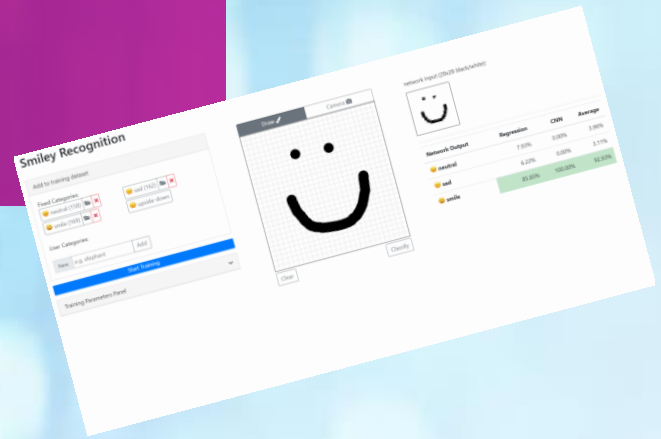
20 JAHRE ITERATEC 1996-2016

iteratec

KOMPETENZ, DIE ENTLASTET

TIP: Zur Vorbereitung gerne schon mal runterladen: http://scytec.de/ml-with-a-smile.zip

# Machine Learning mit einem Smile




Yves Schubert

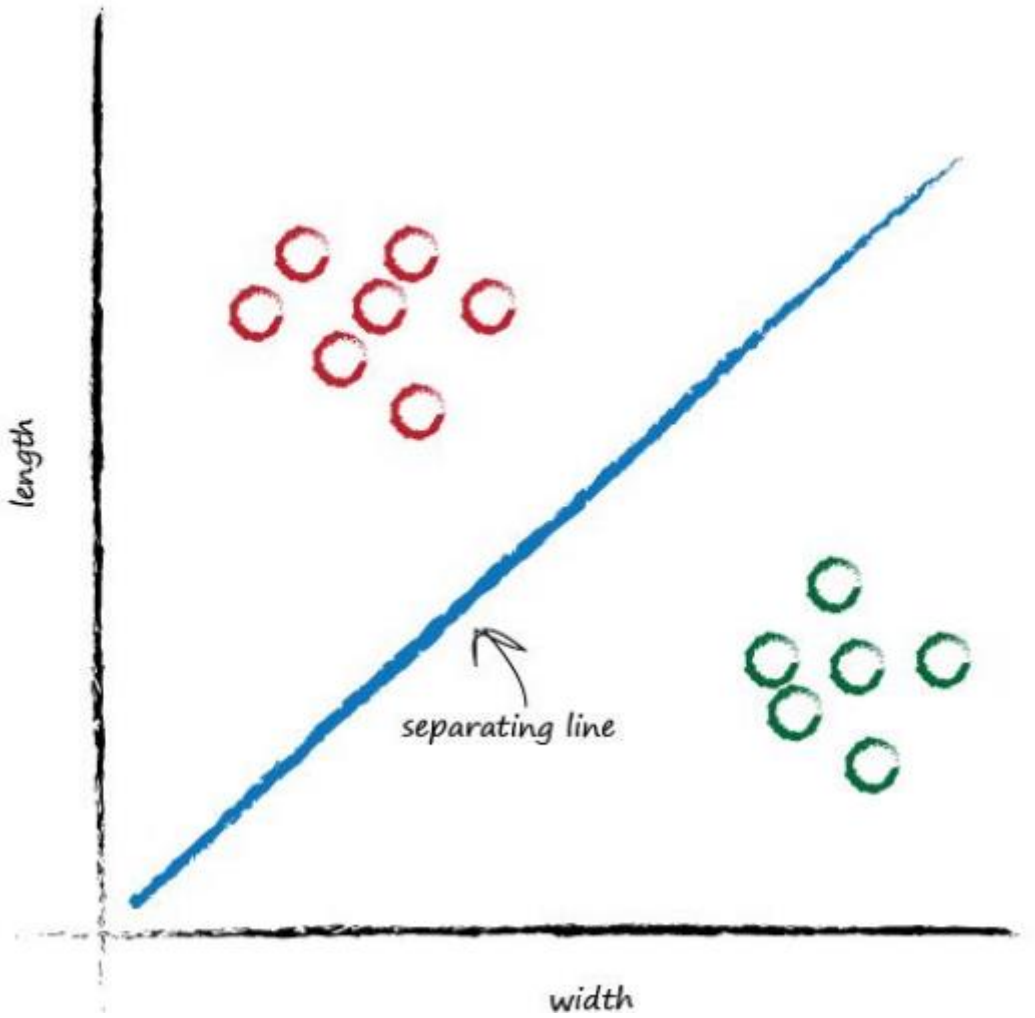
Alexander Blehm

# Agenda - Was machen wir heute?

- Kleine Einführung in neuronale Netze (optional)
- Installation
- Zeichenerkennung mit einem neuronalen Netzwerk
  - IDE und Webanwendung zum Laufen bringen
  - Testdaten erzeugen
  - Netzwerke trainieren
  - Testen
  - Netzwerke Tweaken
  - Rumspielen
- Testdaten sammeln und zusammenwerfen?

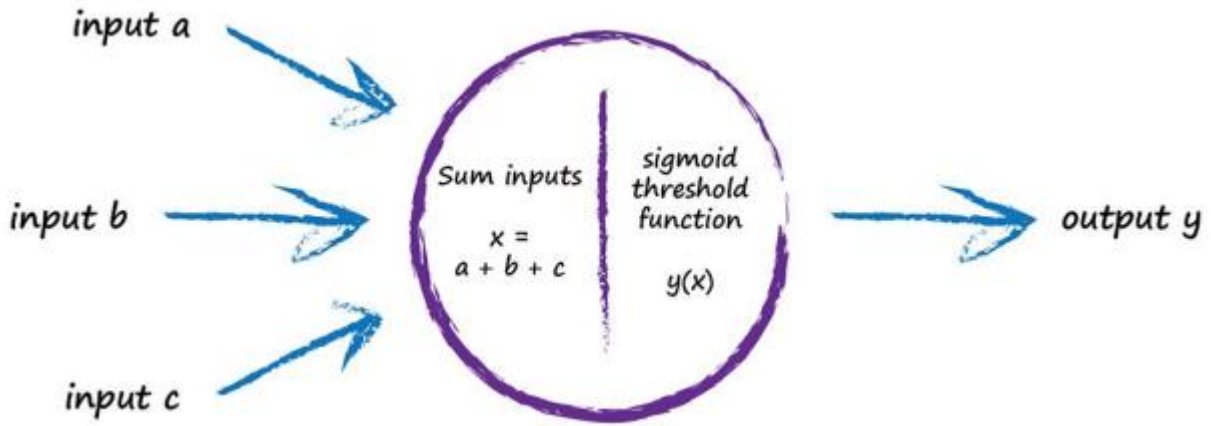
# Was sind neuronale Netze und wofür braucht man das?

- Problem: Klassifikation und Mustererkennung ist mit klassischen Algorithmen direkt nur sehr schwierig machbar.
- In den 40er und 50er Jahren fand man heraus, dass sich neuronale Netze sehr gut zur Klassifikation von Daten eignen
- Das Ziel ist es, eine oder mehrere Funktionen zu finden, die die einzelnen Klassen voneinander trennt



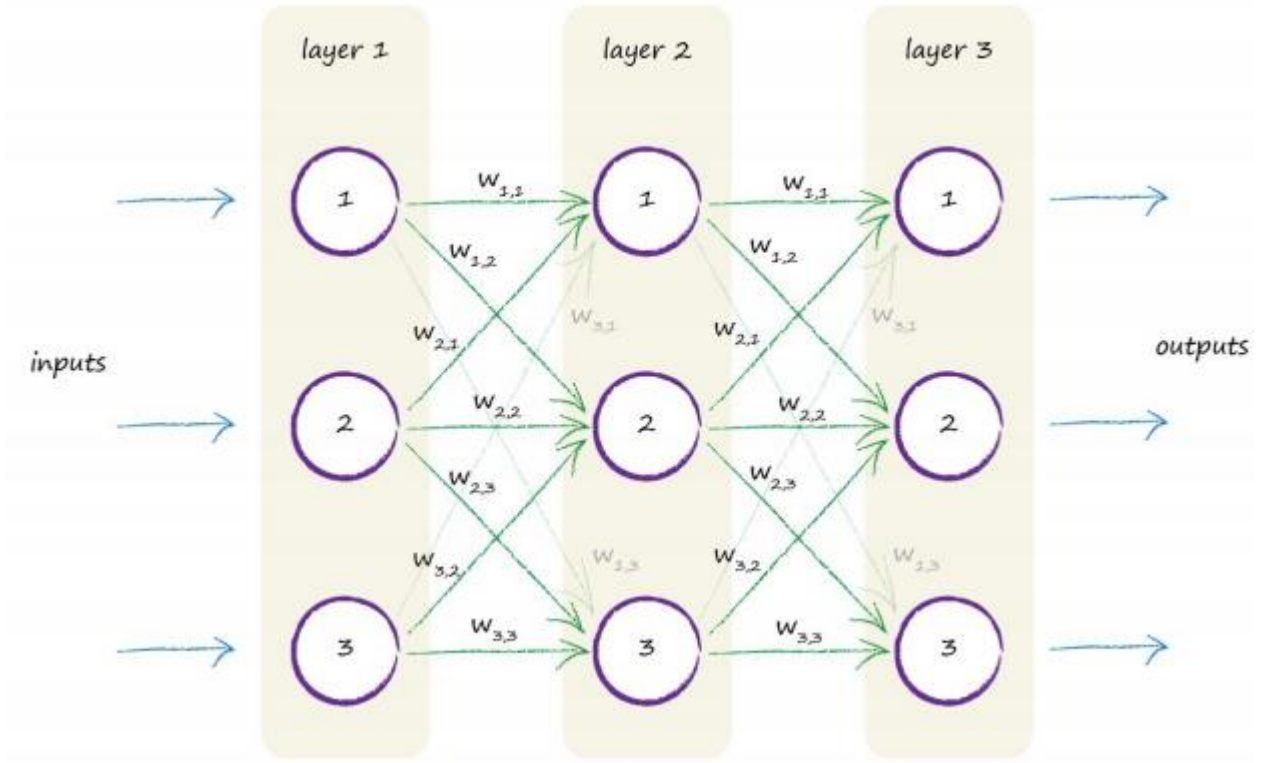
# Ein Neuron

- Inputs: Eingabe-Daten
- Aktivierungsfunktion: Das Neuron verarbeitet (in der Regel) die Summe der Eingabewerte zu einem Ausgangswert z.B. mit der sigmoid Funktion



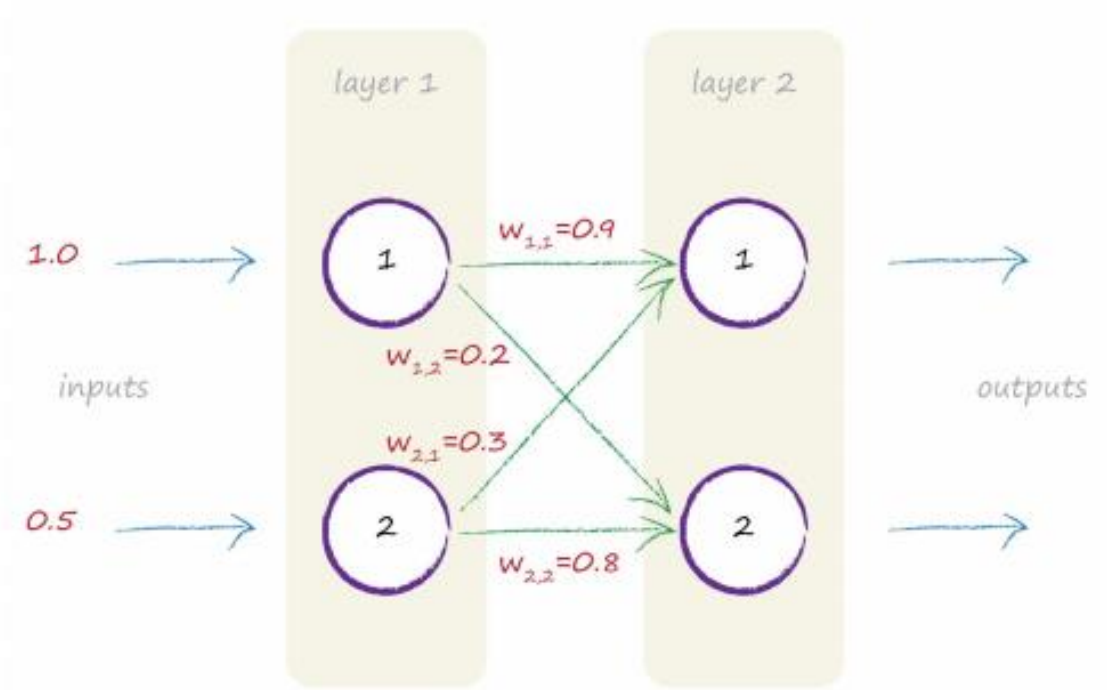
# Was ist nun also ein Neuronales Netz?

- Inputs: Eingabe-Daten (z.B. ein Bild)
- Neuronen: Nehmen beliebig viele Eingabe-Daten entgegen und geben eine Kombination (z.B. Summe, etc...) wieder aus
- Layer: Die Neuronen sind in Layern angeordnet (es gibt unterschiedliche Typen)
- Verbindungen: Die Neuronen können beliebig untereinander Verbunden werden. Die Verbindungen haben eine Gewichtung, die die Ausgabe eines vorhergehenden Neurons entsprechend verstärken oder abschwächen.
- Outputs: Ausgabe-Daten (z.B. der Name eines Gegenstands in dem Eingabebild)



# Gewichtungen der Verbindungen

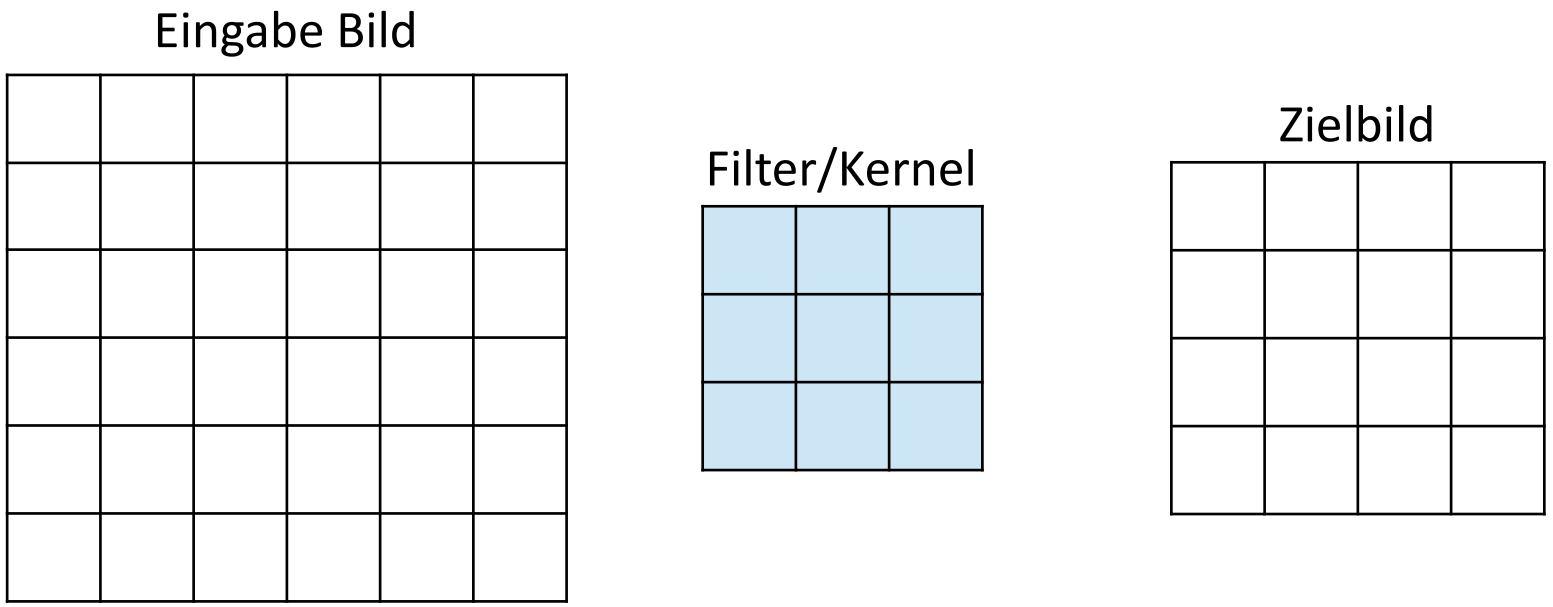
- Anhand der Gewichtung wird der Ausgabewert eines vorhergehenden Neurons verstärkt oder abgeschwächt
- Die Gewichtung hat den größten Einfluss bei der Lernfähigkeit!



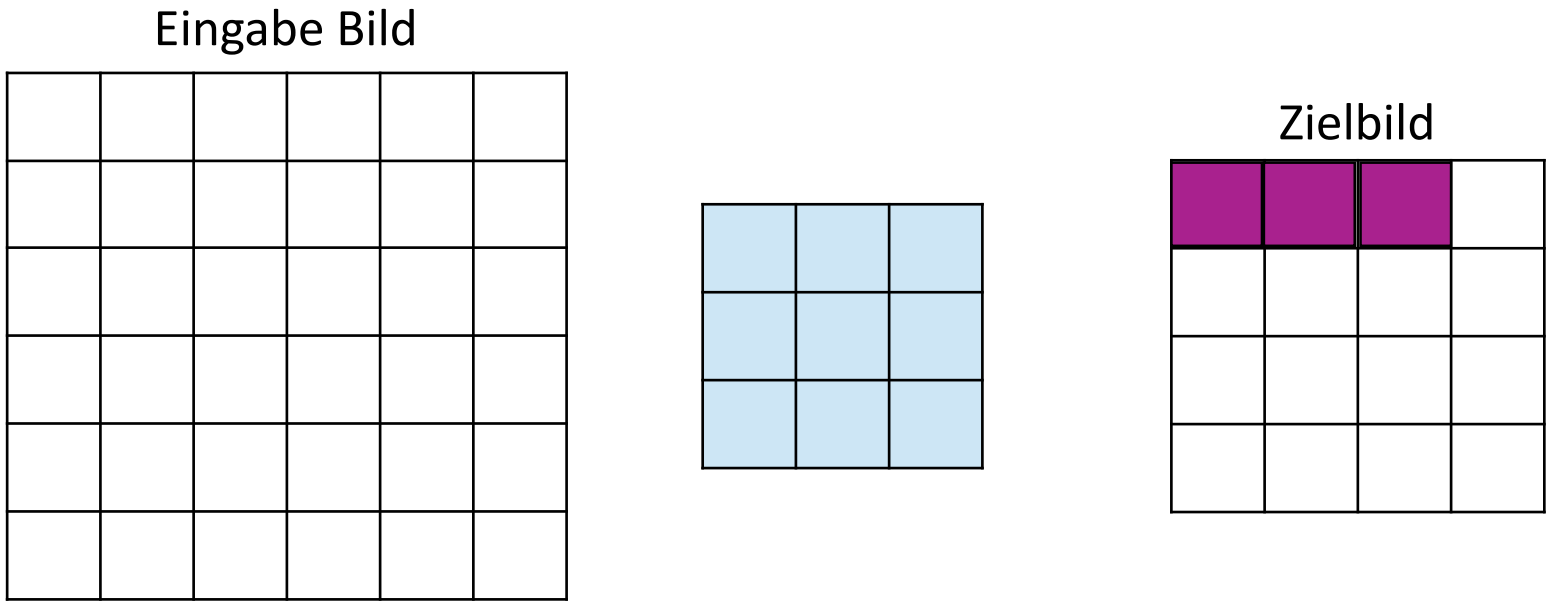
# Convolutional Neuronal Network - CNN

- Zur Klassifizierung (Erkennung) von Objekten in Bildern => Bilderkennung
- Convolution bedeutet: Faltung
  - Etwa: Mathematisches Verfahren um auf jeden Wert einer Funktion eine weitere Funktion „loszulassen“
- Ist letztlich ein Hilfsmittel, die Anzahl der Trainingsdaten zu verkleinern

# Convolutional Neuronal Network - CNN

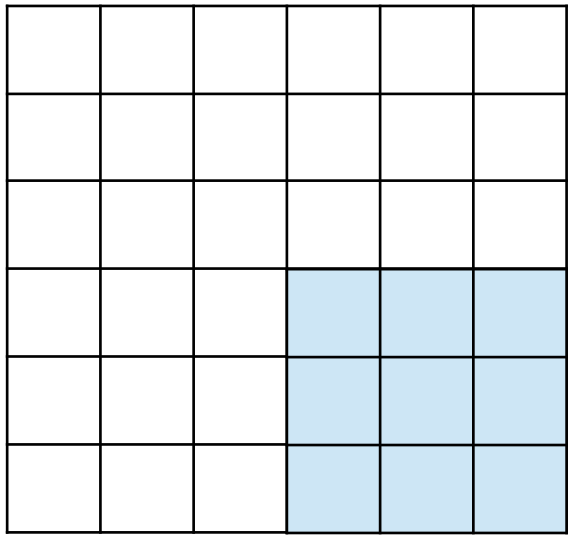


# Convolutional Neuronal Network - CNN

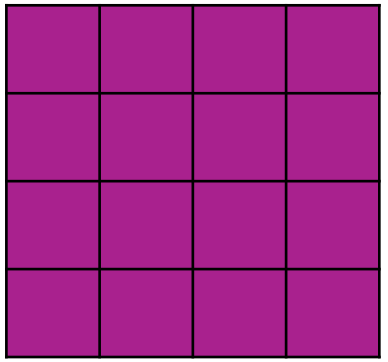


# Convolutional Neuronal Network - CNN

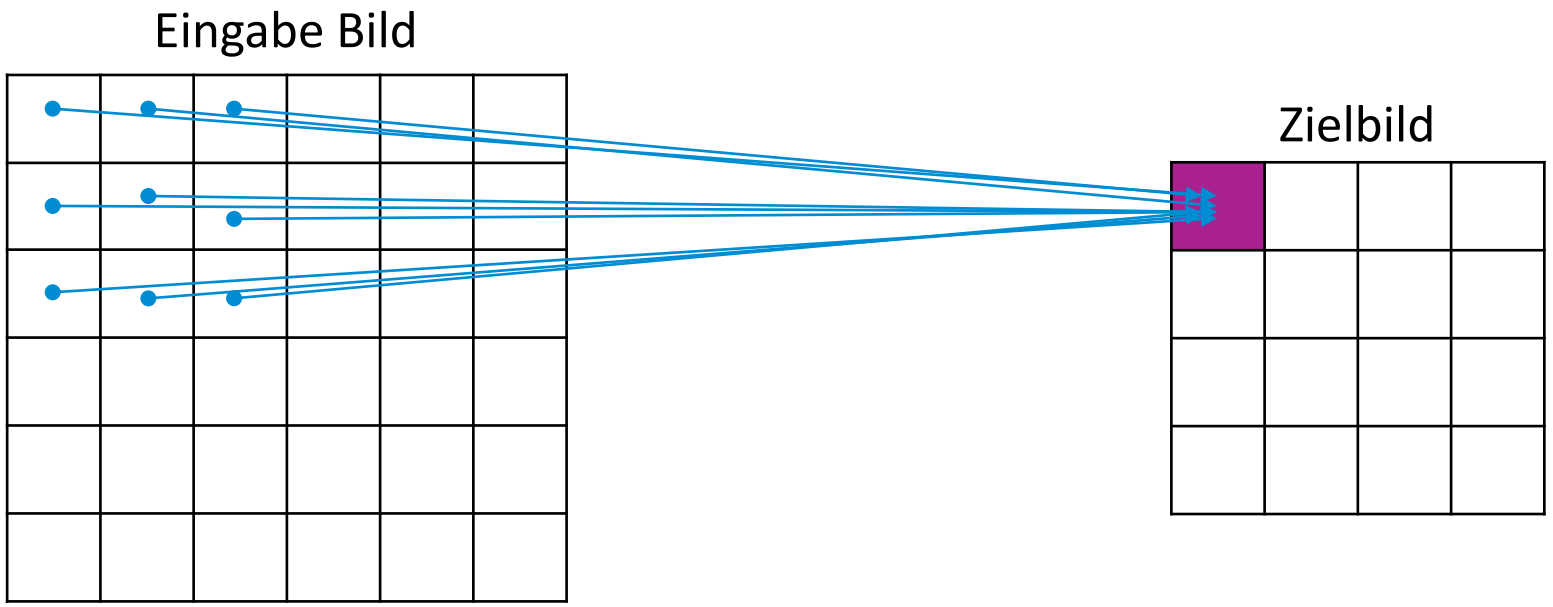
Eingabe Bild



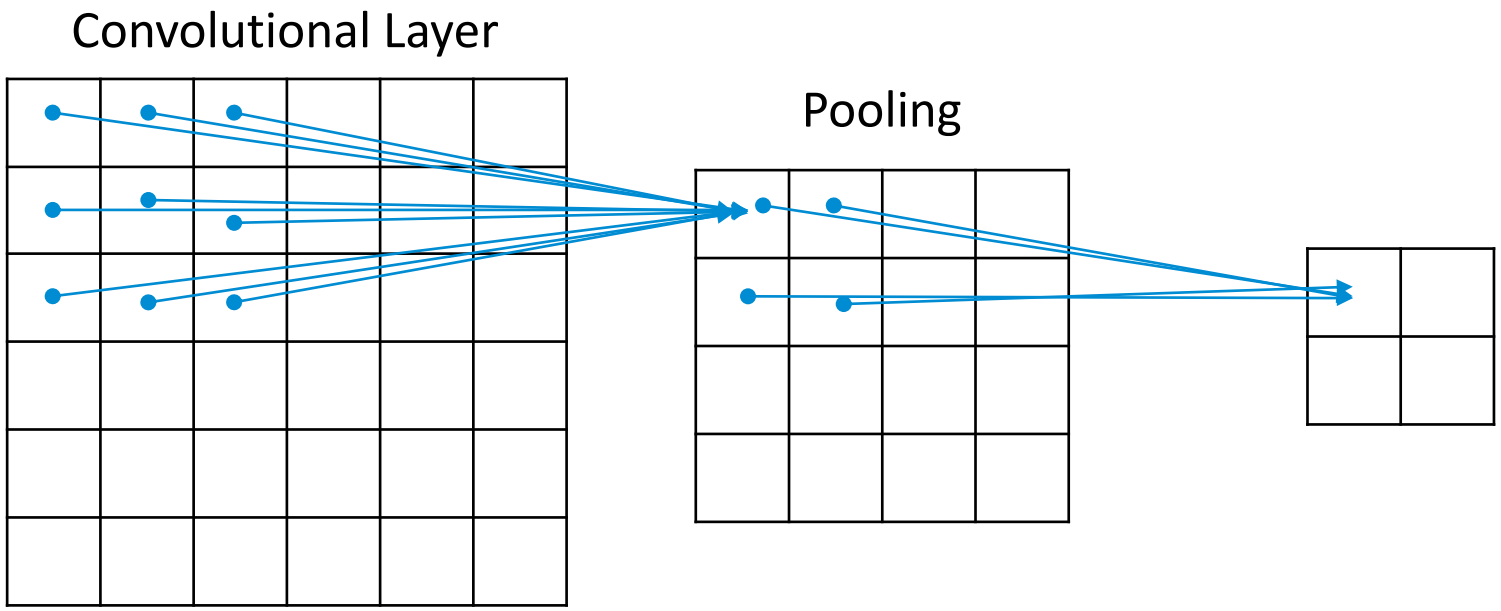
Zielbild



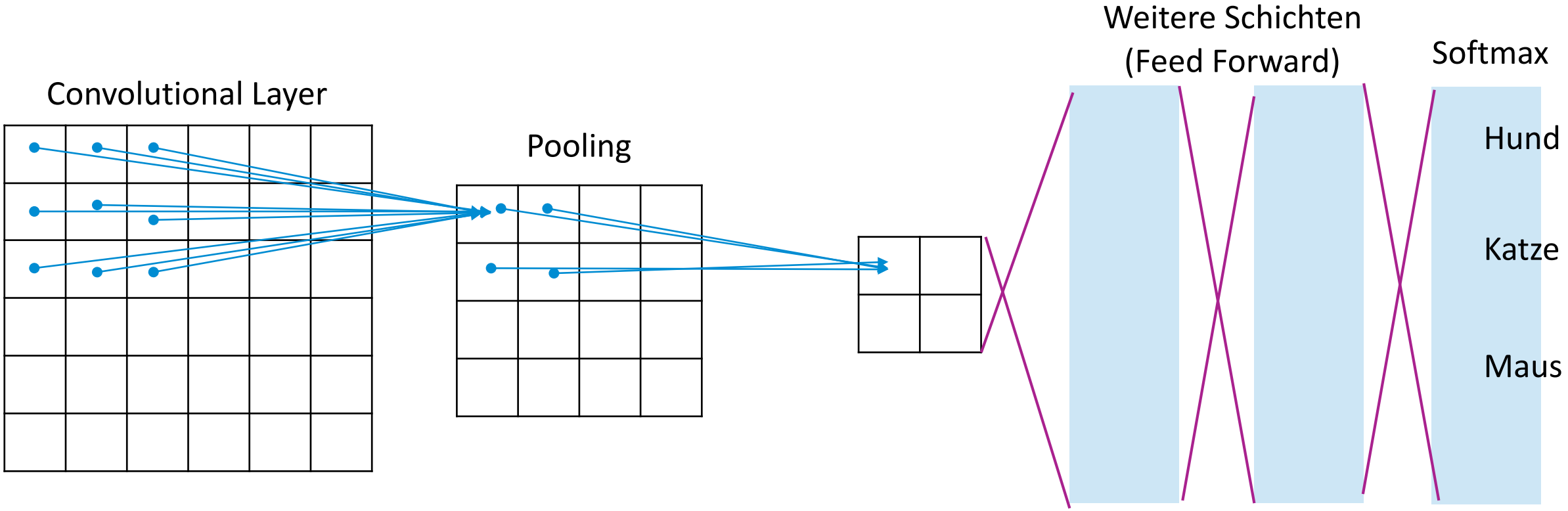
# Convolutional Neuronal Network - CNN



# Convolutional Neuronal Network - CNN



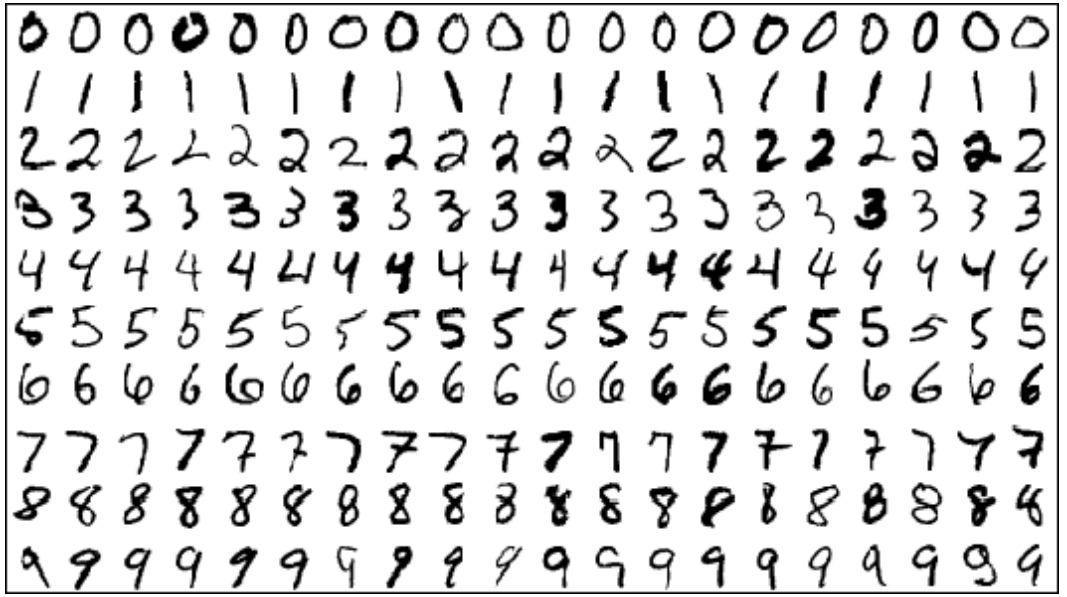
# Convolutional Neuronal Network - CNN



# (M)NIST

## Trainings- / Testdaten zu Handschriftenerkennung

- (modified) National Insitute for Standards and Technology – Dataset
- Handschriftliche Daten von Schülern und Behördenmitarbeitern
- 60.000 Trainingsdatensätze
- 10.000 Testdatensätze
- Ursprünge gemischt

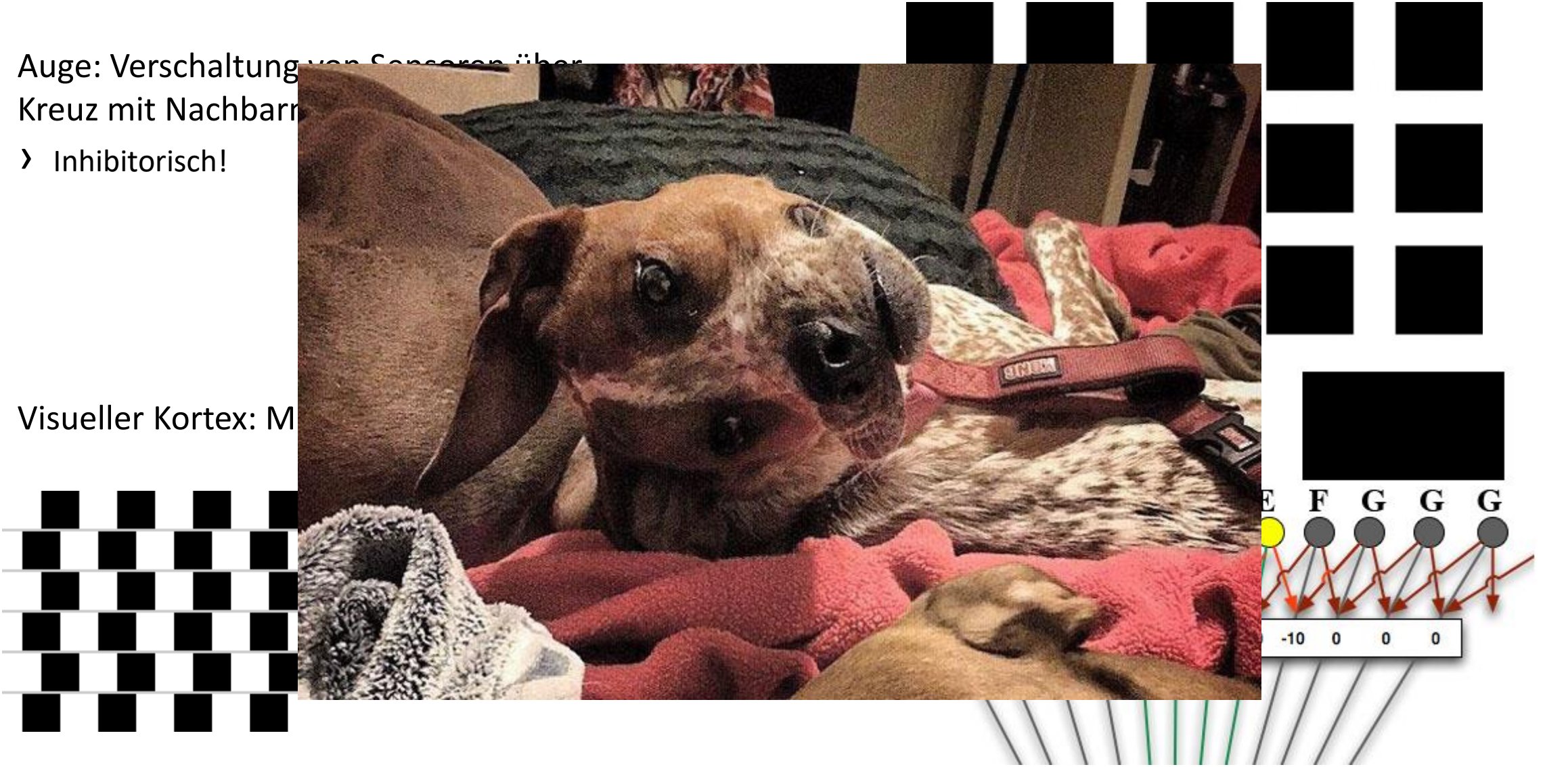


- https://en.wikipedia.org/wiki/MNIST_database

# Menschliche Bildverarbeitung

## Grob vereinfacht – mit Beispielen

- Auge: Verschaltung von Sensoren über Kreuz mit Nachbarn
  - Inhibitorisch!
- Visueller Kortex: M



# Vorbereitung/Installation

Variante 1: From Zip (nur Windows!)

- Zip herunterladen:
  - http://scytec.de/ml-with-a-smile.zip
- Zip entpacken
- Eingabeaufforderung/Shell öffnen
- cd smiley
- ../python/python main.py

Variante 2: From Scratch

- (Git installieren)
- Github Repository klonen:
  - git clone https://github.com/scyv/Smiley.git
- Anleitung auf https://github.com/scyv/Smiley.git befolgen (Abschnitt „Installation“)
  - Python Installieren (mind. 3.6) (am besten Miniconda!)
  - Eingabeaufforderung/Shell öffnen
  - Conda Environment öffnen
  - Mit pip install –r requirements.txt die Abhängigkeiten installieren
  - Starten mit: python main.py

# Workshop



1. Testbilder malen (mind. 7 pro Kategorie)
2. einmal auf „Start Training“ klicken (dauert ca. 3 mins)
3. Sollte jemand die Parameter angepasst haben und weiß die Default Werte nicht mehr: Blick in die „config.ini“ Datei werfen
4. Wenn Training fertig ist -> Ein beliebiges Bild aus den Kategorien malen und per „Classify“ das Programm raten lassen. Vergleiche CNN Werte vs Regression.
5. Wer schon 1-2 mal trainiert hat und fertig ist mit 1) – 3) einfach zurücklehnen und gelangweilt die Decke anschauen… dann machen wir weiter

# Mögliche Kategorien

- Einfach
  - Smiley
    - neutral
    - smile
    - sad
    - upside-down
    - crying
    - angry
    - cool
- Mittel
  - Buchstaben
    - A-Z (Großbuchstaben)
- Schwer
  - Tiere
    - pig
    - bunny
    - cat
    - crocodile
    - giraffe
    - mouse
    - bear

1. Testbilder malen (mind. 7 pro Kategorie)
2. einmal auf „Start Training“ klicken (dauert ca. 3 mins)
3. Sollte jemand die Parameter angepasst haben und weiß die Default Werte nicht mehr: Blick in die „config.ini“ Datei werfen
4. Wenn Training fertig ist -> Ein beliebiges Bild aus den Kategorien malen und per „Classify“ das Programm raten lassen. Vergleiche CNN Werte vs Regression.
5. Wer schon 1-2 mal trainiert hat und fertig ist mit 1) – 3) einfach zurücklehnen und gelangweilt die Decke anschauen… dann machen wir weiter

- Dokumentation: https://www.tensorflow.org/versions/r1.13/api_docs/python/tf/layers
- Optional: Daten hochladen zu: https://github.com/scyv/smiley_data

# Spaß mit AI

- https://quickdraw.withgoogle.com/
- https://experiments.withgoogle.com/collection/ai
- http://yann.lecun.com/exdb/mnist/
- https://playground.tensorflow.org

iteratec

KOMPETENZ, DIE ENTLASTET

# Viel Spaß!

www.iteratec.de