

Convolutional Neural Networks



Gliederung

- Motivation
- Convolutional Layer
- Pooling Layer
- Aktivierungsfunktion
- Genereller Aufbau
- Anwendungsbeispiele

Motivation (I): Machine Learning




42 4d e6 4a 05 00 00 00 00 00 36 00 00 00 28 00
00 00 54 01 00 00 54 01 00 00 01 00 18 00 00 00
00 00 00 00 00 00 23 2e 00 00 23 2e 00 00 00 00
00 00 00 00 00 00 5d 9d af 4d 8b 9d 6a a5 b5 54
8d 9c 6c a3 b2 47 7f 90 59 94 a4 63 9e b2 5d 9c
b0 63 a2 b6 8b cd df 5f a0 ae 51 94 9d 53 95 9a
60 a4 a9 54 96 a1 72 b7 c0 5c a2 a2 67 ad ad 5b
99 a3 7a b8 c2 42 7f 81 32 71 6f 45 84 82 5e 9e



Motivation (I): Machine Learning

Problem: Semantik des Bildes nicht maschinenverarbeitbar

- ❖ Machbar: Alles was vom Inhalt abstrahiert
 - Kontrast/Helligkeit (global) verändern
 - Alle Pixel einer bestimmten Farbe verändern (z.B. transparent machen)
 - ...
- ❖ Nicht (“einfach”) machbar: Inhaltsabhängige Operationen
 - Helikopter isolieren
 - Wolken entfernen
 - Gras grüner machen
 - ...



Motivation (II): Convolution

“Standardlösung” für maschinelles Lernen: Multilayer perceptron (MLP) - “fully-connected”

- Funktioniert auch für andere, ähnliche Probleme zuverlässig gut
 - Mittels Backpropagation auf beliebige Anwendungen trainierbar
- Warum nicht einfach MLP verwenden?

Motivation (II): Convolution



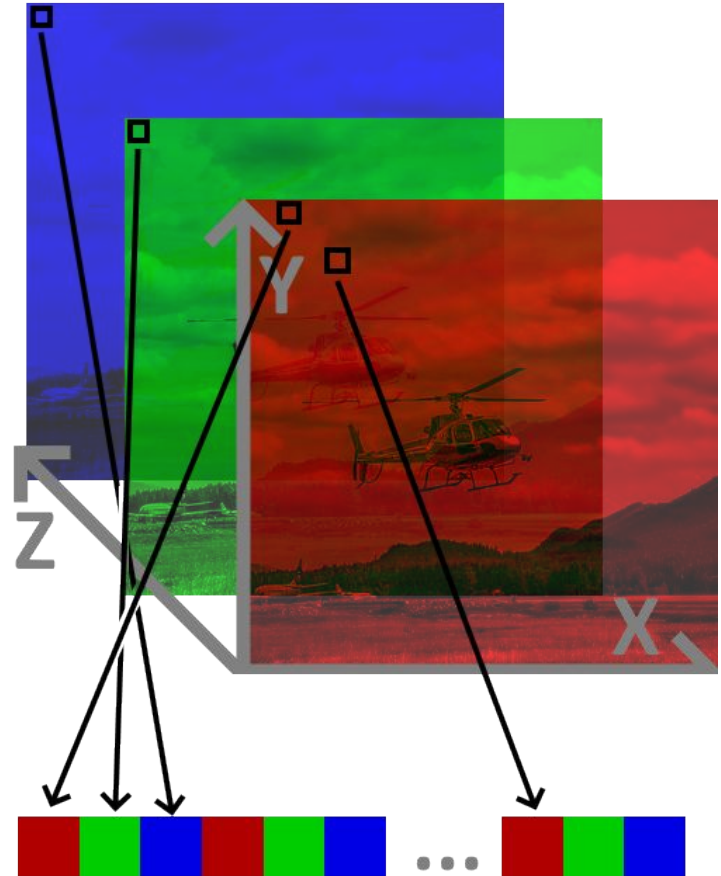
Motivation (II): Convolution


Problem 0: Dimensionalität

- (Farb-)Bild: 3 Dimensionen
- MLP kann nur auf eindimensionalen Daten arbeiten

Einfach lösbar: Daten auf eine Dimension projizieren (wie bei Speicher auf Festplatte)

→ kein wirkliches Problem





Motivation (II): Convolution

Problem 1: Positionsunabhängigkeit

$$(0 \ 42 \ \dots \ 255) * \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 & 9 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 1 & 2 & 3 & 4 \end{pmatrix} = (a \ b \ c \ d \ e)$$

Motivation (II): Convolution

Problem 1: Positionsunabhängigkeit

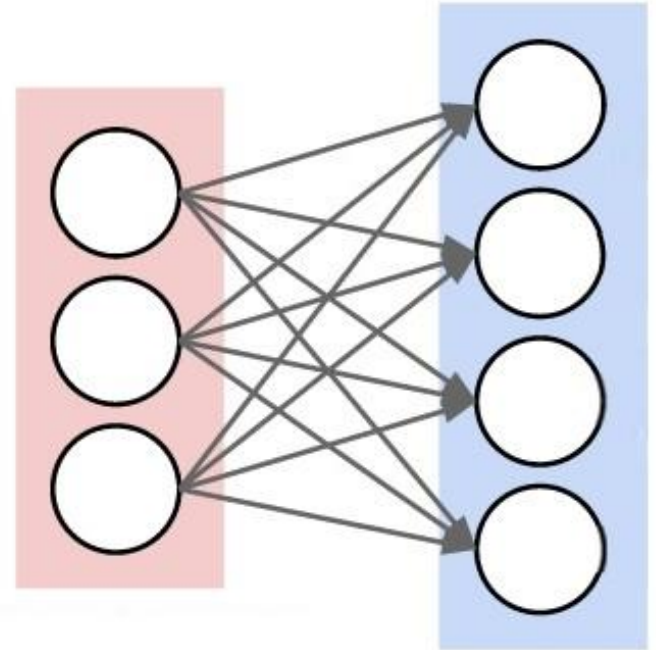
- Formel: Alle Daten/Pixel haben gleichwertige Korrelation zueinander
- Wahrheit: Dicht beieinander liegende Pixel haben vermutlich mehr miteinander zu tun als weit entfernte
- MLP müsste diese “Erkenntnis” erst gewinnen



Motivation (II): Convolution

Problem 2: Anzahl der Parameter

- Fully-Connected: Alle Neuronen übereinanderliegender Schichten miteinander verknüpft
- Pro Verknüpfung ein Parameter
- Eingabedaten: schon kleines Bild viele Einzeldaten (32x32x3 (RGB) -> 3072)
- Verkleinerung = Qualitätsverlust





Convolutional Layer

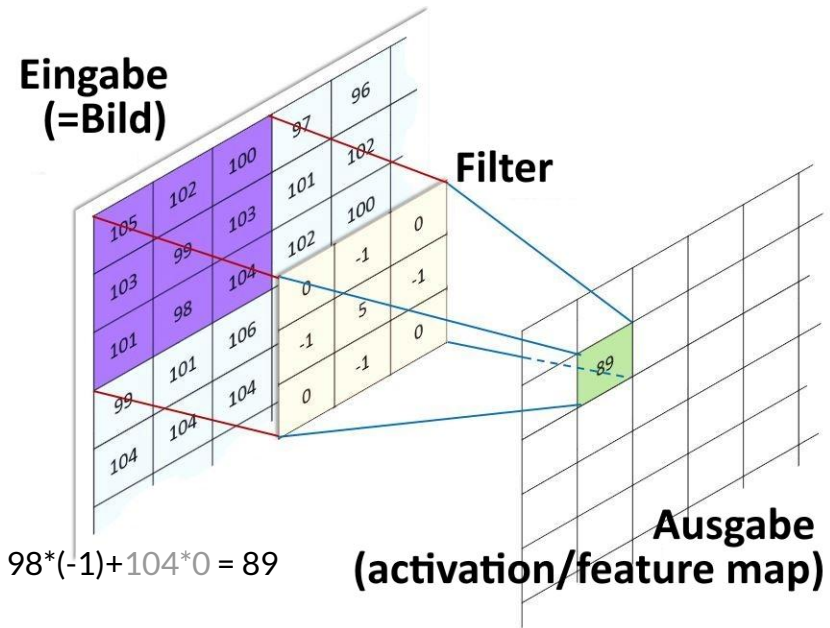
- Namensgebend für CNN -> zentrale Komponente
- “Convolution” - engl. für Faltung
- Eingabe: 3-dimensionale Daten, z.B. Farbbild
- Berechnung: Faltung mit mehreren, kleinen Filtern
- Ausgabe: 3-dimensionale Daten, Breite(Eingabe) x Höhe(Eingabe) x Filteranzahl
- Jeder Filter erkennt andere Eigenschaft (Kanten, Flächen bestimmter Farbe, ...)
- Faltung wird in der Bildverarbeitung häufig angewandt, um Bilder für eine Aufgabe anzupassen
 - Tiefpassfilter für Rauschreduzierung
 - Hochpassfilter / “Mexican Hat” für Kantenschärfung/-detektion
 - u.v.m. - siehe auch Themen aus Vorlesung

Faltung mit individuellem Filter

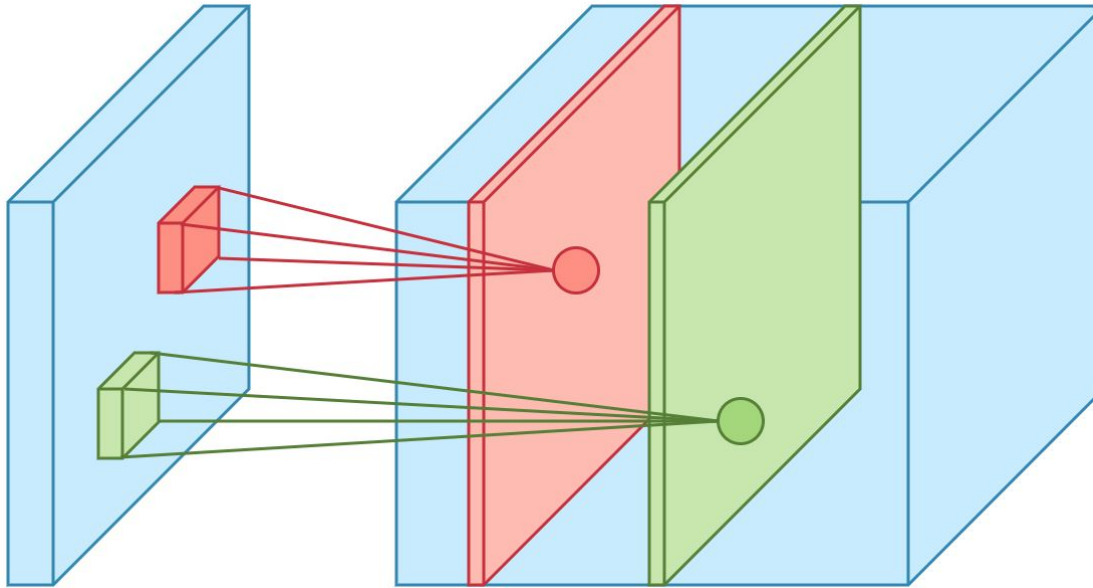
- Filter wird über Eingabe gelegt
- Übereinanderliegende Zellen multiplizieren
- Ergebnisse addieren -> Ausgabe: *Ein* Wert
- Diesen in Ausgabe schreiben
- Nächste Spalte ... Zeile: gleiches Vorgehen

Beispielrechnung (aus Bild):

$$105 \cdot 0 + 102 \cdot (-1) + 100 \cdot 0 + 103 \cdot (-1) + 99 \cdot 5 + 103 \cdot (-1) + 101 \cdot 0 + 98 \cdot (-1) + 104 \cdot 0 = 89$$



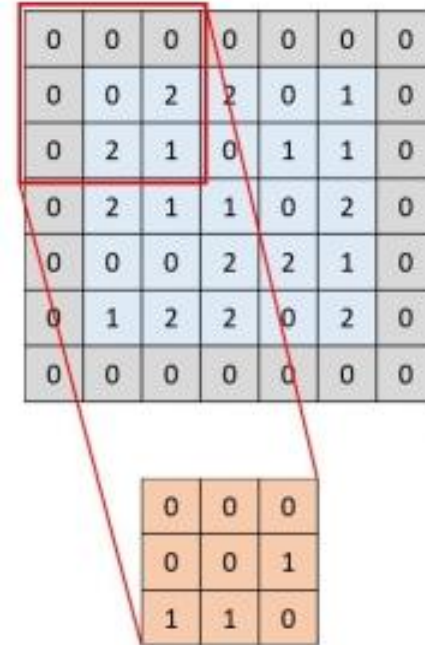
Convolutional Layer



All Feature Maps

Convolutional Layer

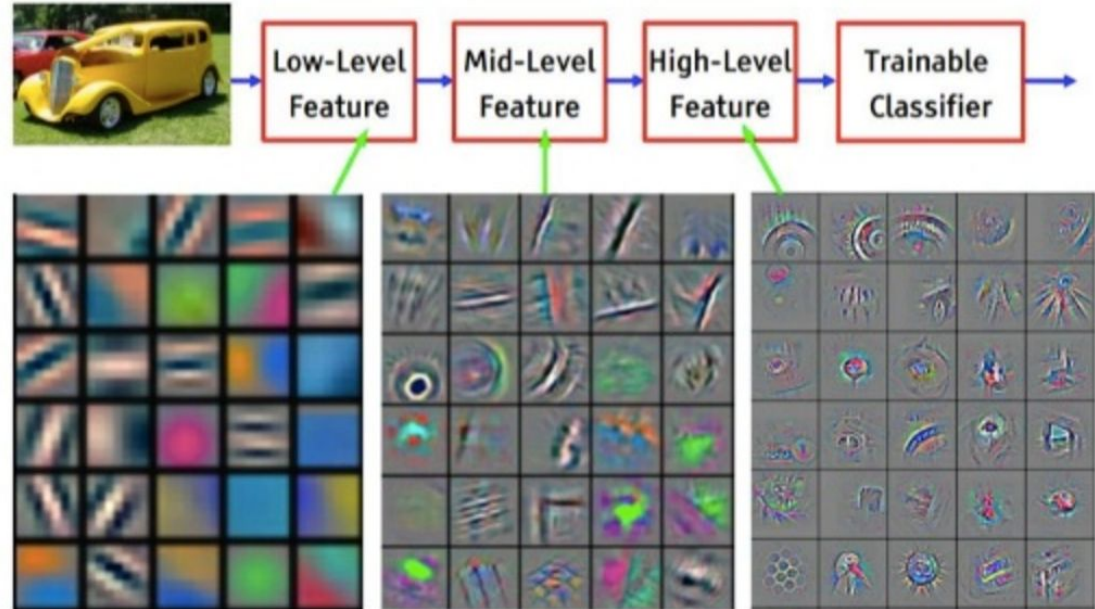
- Problem: Filter nicht auf Rand-Pixel zentrierbar
 - (falls Filter größer 1x1)
- Ausgabe kleiner als Eingabe
 - Probleme beim Konkatenerieren von Schichten
- Informationsverlust
- Lösung: Padding
 - Bild wird am Rand erweitert, sodass Filter passt
- Üblich: Padding mit Null



Convolutional Layer

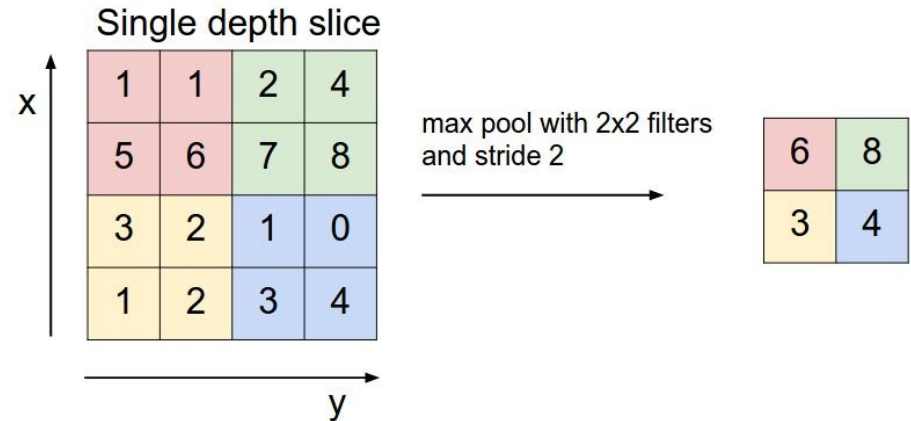
Bildquelle: <https://towardsdatascience.com/identifying-traffic-signs-with-deep-learning-5151eece09cb>

Beispiele für Filter



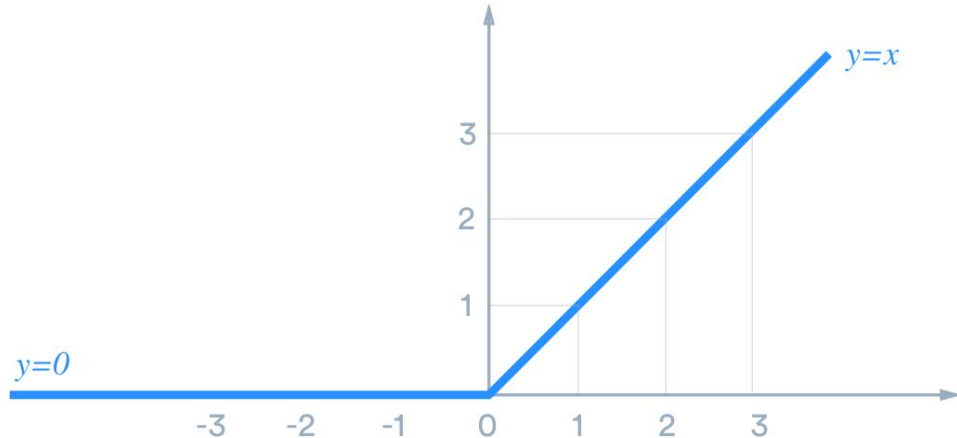
Pooling Layer

- Reduziert Datenmenge
 - Kann periodisch eingesetzt werden
 - Verschiedene Pooling-Operationen möglich
 - Max-Pooling
 - Average-Pooling
 - Median-Pooling
-
- Datenreduktion durch erhöhten **stride**
 - Nicht zwingend notwendig
 - **stride** kann auch direkt an CONV-Layer erhöht werden

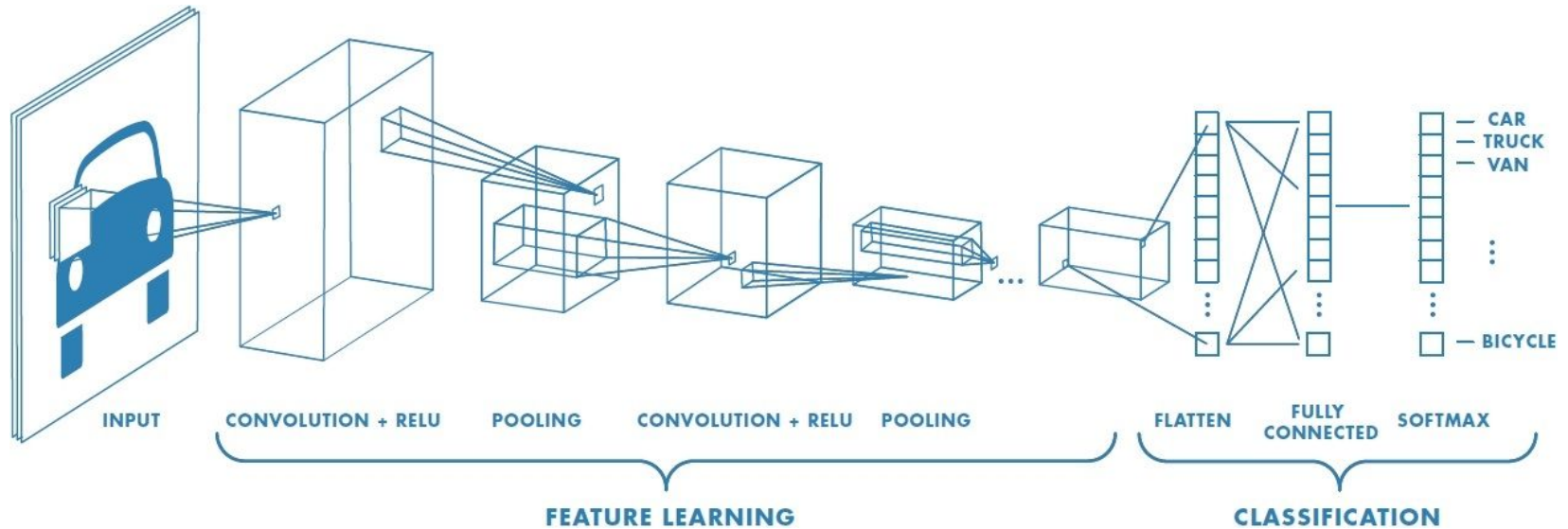


Aktivierungsfunktion

- Zusammenhang zwischen:
 - Netinput \Leftrightarrow Neuron-Aktivitätslevel
- Meist direkt nach CONV-Layer eingesetzt
- ReLU (Rectified Linear Unit)
 - Einfach zu implementieren
 - Schnell (vgl. Sigmoid)
 - Gute Ergebnisse

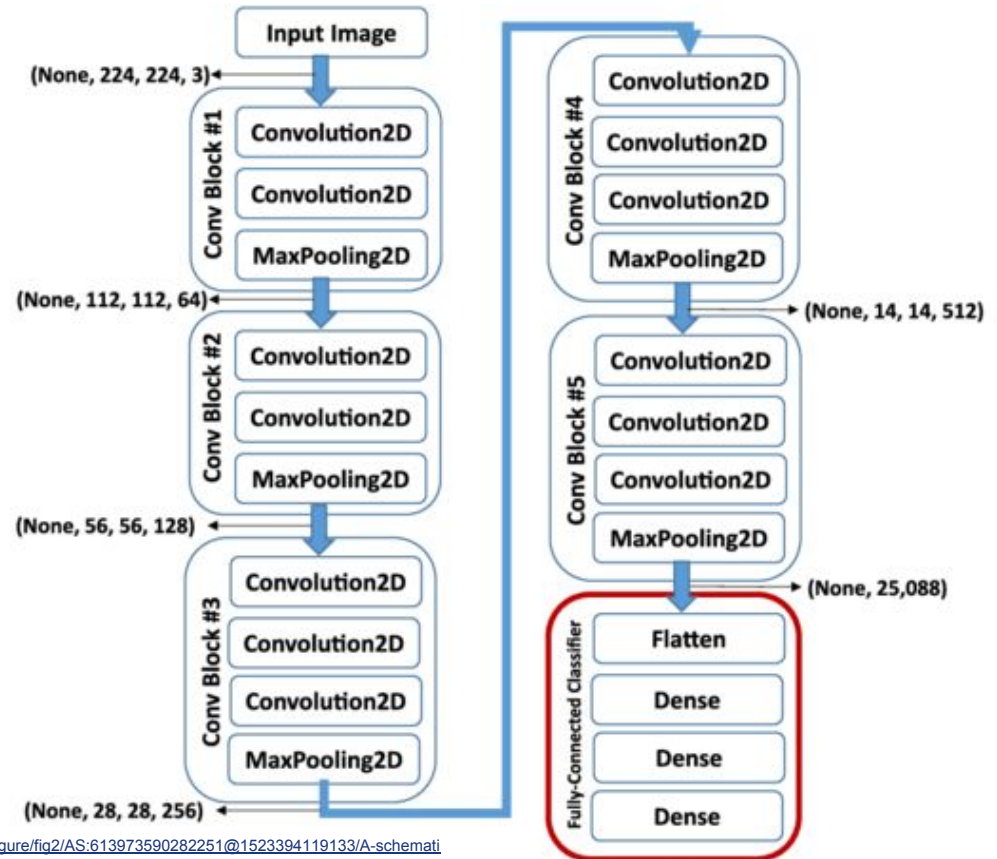


Aufbau



Beispiel: VGG16

- Klassifikationsnetz
- Besteht aus:
 - 13 Conv Layer mit 3x3
 - 5 Max Pooling Layer mit 3x3
 - 3 Dense Layer
 - Softmax

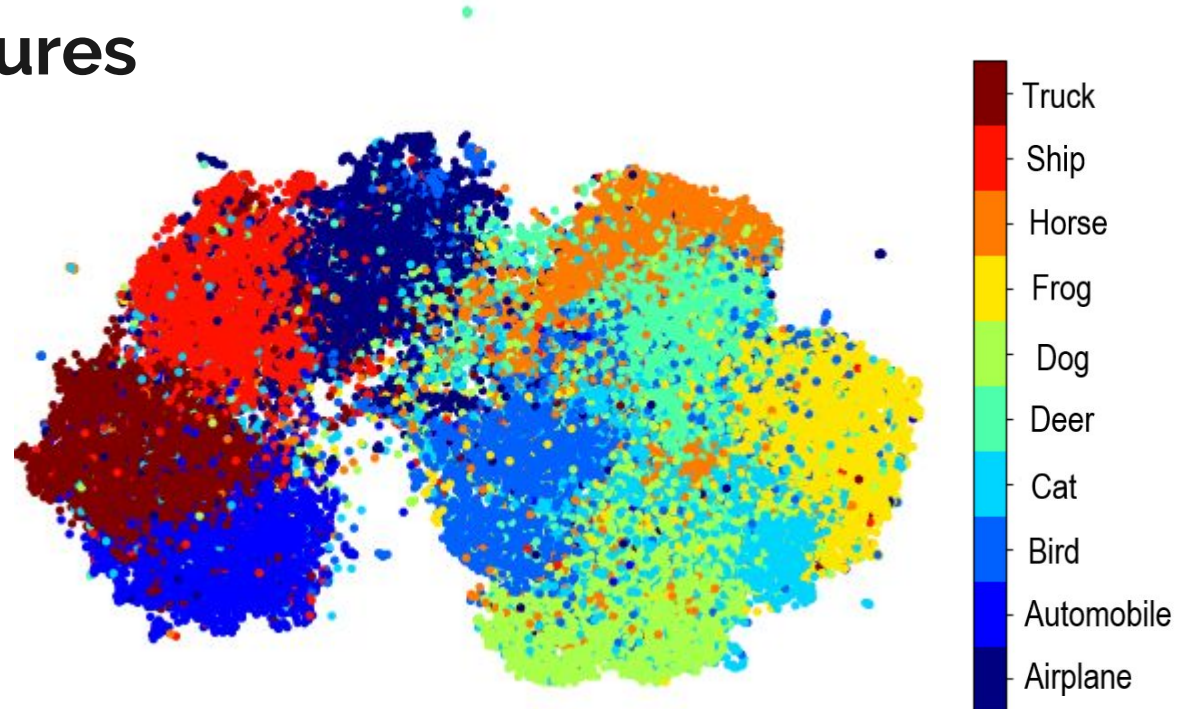


Bildquelle:

https://www.researchgate.net/profile/Kasthurirangan_Gopalakrishnan/publication/319952138/figure/fig2/AS:613973590282251@1523394119133/A-schematic-of-the-VGG-16-Deep-Convolutional-Neural-Network-DCNN-architecture-trained.png

Bottleneck Features

- UMAP
 - Korrelation
- Beobachtungen
 - semantisch ähnliche Objekte nah beieinander
 - geometrisch ähnliche Objekte nah beieinander



Code verfügbar auf
https://github.com/infomon/understanding_cnn

Bildgenerierung: Echt oder Fake?



Bildquelle: <https://arxiv.org/pdf/1812.04948.pdf>

Generative Adversarial Networks

- Generator
 - Arbeitet rückwärts
 - Erstellt Bilder
- Discriminator
 - Unterscheiden: Echt oder Fake?
- Ziel: Discriminator kann nicht mehr zwischen echten und erzeugten Bildern unterscheiden

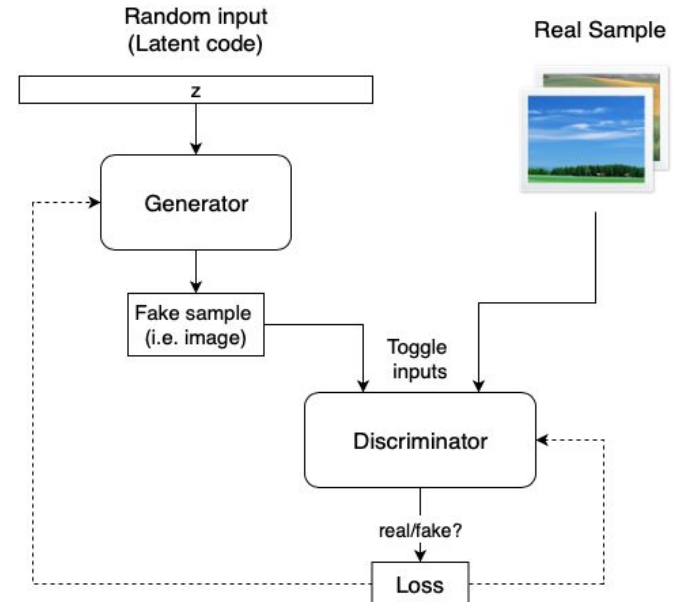


Image-to-Image Translation

Monet \leftrightarrow Photos



Monet \rightarrow photo

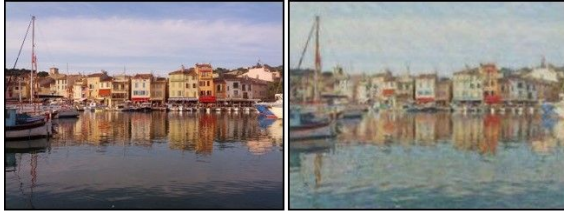


photo \rightarrow Monet

Zebras \leftrightarrow Horses



zebra \rightarrow horse



horse \rightarrow zebra

Summer \leftrightarrow Winter



summer \rightarrow winter



winter \rightarrow summer

Paired Data

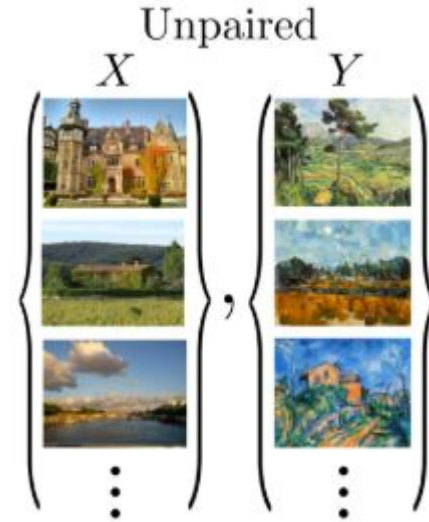
- Trainingsdaten als Paare
- GAN mit Ausgangsbild als Eingabe



Bildquelle: <https://hardikbansal.github.io/CycleGANBlog/>

Unpaired Data

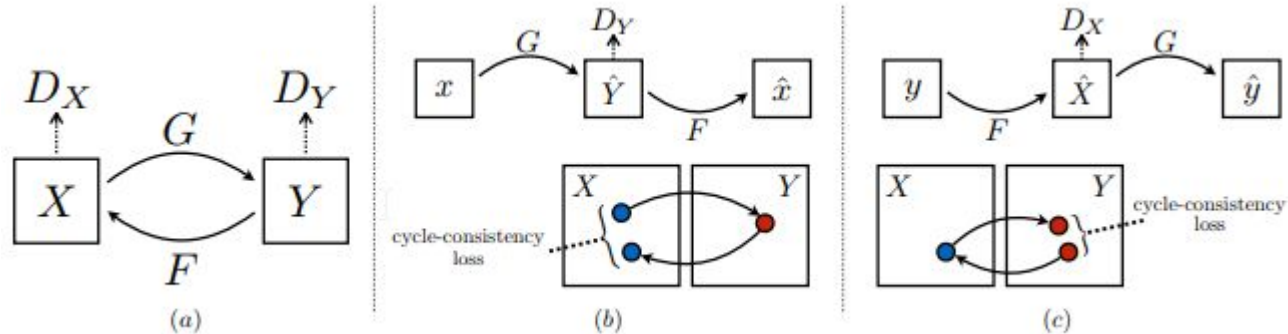
- Keine direkte Zuordnung in den Trainingsdaten
- andere Trainingsmethode benötigt



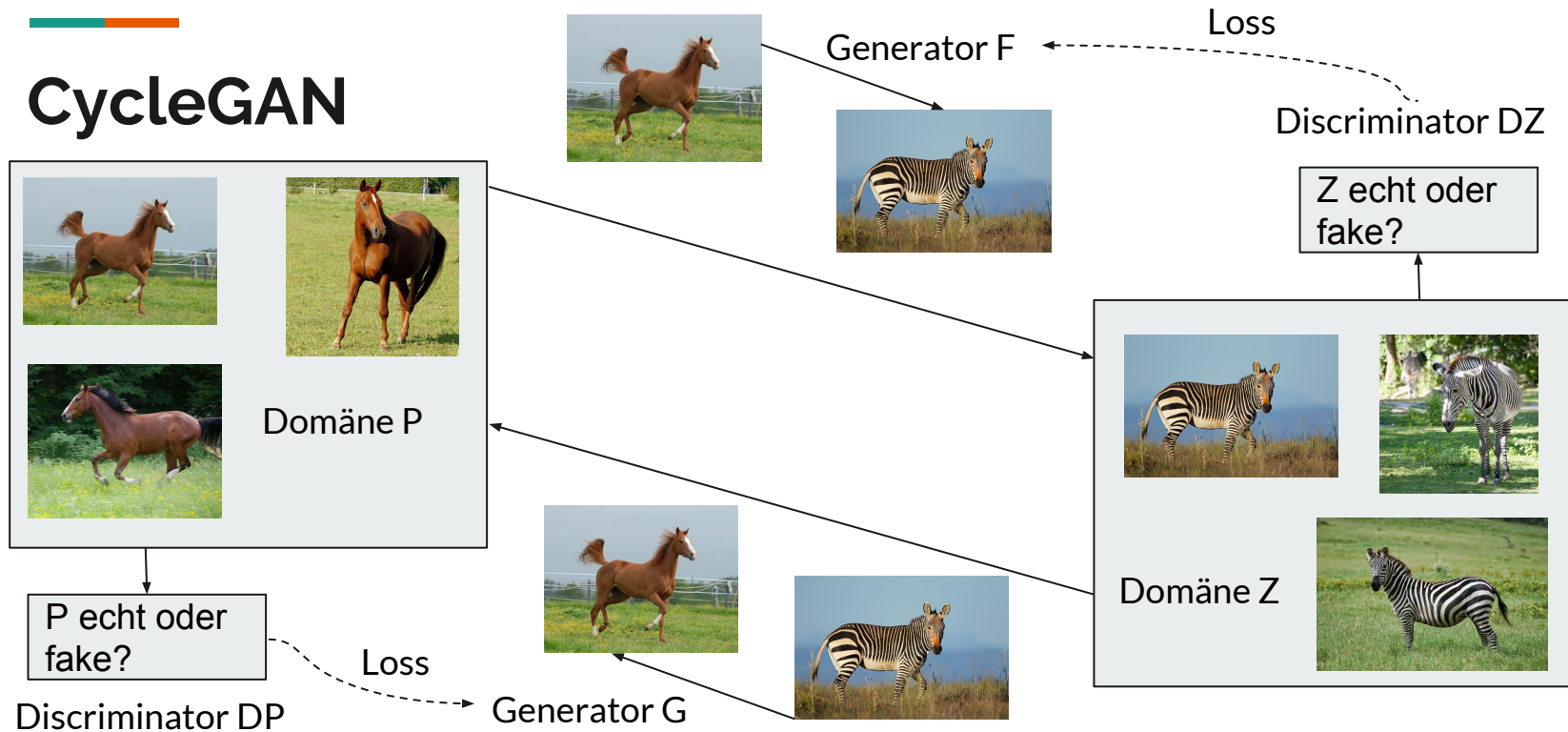
Bildquelle: <https://hardikbansal.github.io/CycleGANBlog/>

CycleGAN

- GAN in beide Richtung
- Paarweise Feedback
- Feedback nach zweifacher Umwandlung



CycleGAN



CycleGAN cycle consistency

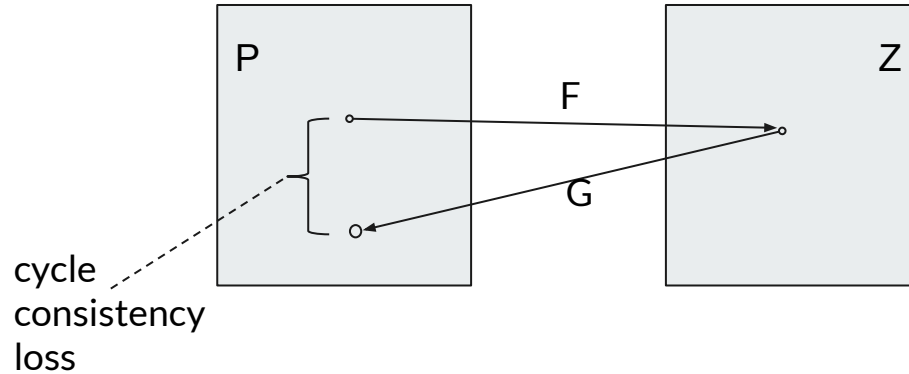


Image-to-Image Translation gone wrong





Literatur

- Fei-Fei Li, Johnsons Justing und Qeung Serena. Convolutional Neural Networks for Visual Recognition. 2018.
- Karen Simonyan und Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. 2014.
- Leland McInnes, John Healy und James Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. 2018.
- Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative Adversarial Networks. 2014.
- Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. 2017