



Neuronale Netze: Deep Convolutional Neural Networks

Jens Ostertag | 6. Juli 2022



universität
uulm

1. Computer lernen lassen

1. Computer lernen lassen

- Allgemeines

- Aufbau neuronaler Netze

- Arten des maschinellen Lernens

- Umgang mit neuronalen Netzen

2. Deep Convolutional Neural Networks

3. Praxisbeispiel

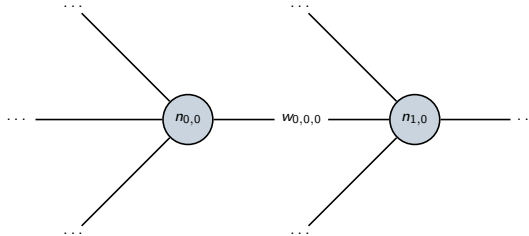
Allgemeines

Wo werden neuronale Netze angewendet?

- Straßenverkehr
- Medizinischer Bereich
- Industrie
- Soziale Netzwerke

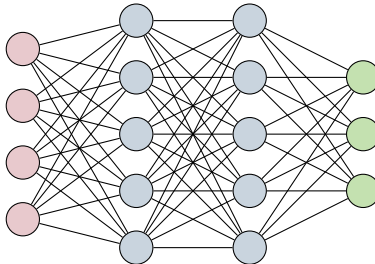
Aufbau neuronaler Netze

- Sehr stark an natürliche neuronale Netze angelehnt
 - Neuronen mit Synapsen verbunden
 - Synapsengewicht regelt Stromfluss zwischen zwei Neuronen



Aufbau neuronaler Netze

- Ausgaben von Neuronen und Synapsengewichte werden als Zahl betrachtet → Berechenbarkeit
- Einteilung der Neuronen in unterschiedliche Schichten
 - Input Layer
 - Hidden Layer
 - Output Layer



Aufbau neuronaler Netze

Berechnung der Ausgabe

$$o_{i,j} = \varphi \left(\sum_{k=0}^{|n_{i-1}|-1} o_{i-1,k} * w_{i-1,k,j} \right)$$

- $n_{i,j}$: Neuron in der Schicht i an der Stelle j
- $o_{i,j}$: Ausgabe des Neurons $n_{i,j}$
- φ : Differenzierbare Aktivierungsfunktion
- $|n_i|$: Anzahl der Neuronen in der Schicht i
- $w_{i,k,j}$: Synapsengewicht zwischen den Neuronen $n_{i,k}$ und $n_{i+1,j}$

Arten des maschinellen Lernens

- Überwachtes Lernen
 - Lernen mit einem vorgegebenen Datensatz
 - Erkennen von Eigenschaften der Daten
 - z.B. zur Klassifikation von Daten
- Unüberwachtes Lernen
 - Erkennen von Ähnlichkeiten in einer Datenmenge
 - z.B. zur Gruppierung von Daten
- Bestärktes Lernen
 - Lernen mit einem Belohnungssystem
 - z.B. zur Durchführung von Aufgaben

Umgang mit neuronalen Netzen

Hier am Beispiel des überwachten Lernens

- Vorbereiten eines Trainingsdatensatzes
 - Beispieldaten werden händisch klassifiziert
 - Evtl. einheitliche Bearbeitung von Daten
 - Erstellen eines Testdatensatzes zur Auswertung
- Trainieren
 - Trainieren mit unterschiedlichen Algorithmen
- Auswertung und Anwendung
 - Auswertung mithilfe des Testdatensatzes
 - Exportieren der Netzstruktur und der Gewichte
 - Importieren in Anwendung

2. Deep Convolutional Neural Networks

1. Computer lernen lassen

2. Deep Convolutional Neural Networks

- Abwandlung neuronaler Netze

- Convolution / Cross Correlation im eindimensionalen Raum

- Padding

- Convolution / Cross Correlation im zweidimensionalen Raum

- Convolutional Layer

- Subsampling

- Aufbau eines Convolutional Neural Networks

3. Praxisbeispiel

Abwandlung neuronaler Netze

- Anwendung zur Bilderkennung
- Warum werden Convolutional Neural Networks benötigt?
 - Zu lange Trainingsdauer
 - Art und Weise der Erkennung eines Objekts
 - Zwei Pixel-Paare sind gleich relevant, unabhängig von Entfernung

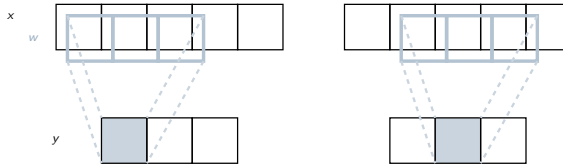
⇒ Risiko des **Overfittings**
- Vorteile:
 - Viel bessere Genauigkeit
 - Lernvorgänge noch ähnlicher zu denen des Menschen
 - Schnelleres Training
 - Weniger anzupassende Gewichte

Convolution / Cross Correlation im eindimensionalen Raum

Berechnung der Cross Correlation

$$y_i = \sum_{k=0}^{|w|-1} x_{i+k} * w_k$$

- y_i : Ausgabe an der Stelle i
- x_i : Eingabe an der Stelle i
- w_i : Wert des Kernels an der Stelle i

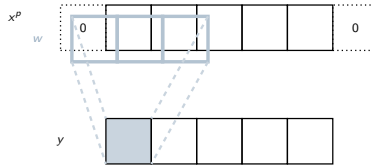


Padding

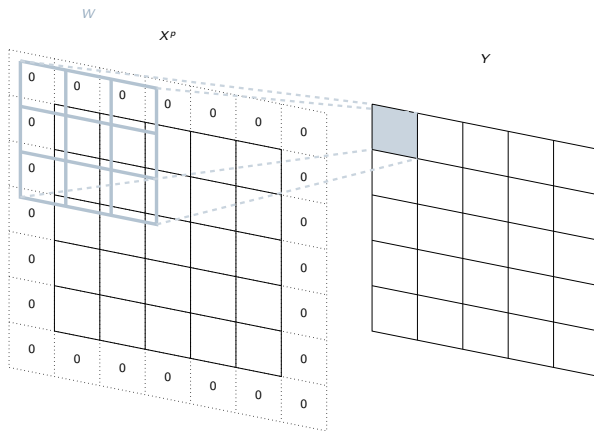
Problem: Ausgabe wird kleiner als Eingabe

⇒ Detailverlust

Lösung: Eingabe mit Nullen erweitern



Convolution / Cross Correlation im zweidimensionalen Raum

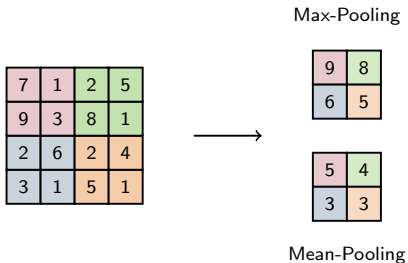


Convolutional Layer

- Mehrere Eingabechannel (zum Beispiel unterschiedliche Farbkanäle)
- Mehrere Ausgabechannel
- Jeder Eingabechannel hat Einfluss auf jeden Ausgabechannel
 - ⇒ Mehrere Kernel pro Convolutional Layer
- Ausgabe eines Convolutional Layer: Feature Map
 - Allgemeine Eigenschaften in höheren Schichten
 - Komplexere Eigenschaften in tieferen Schichten

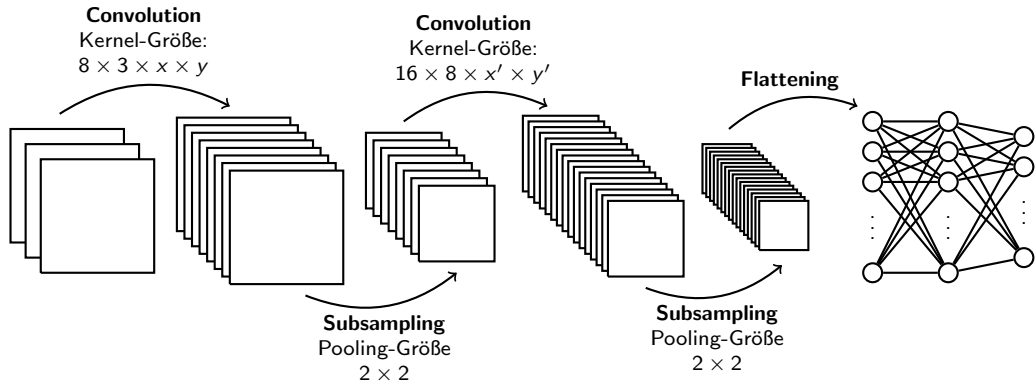
Subsampling

- Verringerung der Eigenschaften durch Pooling-Operationen
- Schnellerer Trainingsvorgang, nur geringer Leistungsverlust



- Keine anzupassenden Gewichte

Aufbau eines Convolutional Neural Networks



3. Praxisbeispiel

1. Computer lernen lassen
2. Deep Convolutional Neural Networks
- 3. Praxisbeispiel**
Ziffernerkennung

Ziffernerkennung

MNIST-Datensatz

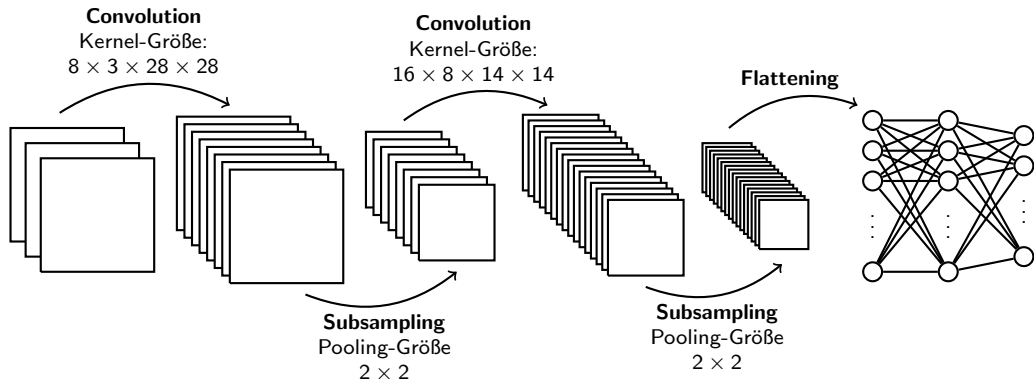
- Handgeschriebene Ziffern von 0 bis 9
- Schwarz-Weiß-Bilder der Größe 28×28

TensorFlow

- ML-Framework von Google
- Viele Architekturen neuronaler Netze
 - Allgemeine Datenverarbeitung
 - Bilderkennung
 - Texterkennung
 - Audioerkennung
 - Verstärktes Lernen
- Großer Komfort
 - Einfaches, aber genaues Training
 - GPU-Berechnungen

Ziffernerkennung

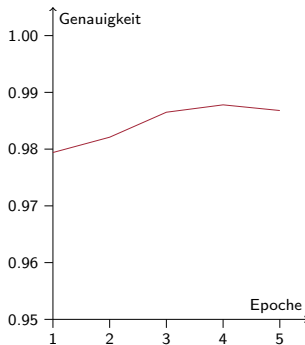
Aufbau des Netzes:



Ziffernerkennung

Ergebnisse:

- Erreichte Genauigkeit: 98.68% (nach fünf Epochen)



- Geringer Unterschied zu vollständig verbundenem Netz (97.5%)

Literaturverzeichnis

- Robert Callan: *Neuronale Netze im Klartext*. Pearson Studium, 1. Auflage, 2003.
- Sebastian Raschka und Vahid Mirjalili: *Python Machine Learning*. 3. Auflage, 2015. Kapitel 1 und Kapitel 15.
- Manfred Spitzer: *The Mind Within the Net: Models of Learning, Thinking and Acting*, Seite 21. MIT Press, 2. Auflage, 1999. Verweis von Wikipedia / Synapsengewicht.