Wissensentdeckung in Datenbanken

Deep Learning

Nico Piatkowski und Uwe Ligges

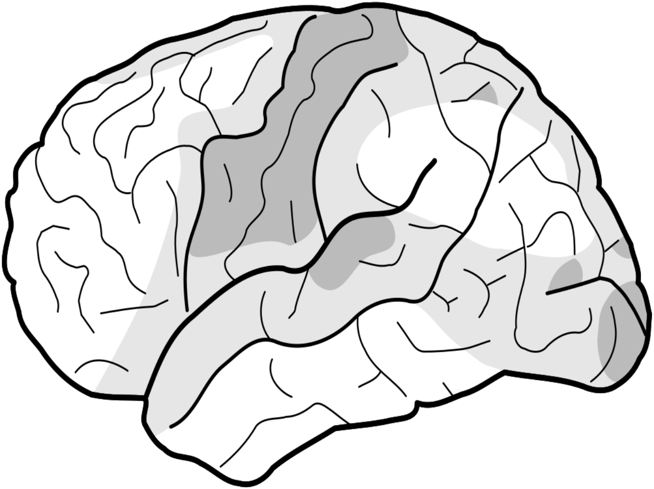
Informatik—Künstliche Intelligenz
Computergestützte Statistik
Technische Universität Dortmund

20.07.2017

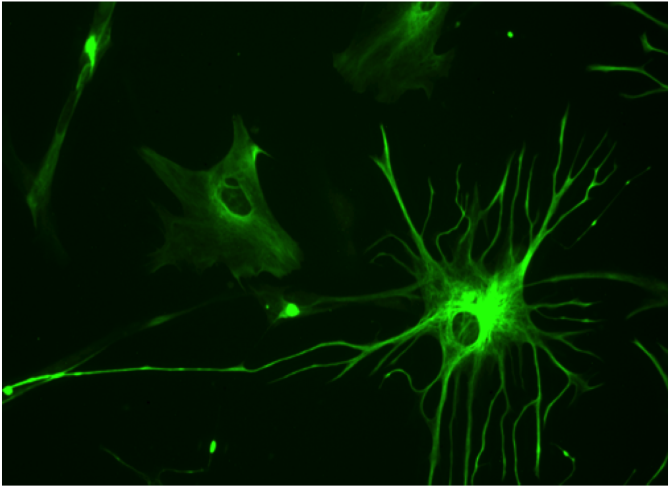
Überblick

- Künstliche Neuronale Netze
  - Motivation
  - Formales Modell
  - Aktivierungsfunktionen
  - Vorhersage
  - Parameterscätzung
- Varianten
  - Faltungsnetze
  - Rekurrente Netze
  - Autoencoder

Motivation

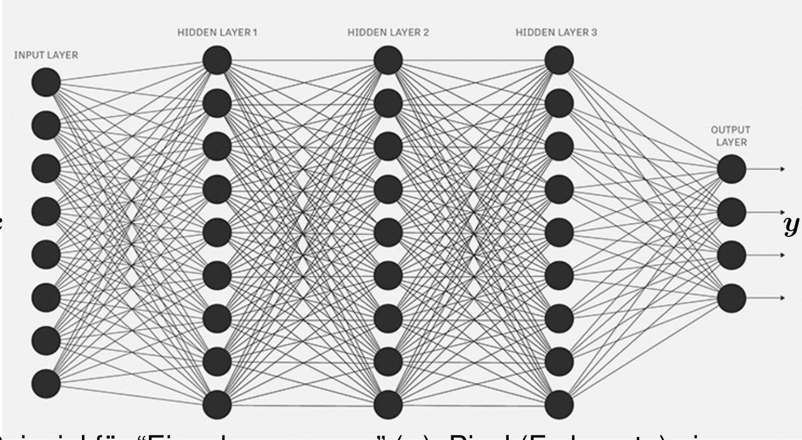


Motivation



Motivation (Deep Vs. Not-Deep)

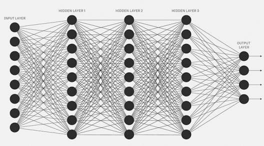
$\boldsymbol{x}$



$\boldsymbol{y}$

Beispiel für “Eingabeneuronen” ($\boldsymbol{x}$): Pixel (Farbwerte) einer Bilddatei, Wortvektoren (bag-of-words, TF-IDF), ...

Formalisierung



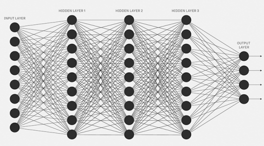
Künstliches Neuronales Netz f:

- Graph $G = (V, E)$. Knoten heißen **Neuronen**
- Jedes Neuron liegt in einer **Schicht** $L : V \to \{0, 1, 2, \dots, K\}$,
- Eine **Aktivierungsfunktion** pro Knoten: $a_v : \mathbb{R} \to \mathbb{R}$, $\forall v \in V$
- Schicht 0 heißt **Eingabeschicht**, Schicht K heißt **Ausgabeschicht**

Jedes Neuron v auf Schicht $k > 0$ (d.h. $L(v) = k$) repräsentiert eine Funktion $v : \mathcal{X}_v \to \mathcal{Y}_v$:

- Eingabe ist eine gewichtete Summe der Ausgaben alle Neuronen auf Schichte $k - 1$
- Ausgabe ist eine Reele Zahl

Formalisierung



Künstliches Neuronales Netz f:

- Graph $G = (V, E)$. Knoten heißen **Neuronen**
- Jedes Neuron liegt in einer **Schicht** $L : V \to \{0, 1, 2, \ldots, K\}$,
- Eine **Aktivierungsfunktion** pro Knoten: $a_v : \mathbb{R} \to \mathbb{R}$, $\forall v \in V$
- Schicht 0 heißt **Eingabeschicht**, Schicht K heißt **Ausgabeschicht**

Jedes Neuron v auf Schicht $k > 0$ (d.h. $L(v) = k$) repräsentiert eine Funktion $v : \mathcal{X}_v \to \mathcal{Y}_v$:

- Eingabe ist eine gewichtete Summe der Ausgaben alle Neuronen auf Schichte $k - 1$
- Ausgabe ist eine Reele Zahl

Formalisierung (II)

Jedes Neuron v auf Schicht $L(v) > 0$ repräsentiert eine Funktion $v : \mathcal{X}_{k-1} \to \mathcal{X}_v$

Notation: Ausgabe von Neuron v ist $\text{out}(v)$.

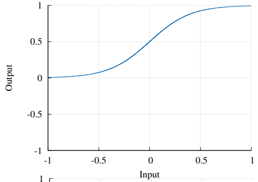
$$\text{out}(v) = a_v \left(\sum_{w:L(w)=L(v)-1} \beta_{vw} \text{out}(w) \right)$$

- $\mathcal{X}_{k-1}$ ist der gemeinsame Ausgaberaum aller Neuronen auf Ebene $k-1$
- $\mathcal{X}_v$ ist der Ausgaberaum von v
- Dann ist das gesamte Netz eine Funktion $f : \mathcal{X}_0 \to \mathcal{X}_K$

Aktivierungsfunktionen an Neuron $v \in V$

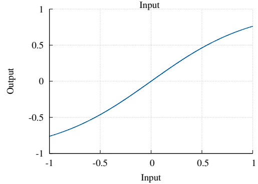
Sigmoid:

$$a_v^{\text{sig}}(z) = \frac{1}{1 + \exp(-z)}$$



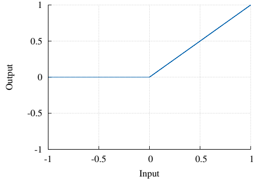
Tangens Hyperbolicus (tanh):

$$a_v^{\text{tanh}}(z) = \frac{\exp(z) - \exp(-z)}{\exp(z) + \exp(-z)}$$



Rectified linear unit (ReLU):

$$a_v^{\text{ReLU}}(z) = \max\{0, z\}$$



Parameter lernen

Neuronales Netz ist eine Modellfunktion f.

Grundsätzlich sind alle aus der Vorlesung bekannten Verlustfunktionen $\ell(f, \mathcal{D})$ möglich.

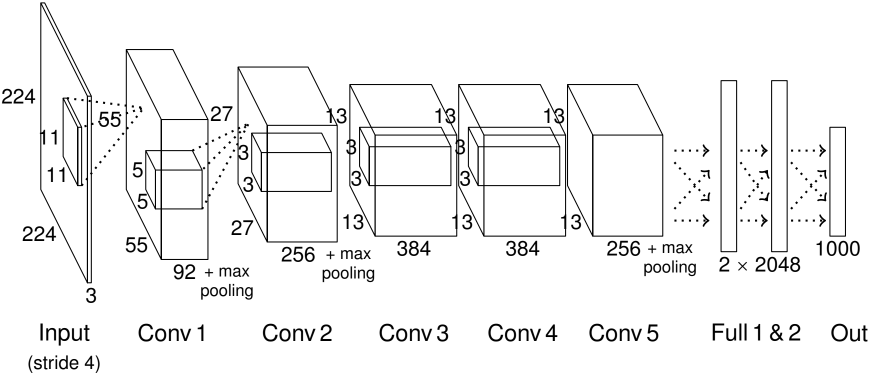
Allgemein:

- Die Anzahl der Parameter eines Neurons v ist gleich der Anzahl der Neuronen in Schicht $L(v) - 1$
- Berechnung des Gradienten $\nabla \ell(f, \mathcal{D})$ durch Anwendung der Kettenregel (Backpropagation)
- Optimierung mittels stochastischem Gradientenabstieg

$$\boldsymbol{\beta}^{t+1} = \boldsymbol{\beta}^{t} + \eta \nabla \ell(\boldsymbol{\beta}^{t}, S)$$

$S \subset \mathcal{D}$, Extremfall: $S = \{(\boldsymbol{x}, \boldsymbol{y})\}$ (nur ein Datenpunkt pro Gradientenschritt)

Beispiel: “AlexNet”



- Gewinner der ImageNet 2012 Challenge

Representationslernen mittel Faltung

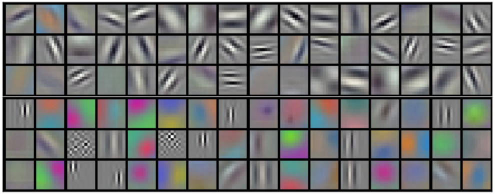


Abbildung: Gelernte Filter in AlexNet Schicht “Conv1”: Gabor Filter & Color Blobs

Beobachtung: Tiefe des Netzes erhöht Abstraktionsgrad der internen Representation

- Erste Schichten: **Allgemein**, lokale Representation
- Tiefe Schichten: **Spezifisch**, globale Representation