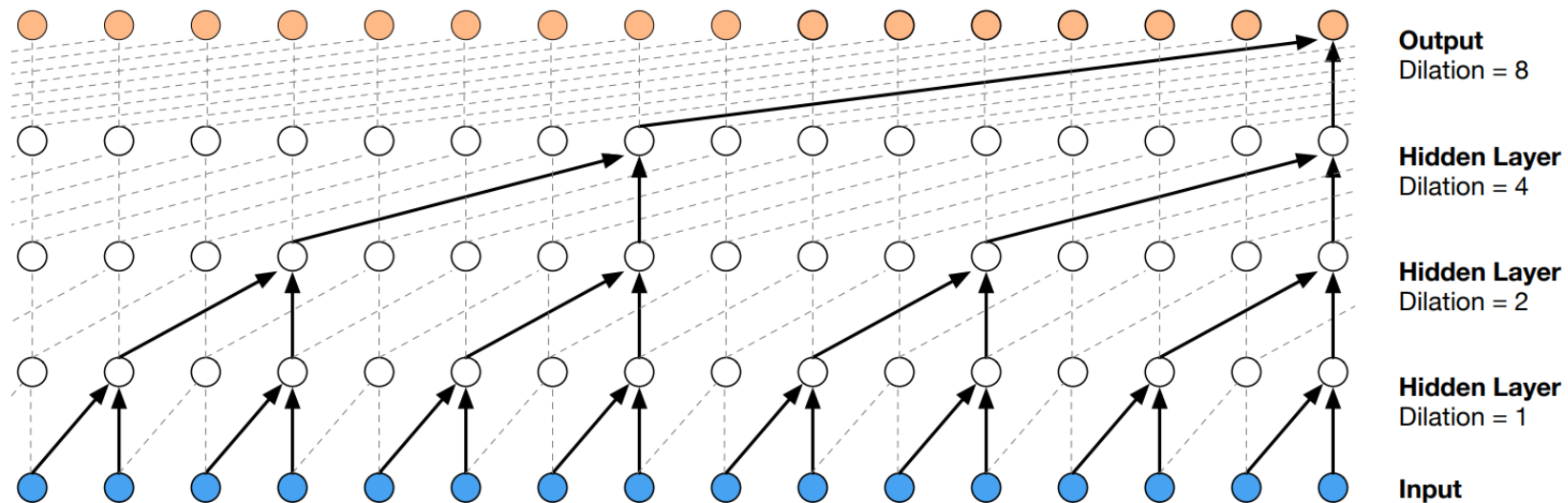
$$\mathbf{z} = \tanh(W_{f,k} * \mathbf{x} + V_{f,k} * \mathbf{y}) \odot \sigma(W_{g,k} * \mathbf{x} + V_{g,k} * \mathbf{y})$$

figure credit - [Aaron van den Oord et al.](#)

- Gleicher Gating Mechanismus und Residual Connection wie in Gated PixelCNN
- Skip Connection zwischen Gated Blocks
- Gleiche Conditionierung wie PixelCNN
 - $\mathbf{y}$: transposed CNN
 - V^T : lineare Projektion
 - V : 1x1 convolution

Input Length

- Input Daten sehr lang
- Eine Sekunde bei 16 kHz benötigt 16000 Elemente
- Dilation in Exponentiellen Blöcken (1, 2, 4, ..., 512, 1, 2, 4, ..., 512, 1, 2, 4, ..., 512)
 - Ein 512-Block hat gleiche Sichtweite wie eine 1*1024-Convolution

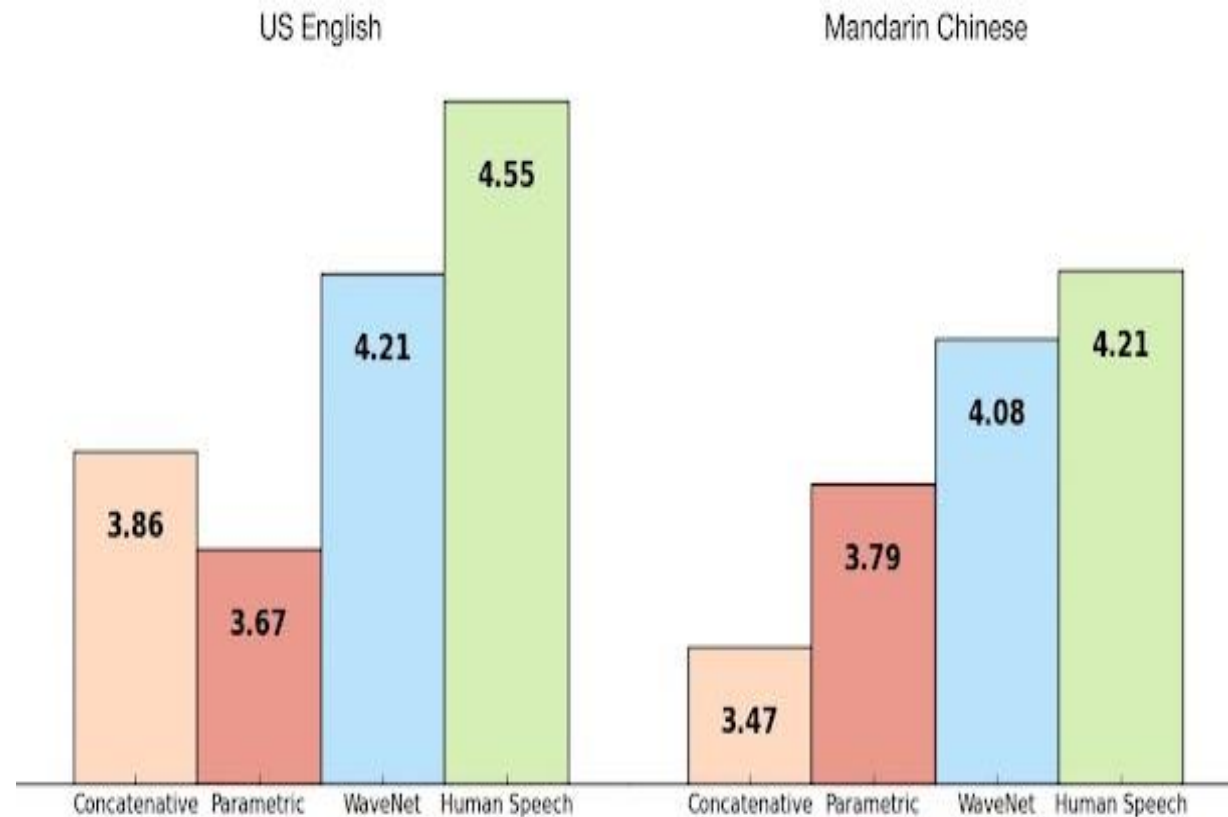


Context Stacks

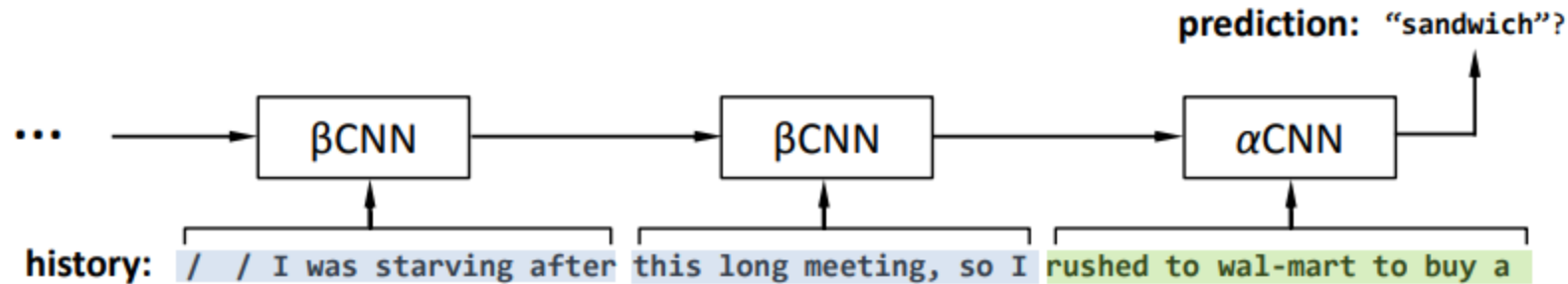
- Context Stacks bearbeiten große Teile des Inputs
- Conditionieren größeres WaveNet, das sich nur kleine Teile anschaut -> lokale Zusammenhänge
- Context Stacks haben große Sichtweite mit wenig Parametern
- Nutzen auch Pooling Layer
 - Pooling Layer extrahieren vereinzelte Elemente um Dimension des Input zu verkleinern
- Führt zu nachhaltig wachsender Komplexität

Tests: Text to Speech

- Vergleich von WaveNet, Concatenative (Segment stitching) und Parametric (LSTM-RNN)
- MOS (Mean Opinion Score): Beurteilung von Menschen
- 2016



Text Generation: genCNN



- "/" ist Padding
- Autoregressiv -> Causal, Masked Convolution
- β CNN: Short Term fokussiert
- α CNN: Long Term fokussiert, mehr Dilation, Layers und größere Kernel

Papers

- [PIXELVAE: A LATENT VARIABLE MODEL FOR NATURAL IMAGES](#)
- [WAVENET: A GENERATIVE MODEL FOR RAW AUDIO](#)
- [A Convolutional Architecture for Word Sequence Prediction](#)
- [Conditional Image Generation with PixelCNN Decoders](#)
- [PIXELCNN++: IMPROVING THE PIXELCNN WITH DISCRETIZED LOGISTIC MIXTURE LIKELIHOOD AND OTHER MODIFICATIONS](#)