UNIVERSITÄT
ZU KÖLN

SPRACHVERARBEITUNG: ÜBUNG

SoSe 2024

Janis Pagel

Institut für Digital Humanities

2024-07-02

01

SOLUTION TO EXERCISE 08

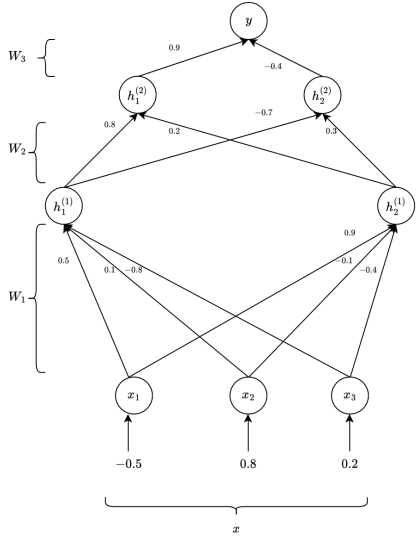
Solution to Exercise 08

- https://lehre.idh.uni-koeln.de/site/assets/files/5151/solution08.pdf

02

FEED-FORWARD NEURAL NETWORK

Network


y
W_3
0.9
−0.4
$h_1^{(2)}$
$h_2^{(2)}$
W_2
0.8
0.2
−0.7
0.3
$h_1^{(1)}$
$h_2^{(1)}$
W_1
0.5
0.1
−0.8
0.9
−0.1
−0.4
x_1
x_2
x_3
−0.5
0.8
0.2
x
Output
Hidden 2
Hidden 1
Input

Dot Product

- Product between two vectors
- For two vectors $\overrightarrow{a}$ and $\overrightarrow{b}$, the dot product is defined as

$$\overrightarrow{a} \cdot \overrightarrow{b} = \sum_{i=1}^{N} a_i b_i \tag{1}$$

- For example, given the input layer x of the network of the previous slide ([-0.5, 0.8, 0.2]) and the weights of the first neuron of the first hidden layer h_1^1 ([0.5, 0.1, -0.8]), the dot product of these two vectors is:

$$[-0.5, 0.8, 0.2] \cdot [0.5, 0.1, -0.8] = -0.5 \cdot 0.5 + 0.8 \cdot 0.1 + 0.2 \cdot -0.8 \approx -0.33 \tag{2}$$

Sigmoid function

- Popular activation function that maps values to a distribution between 0 and 1 (logistic function)
- Can be defined as

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

- For our value for hidden layer neuron h_1^1 (-0.33), the value after applying the sigmoid function would be:

$$sigmoid(-0.33) = \frac{1}{1 + e^{0.33}} \approx 0.42 \tag{4}$$

03

TENSORFLOW AND KERAS

Tensorflow and Keras

- Tensorflow: Implementation in Python which provides convient functions for building and running Neural Networks
- Keras: Tries to make applications of neural networks even easier, also works with other frameworks (e.g. PyTorch)

Tensorflow in Python

```
import pandas as pd
import tensorflow as tf
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense

nn = Sequential()
nn.add(Dense(100, input_dim=3, activation='sigmoid'))
nn.add(Dense(1, activation='sigmoid'))
print(nn.summary())

nn.compile(
  loss='binary_crossentropy',
  optimizer='sgd',
  metrics=['accuracy'],
)

nn.fit(X_train, y_train, epochs=200, initial_epoch=0, verbose=0)
loss, accuracy = nn.evaluate(X_test, y_test, verbose=1)
print(f'Accuracy {round(accuracy*100,2)}')
```

04

EXERCISE 09

Exercise 09

- https://lehre.idh.uni-koeln.de/site/assets/files/5151/exercise09.pdf


UNIVERSITY
OF COLOGNE
Janis Pagel
Institut für Digital Humanities
eMail janis.pagel@uni-koeln.de
Website https://janispagel.de
Foto: Gregor Hübl