DHBW CAS
Duale Hochschule
Baden-Württemberg
Center for Advanced Studies

Convolutional Neural Networks

Gennaro Izzo, Sofie Sharaf

W3M20026.1 – Data Science: Processes and Algorithms

Mannheim, 28.Februar 2024

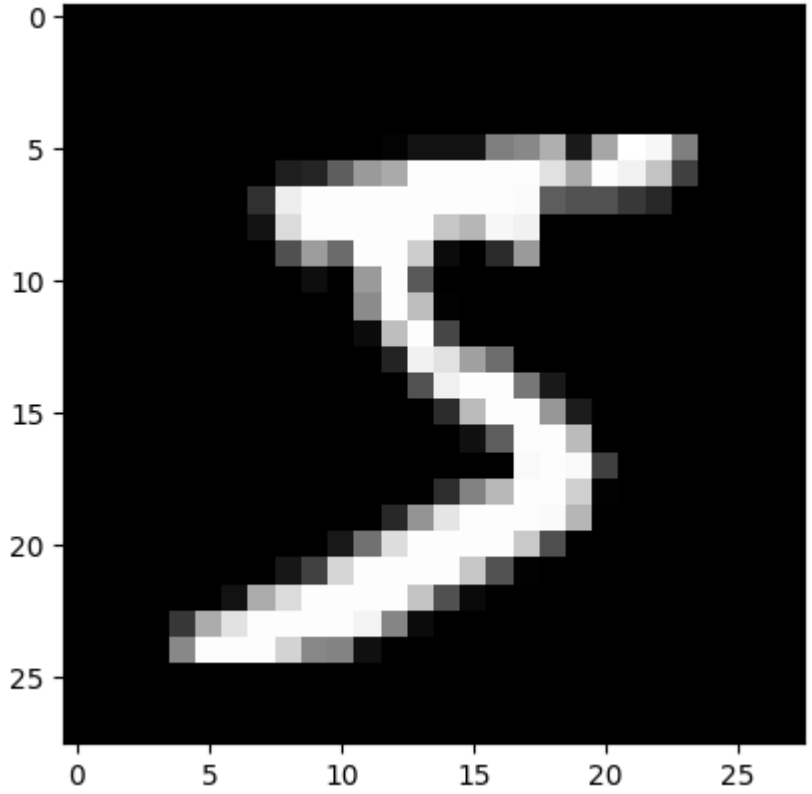
www.cas.dhbw.de

Agenda

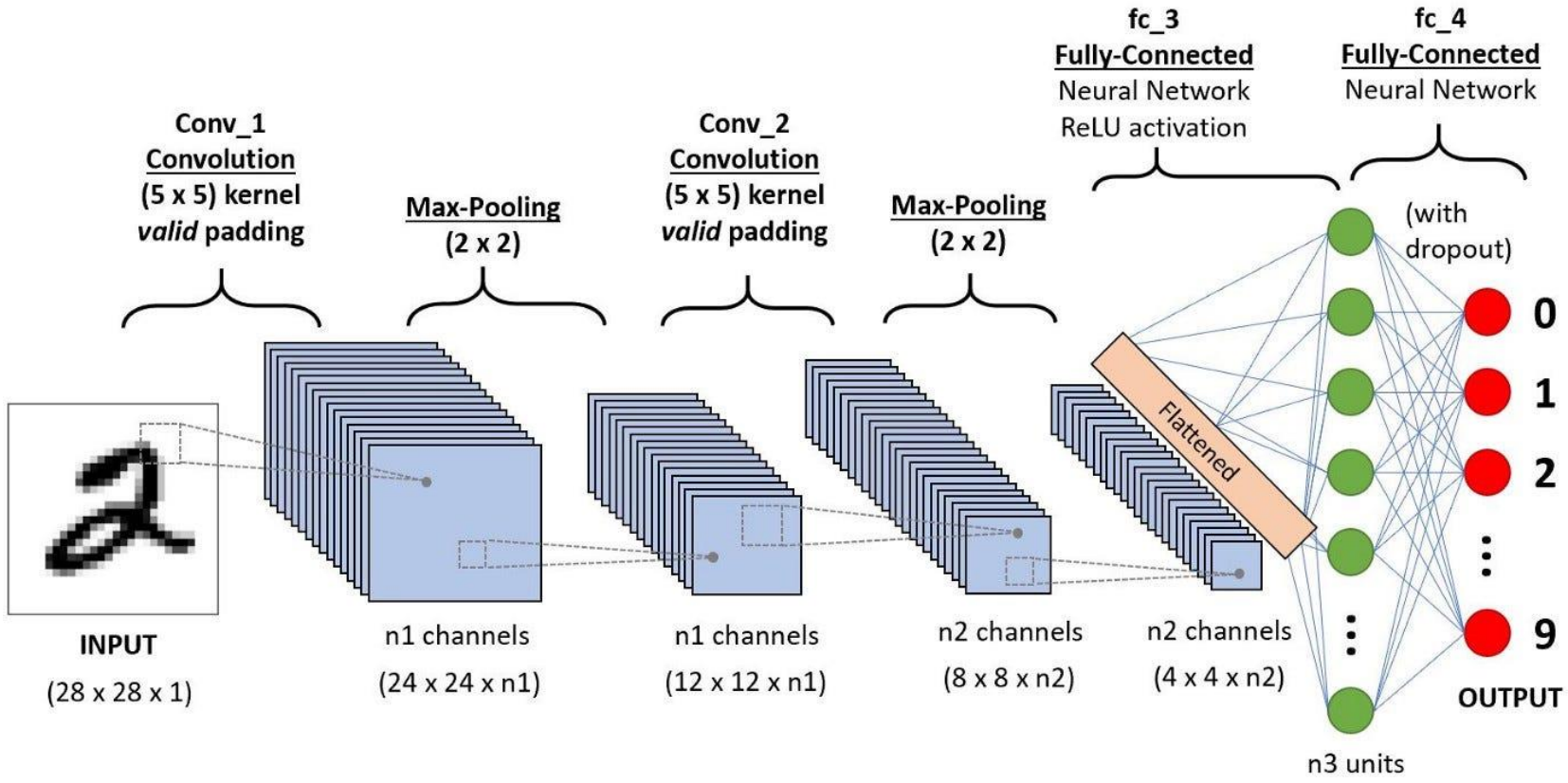
1. Einführung in Convolutional Neural Networks
2. Deep Dive in die verschiedenen Layers
3. Visualisierung eines CNN anhand von MNIST
4. Codebeispiel in Python mit Tensorflow anhand von MNIST
5. Q & A

1.1 Motivation für CNN

- NN zur Bildklassifizierung
  Bilder können als Pixelmatrizen dargestellt werden
- Neuronen sind u.a. Convolutions (Faltungen)
- CNN setzt sich aus verschiedenen Layern zusammen

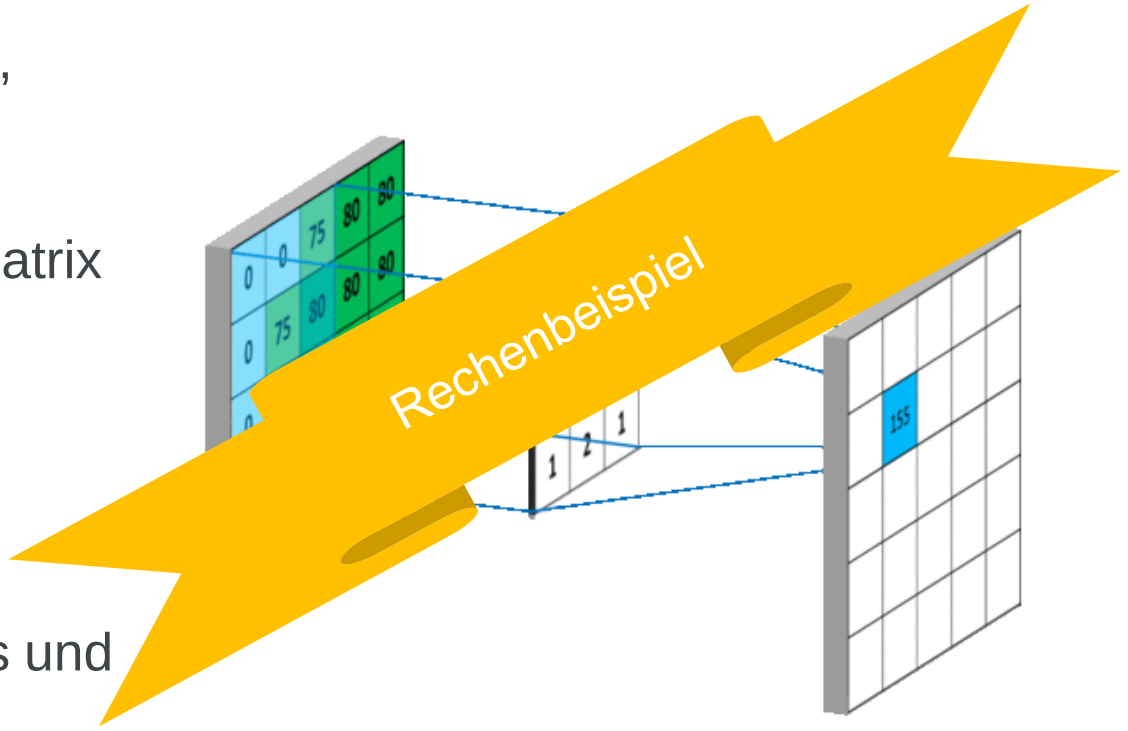


1.2 Aufbau eines Convolutional Neural Networks (CNN)

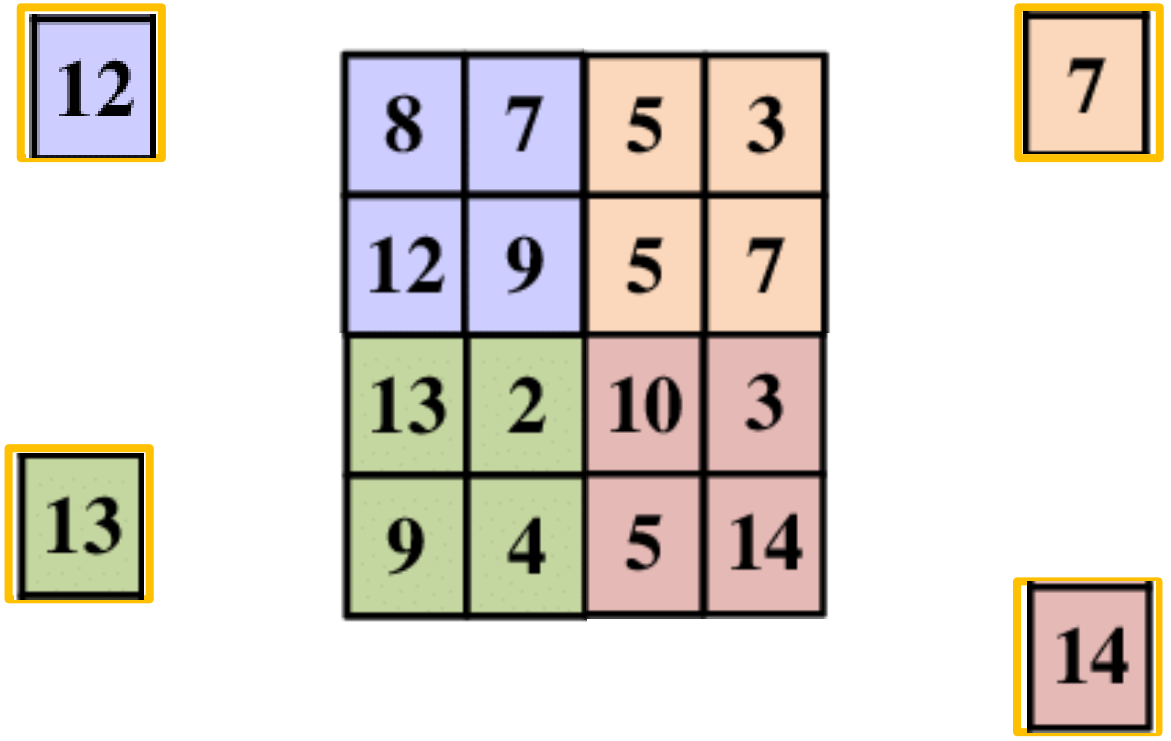


2.1 Convolutional Layer

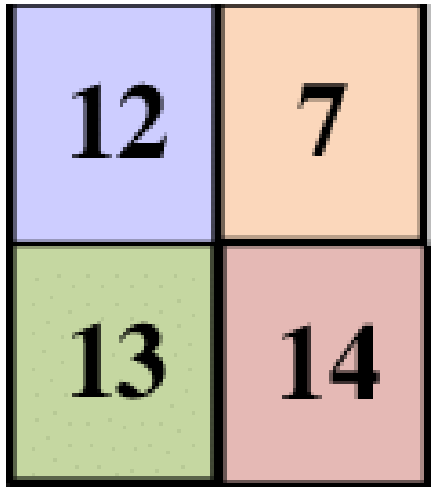
- Convolution ist eine Matrix der Größe n x n,
  n < Inputmatrix m
- Output des Convolutional Layers ist eine Matrix der Größe (m-2) x (m-2)
  → **feature map**
- Werte der feature map werden durch Skalarprodukt des betrachteten Ausschnitts und der Convolution berechnet



2.2 Pooling – Layer am Beispiel von Max Pooling



2.3 Flattening

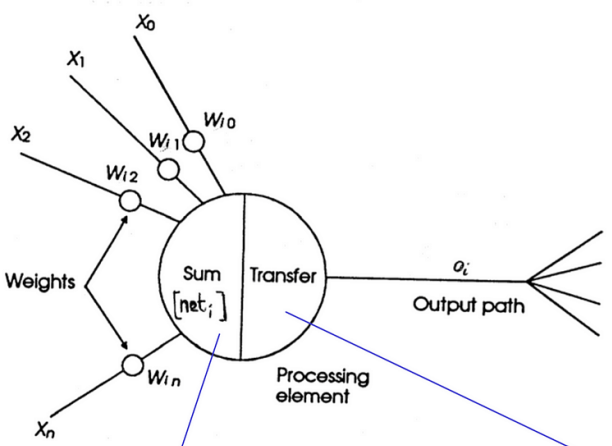


→ Matrix wird zu einem Vektor komprimiert

2.4 Dense Layer

- Fasst den Output der vorherigen Layer zusammen

  -> fully connected
- Implementiert ggf Bias
- Transformiert zum gewünschten Outputformat mithilfe einer Aktivierungsfunktion



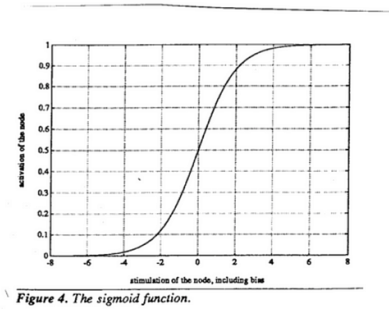
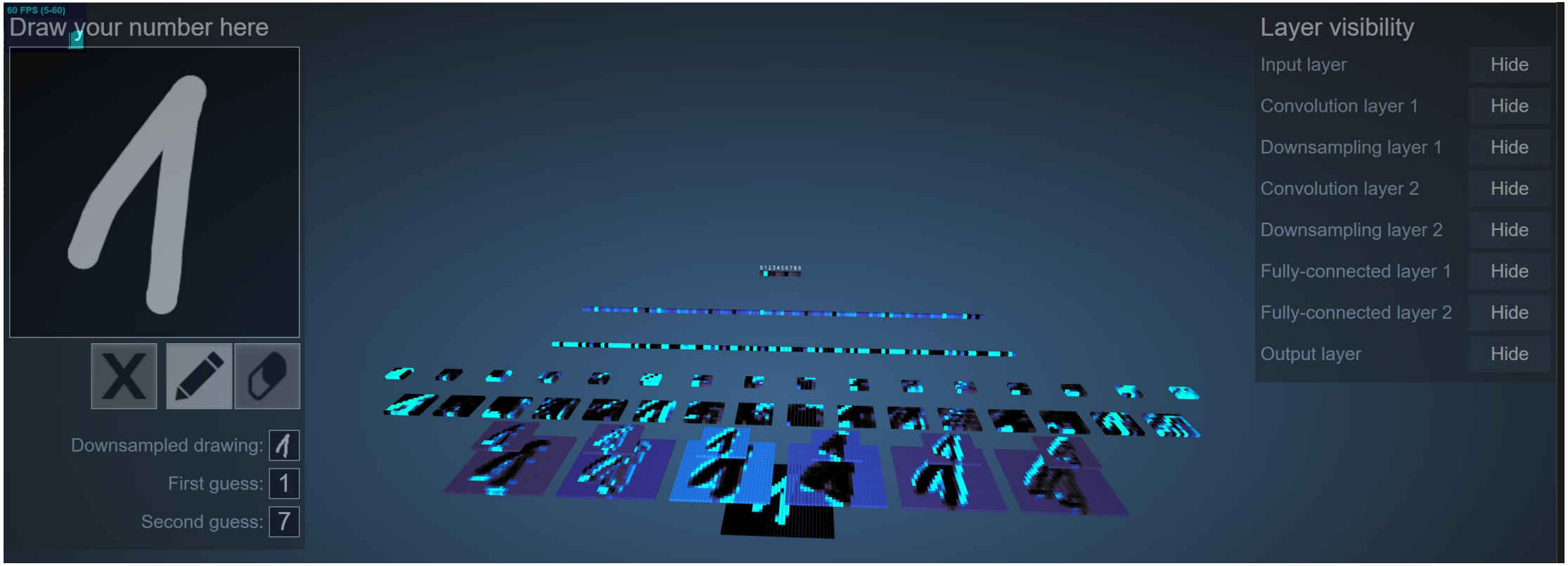


Figure 4. The sigmoid function.

$$net_i = \sum_j^N w_{ij} x_j - \vartheta_i$$

$$o_i = \frac{1}{1+e^{-net_i}}$$

3. Visualisierung eines CNN anhand von MNIST



URL: https://adamharley.com/nn_vis/cnn/3d.html

4. Codebeispiel in Python mit Tensorflow anhand von MNIST

```
from keras.models import Sequential
from keras.layers import Dense, Conv2D, Flatten, MaxPooling2D

model = Sequential()

model.add(Conv2D(128, kernel_size=(3, 3), activation="relu", input_shape=(28, 28, 1)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(128, kernel_size=(3, 3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(128, activation="relu"))
model.add(Dense(10, activation="softmax")) #Output = activation(dot(input, kernel) + bias)

model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"])

model.fit(X_train.reshape(60000, 28, 28, 1), y_train, epochs=10, batch_size=1000)
```

Quellen

Grafiken:
M. Shoaib Ali, 2022, „Flattening CNN layers for Neural Network and basic concepts”, Medium, https://medium.com/@muhammadshoaibali/flattening-cnn-layers-for-neural-network-694a232eda6a

Visualisierung:

A. W. Harley, 2015, "An Interactive Node-Link Visualization of Convolutional Neural Networks," in ISVC, pages 867-877,

Co-Pilot ☺