Robotik Studium

FOKUS

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

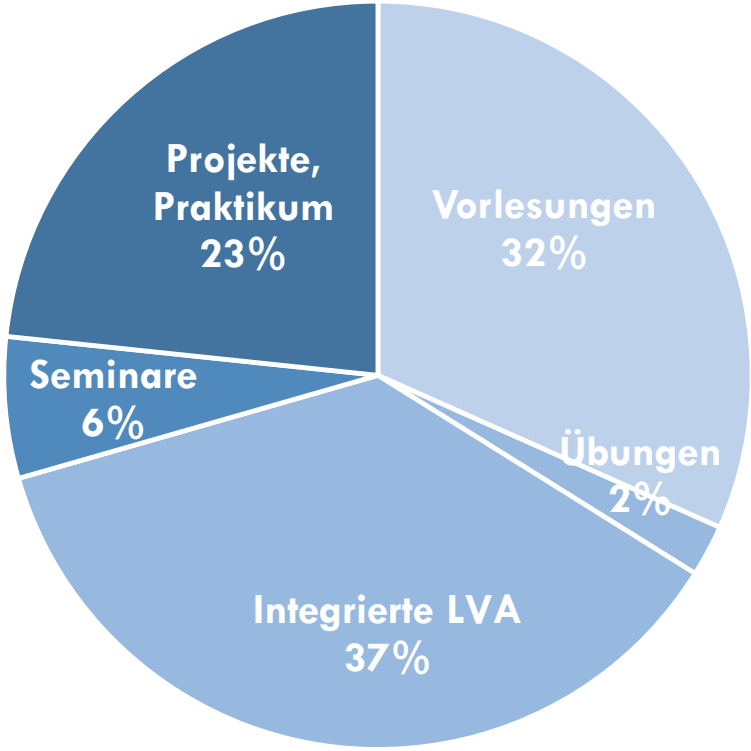
Theorie und Praxis

fhwn.ac.at/bro

Theorie

Praxis

- Vorlesungen
- Übungen
- Integrierte LVA
- Seminare
- Projekte, Praktikum



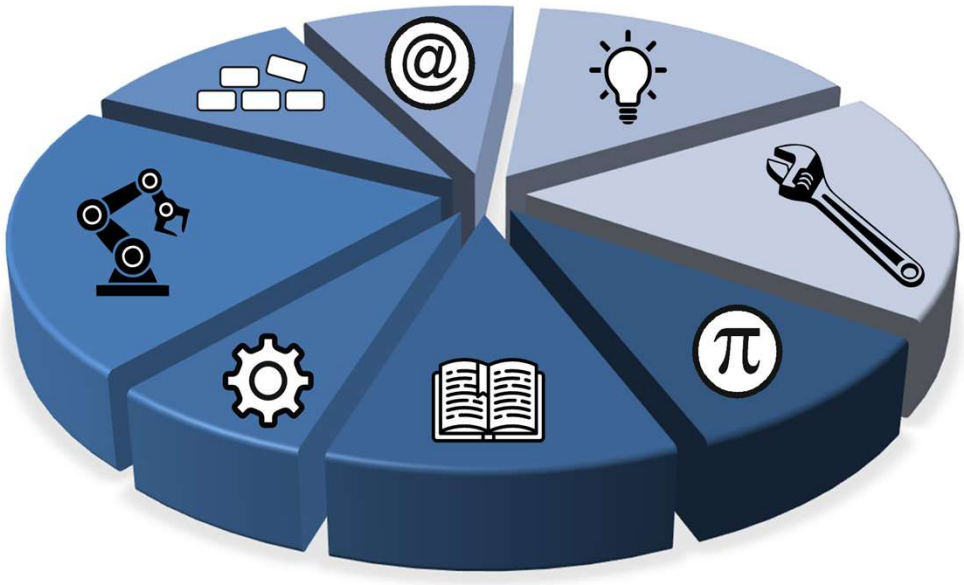
Fokus

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

Hoher Praxisbezug

fhwn.ac.at/bro

Mathematik	17 ECTS
Science & Communication	20 ECTS
Produktion	12 ECTS
Robotik & Automatisierung	37 ECTS
Technische Grundlagen	18 ECTS
Informationstechnologie	17 ECTS
Intelligente Syst. & Virtualisierung	27 ECTS
Spezialisierung & Praktikum	32 ECTS
Gesamt	**180 ECTS**



FOKUS IT, Int. Systeme & Virtualisierung

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

fhwn.ac.at/bro

1. Semester	ECTS	30
Mathematik 1		5
Scientific Computing		3
Grundlagen der Programmierung		4
Grundlagen der Informatik		3
Grundlagen der Robotik		2
Computer Aided Design		4
Communication Skills		4
Projekt Robotik 1		5

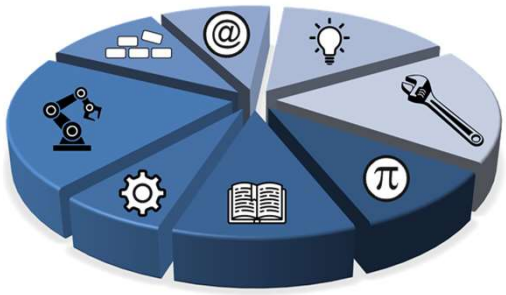
2. Semester	ECTS	30
Mathematik 2		5
Software Engineering		2
Objektorientierte Programmierung		4
Elektrotechnik		5
Mechanik		5
Betriebswirtschaftslehre		3
Scientific Skills Introduction		3
Projekt Robotik 2		3

3. Semester	ECTS	30
Datenanalyse und Statistik		4
Algorithmen und Datenstrukturen		4
Automatisierungstechnik		4
Sensoren und Aktoren		4
Simulation Robotik		4
Elektronik		4
Produktion und Logistik		3
Projekt Robotik 3		3

4. Semester	ECTS	30
Industrielle Robotik		3
Mobile Robotik		3
Computer Vision		4
Künstliche Intelligenz		4
Embedded Systems		4
Regelungstechnik		4
Betriebssicherheit		2
Spezialisierung		6

5. Semester	ECTS	30
Medinzinische Robotik		3
Smart Robotics		3
Computergrafik		4
Datenbanken und Maschin. Lernen		4
Netzwerke und Bussysteme		3
Flexible Produktion		4
Scientific Skills Application		3
Spezialisierung		6

6. Semester	ECTS	30
Berufspraktikum		20
Begleitseminar		2
Bachelorarbeit		8



STUDIUM

WIRTSCHAFT | TECHNIK | SPORT | SICHERHEIT | GESUNDHEIT

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

Key Facts

Akademischer Grad
Bachelor of Science in Engineering (BSc.)

Studienort
Campus 1 Wiener Neustadt | FabLab Mödling

Umfang
6 Semester (180 ECTS)

Sprache
Deutsch

Bewerbung
Nachweis der Zugangsberechtigung (z. B. Maturazeugnis) kann nachgereicht werden

Studienbeginn
September

fhwn.ac.at/bro

Organisationsform
Vollzeit

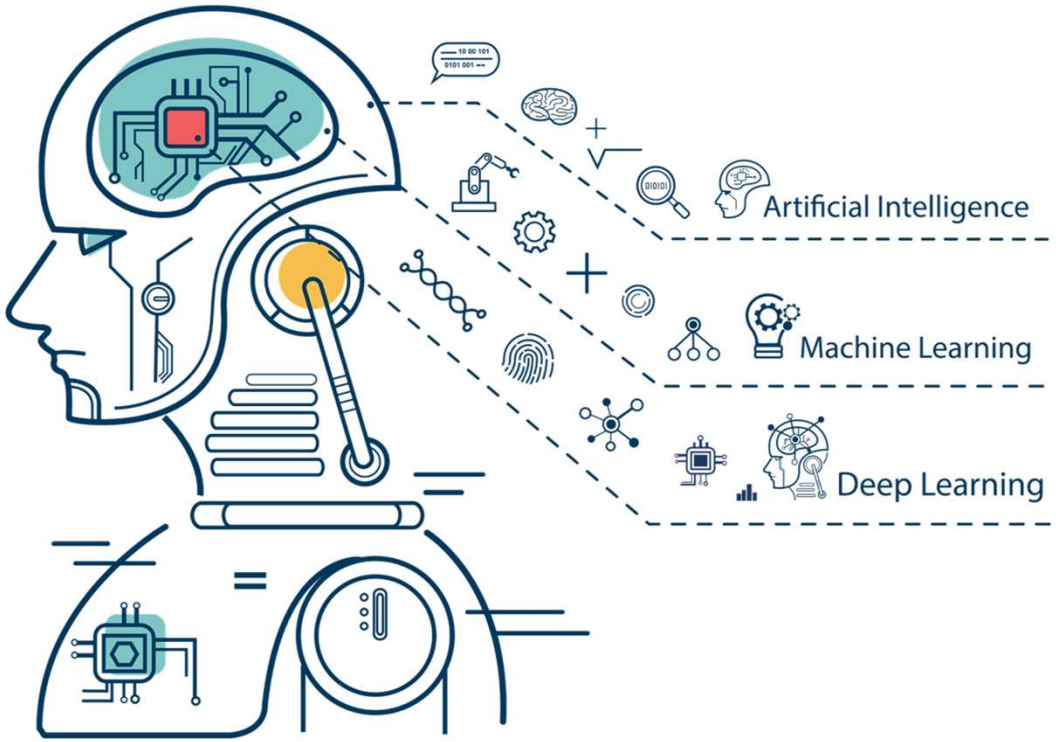
Aufnahmetermine
Laufend

Kosten
€ 363,36 + € 22,70 ÖH-Beitrag pro Semester

Studienplätze
30

Bewerbungsfrist
bis Ende Juni

Pflichtpraktikum
Ja



FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

KI-Workshop: Ziffernerkennung

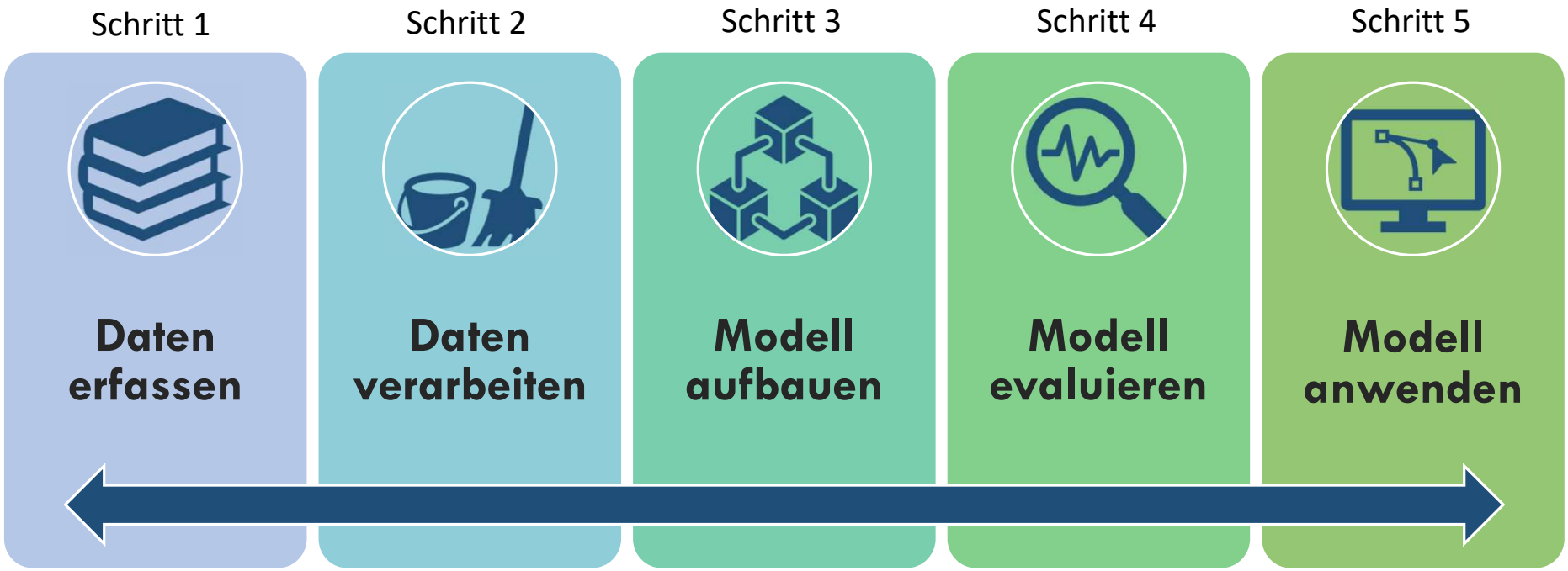
Mustafa Algan, MSc
Studiengang Robotik

fhwn.ac.at/bro
robotikfhwn

Aufgabenstellung

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

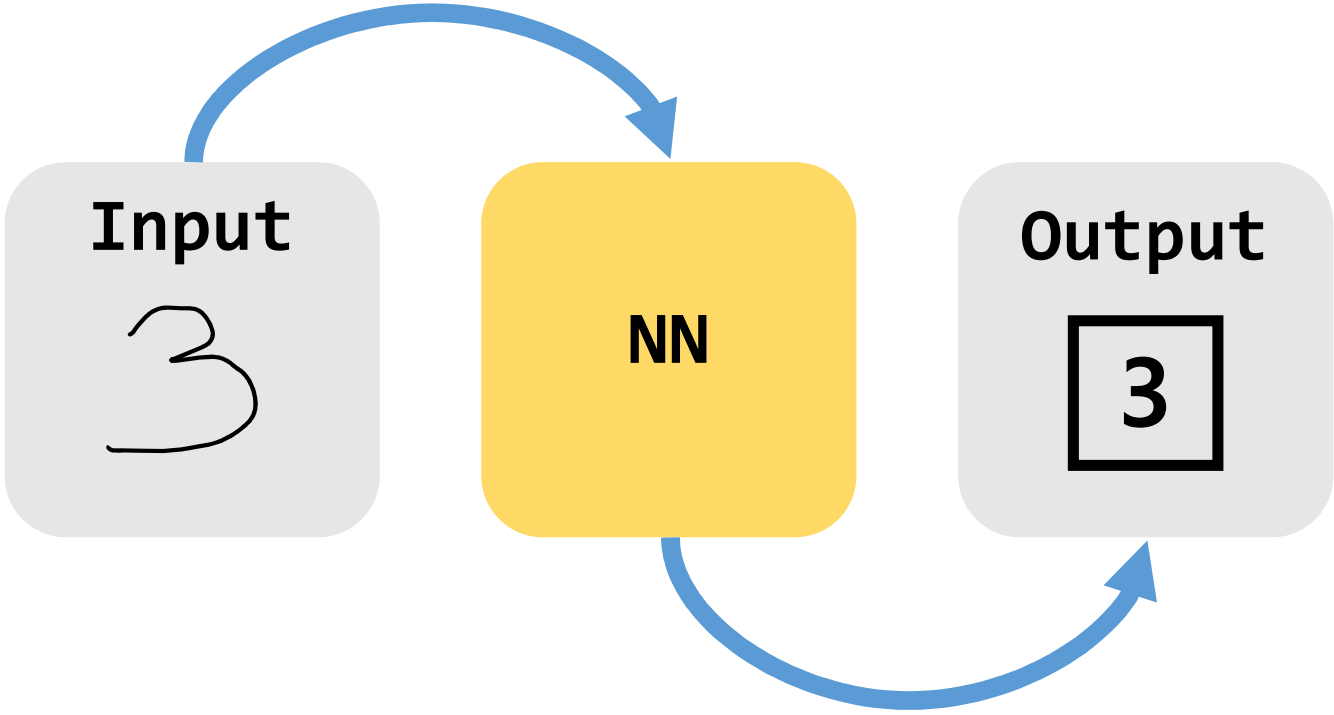
Was muss für die Ziffernerkennung gemacht werden?



1
2
3
4
5

Aufgabenstellung

Was muss für die Ziffernerkennung gemacht werden?

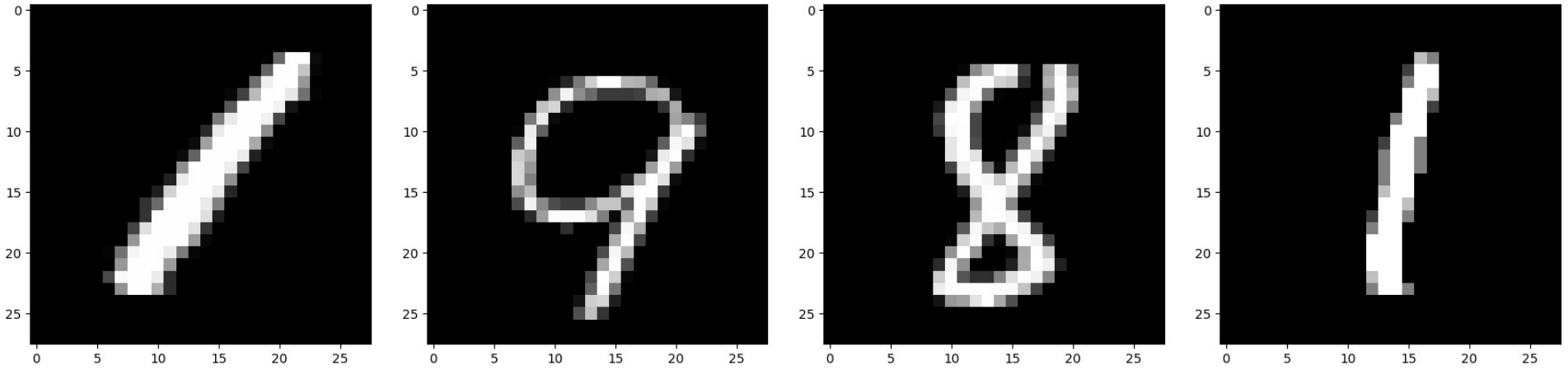


Schritt 1: Daten erfassen

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Welche Daten werden verwendet?

- Handgeschriebene Ziffern aus der MNIST Datenbank
- Bildauflösung 28x28 Pixel
- Welche Zahl entspricht schwarz und weiß?



1
2
3
4
5

MNIST Datensatz: https://www.kaggle.com/datasets/hojjatk/mnist-dataset

Schritt 1: Daten erfassen

Programmierumgebung

Schritt 1: Daten erfassen

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Bibliotheken importieren

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

numpy:
Ermöglicht einfache Handhabung von Vektoren, Matrizen oder generell großen mehrdimensionalen Arrays.
Enthält auch Funktionen für numerische Berechnungen.

pandas:
Verarbeitung, Analyse und Darstellung von Tabellen.

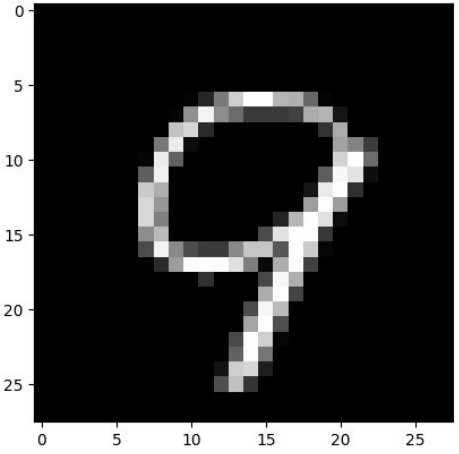
matplotlib:
Erlaubt mathematische Darstellungen aller Art anzufertigen.

Schritt 1: Daten erfassen

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Wie lese ich die Werte aus einer Datei?

```
# Daten einlesen
data = pd.read_csv('train.csv')

# Dimension der Daten zeigen
print(f"Dimension des Datensatzes: {data.shape}")

# Ein paar Zeilen aus den Daten zeigen
data.head()

data = np.array(data)
m, n = data.shape
```



1
2
3
4
5

Schritt 2: Daten verarbeiten

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Vermischen der Daten! Warum?

```
np.random.shuffle(data)
```

Was passiert beim Shuffel?

```
arr = np.arange(9).reshape((3, 3))
np.random.shuffle(arr)
arr
```

Vor Shuffle:

```
array([
[0, 1, 2],
[3, 4, 5],
[6, 7, 8]])
```

Nach Shuffle:

```
array([
[3, 4, 5],
[6, 7, 8],
[0, 1, 2]])
```

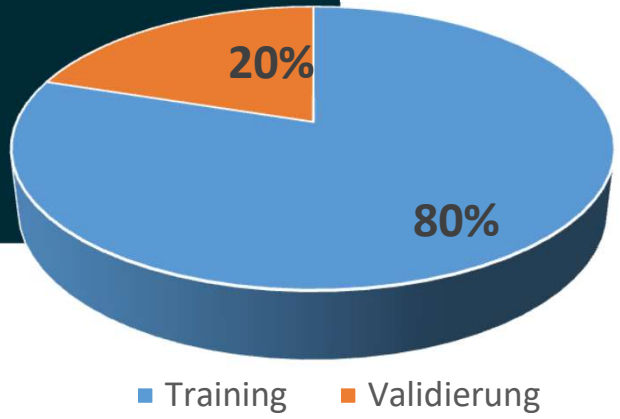
1
2
3
4
5

Schritt 2: Daten verarbeiten

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Trainings- & Validierungsdatensätze

```
validation_samples_number = int(0.2 * samples_number)

data_dev = data[0:validation_samples_number].T
Y_dev = data_dev[0]
X_dev = data_dev[1:pixels]
X_dev = X_dev / 255.
```



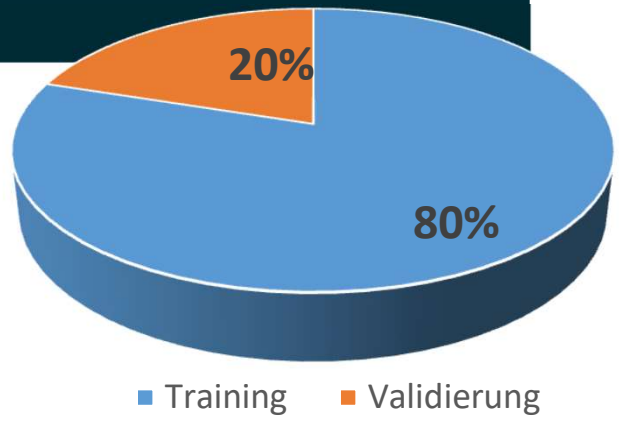
1
2
3
4
5

Schritt 2: Daten verarbeiten

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Trainings- & Validierungsdatensätze

```
data_train = data[validation_samples_number:samples_number].T
Y_train = data_train[0]
X_train = data_train[1:pixels]
X_train = X_train / 255.
```



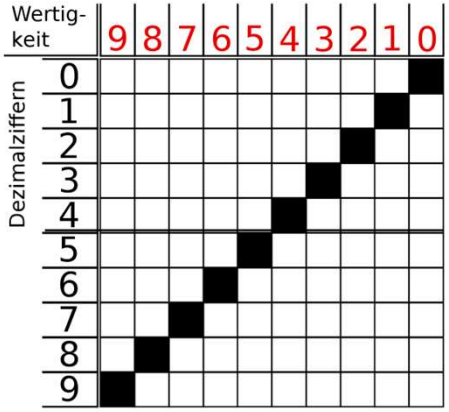
1
2
3
4
5

Schritt 2: Daten verarbeiten

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

one hot encode

```
def one_hot(Y):
    one_hot_Y = np.zeros((Y.size, Y.max() + 1))
    one_hot_Y[np.arange(Y.size), Y] = 1
    one_hot_Y = one_hot_Y.T
    return one_hot_Y
```



Zeile	Ziffer
0	1
1	8
2	1
3	9
4	7
5	8
6	6

Encoded

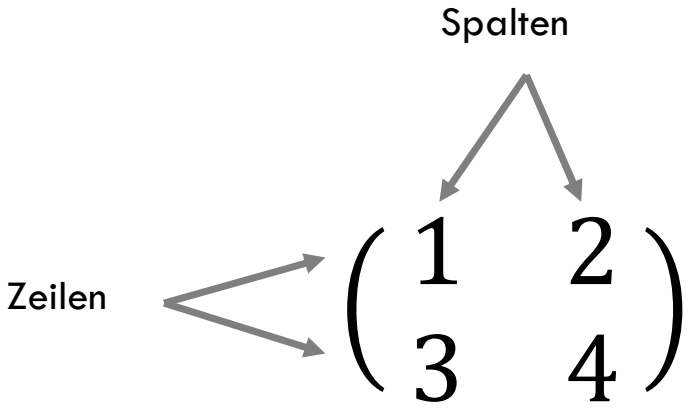
Zeile	0	1	2	3	4	5	6	7	8	9
0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	1	0
2	0	1	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	1
4	0	0	0	0	0	0	0	7	0	0
5	0	0	0	0	0	0	0	0	1	0
6	0	0	0	0	0	0	1	0	0	0

1
2
3
4
5

Schritt 3: Modell aufbauen

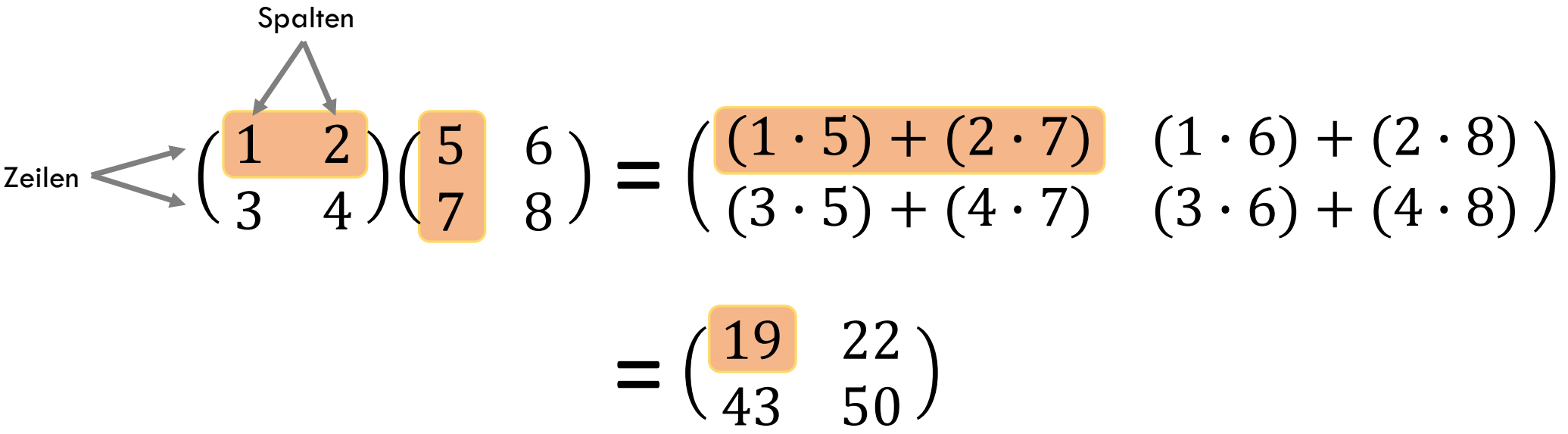
Exkursion → Matrizen 🤒🤧🤧🙄

	A	B	C	D
1	1	2	3	
2	4	5	6	
3				



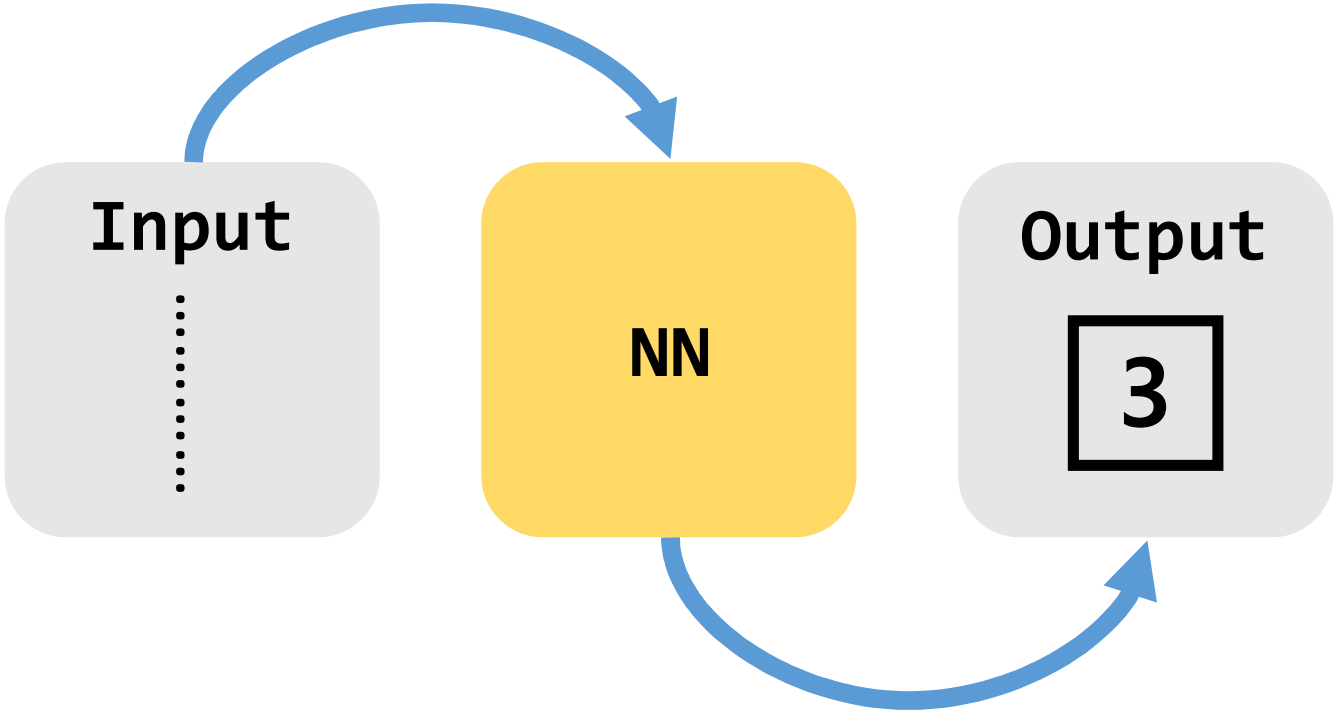
Schritt 3: Modell aufbauen

Exkursion → Matrizen 🤔🤧🤧🙄



Schritt 3: Modell aufbauen

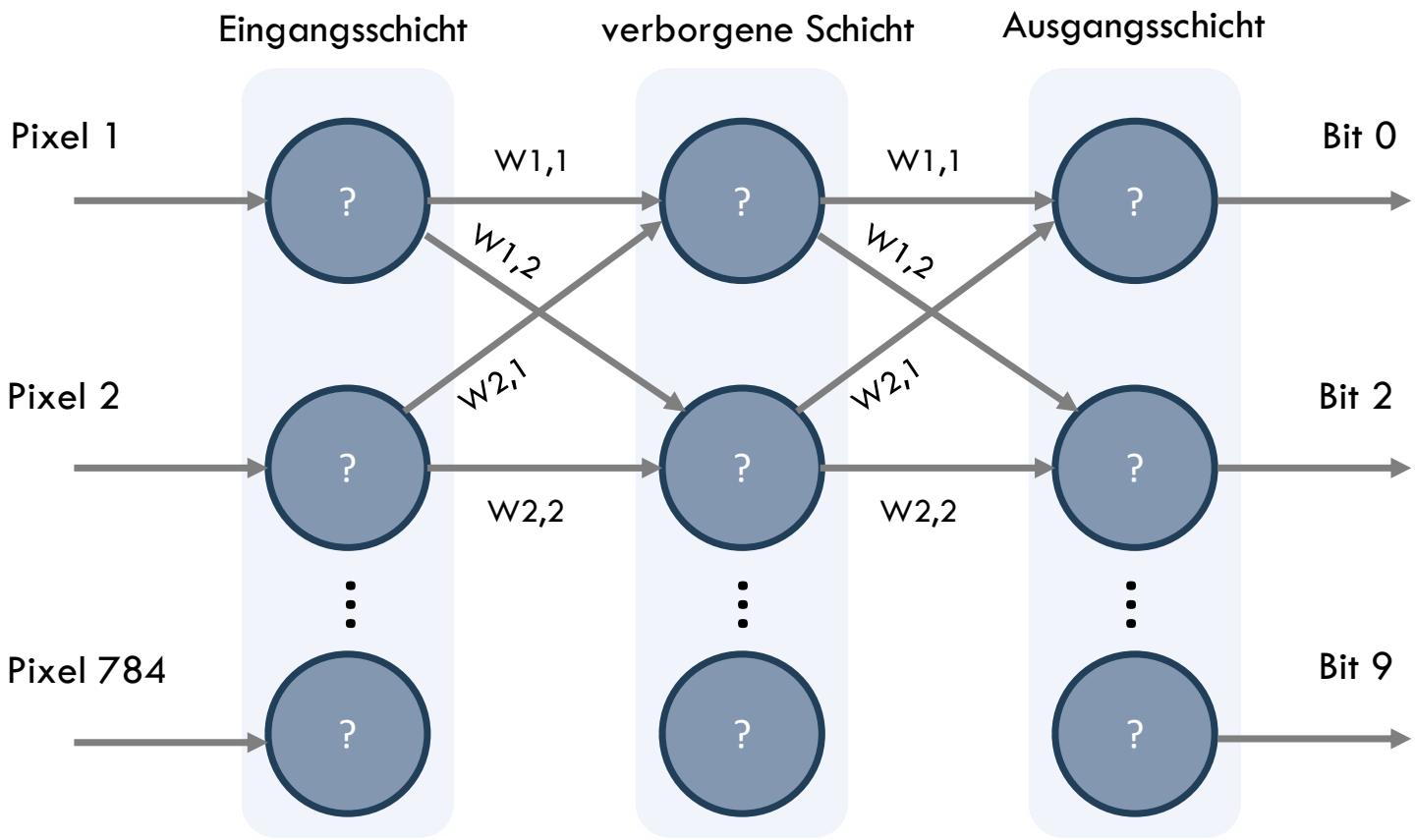
Neuronales Netz!



Schritt 3: Modell aufbauen

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

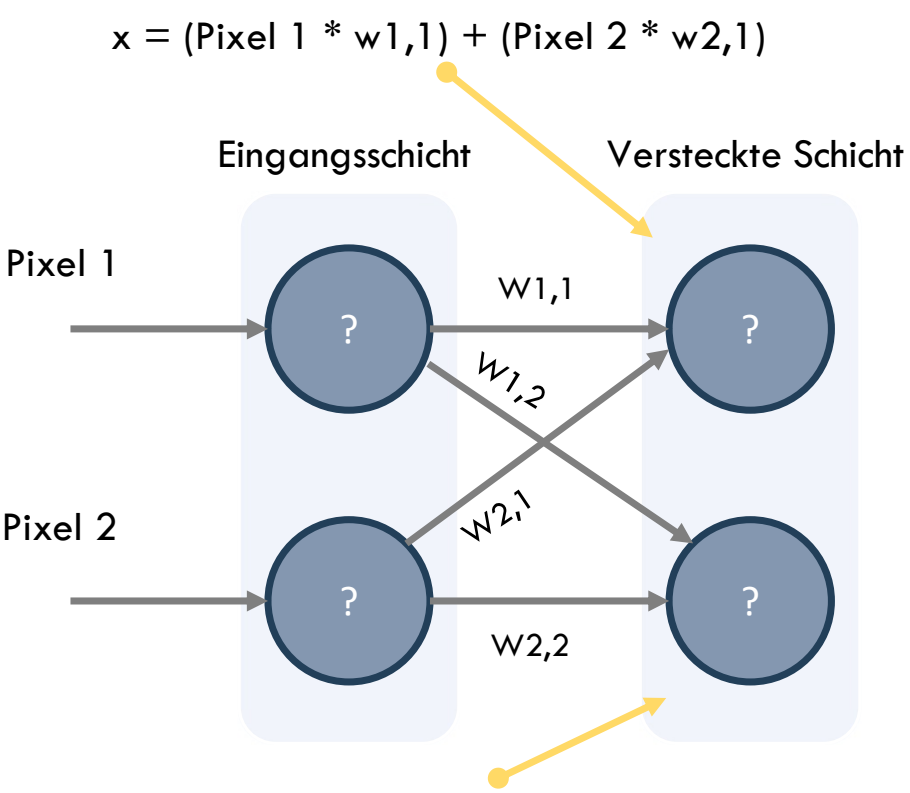
Neuronales Netz!



1
2
3
4
5

Schritt 3: Modell aufbauen

Neuronales Netz!



$$\begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix} \begin{pmatrix} input1 \\ input2 \end{pmatrix} =$$

$$= \begin{pmatrix} (input1 \cdot w_{1,1}) + (input2 \cdot w_{2,1}) \\ (input1 \cdot w_{1,2}) + (input2 \cdot w_{2,2}) \end{pmatrix}$$

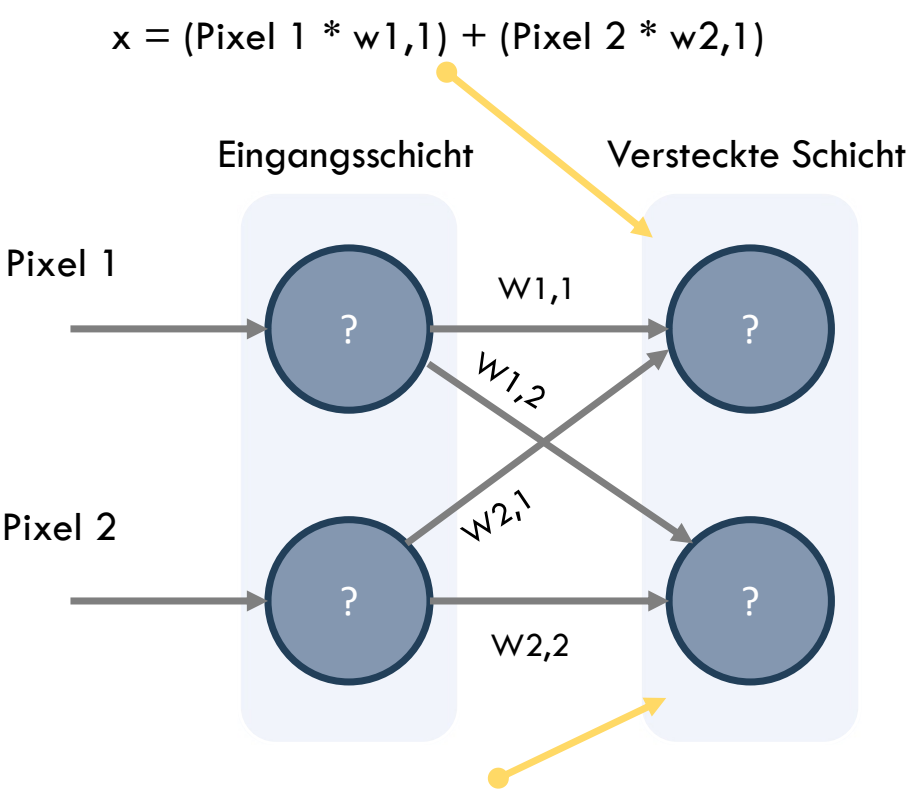
X = W • I

Gewichtsmatrix W & Eingabematrix I

1
2
3
4
5

Schritt 3: Modell aufbauen

Neuronales Netz!



$$\begin{pmatrix} w_{1,1} & w_{2,1} \\ w_{1,2} & w_{2,2} \end{pmatrix} \begin{pmatrix} input1 \\ input2 \end{pmatrix} =$$

$$= \begin{pmatrix} (input1 \cdot w_{1,1}) + (input2 \cdot w_{2,1}) \\ (input1 \cdot w_{1,2}) + (input2 \cdot w_{2,2}) \end{pmatrix}$$

X = W • I

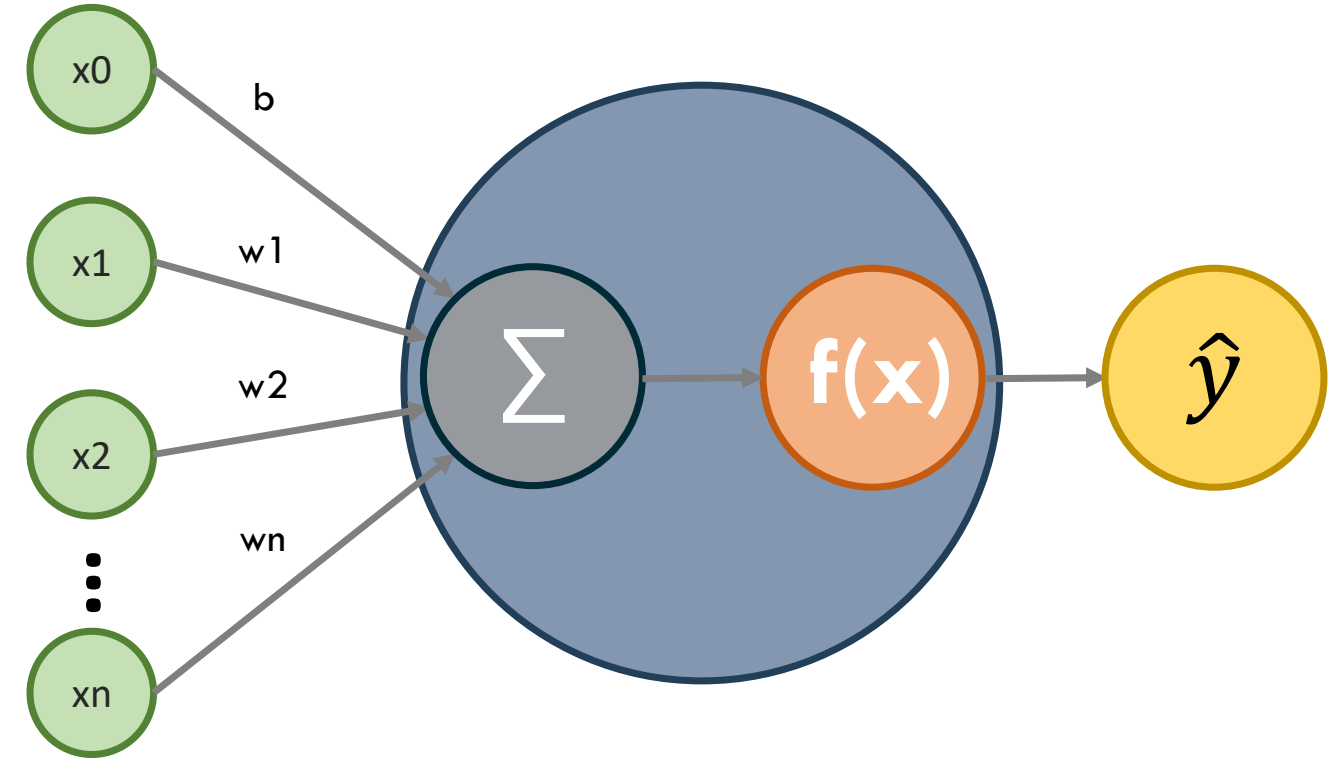
Gewichtsmatrix W & Eingabematrix I

1
2
3
4
5

Schritt 3: Modell aufbauen

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz!



Eingänge Gewichte Summe Aktivierungsfunktion Ausgang

Ausgang

$$\hat{y} = f\left(\sum_{k=1}^{n} x_k \cdot w_k + b\right)$$

Nicht-lineare Funktion

$$\hat{y} = f(X^T W + b)$$

1
2
3
4
5

Schritt 3: Modell aufbauen

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Zufällige Gewichte

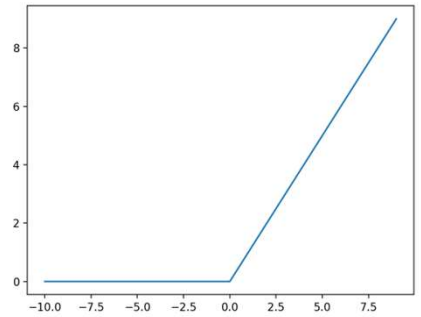
```
def init_params():
    W1 = np.random.rand(10, 784) - 0.5
    b1 = np.random.rand(10, 1) - 0.5
    W2 = np.random.rand(10, 10) - 0.5
    b2 = np.random.rand(10, 1) - 0.5
    return W1, b1, W2, b2
```

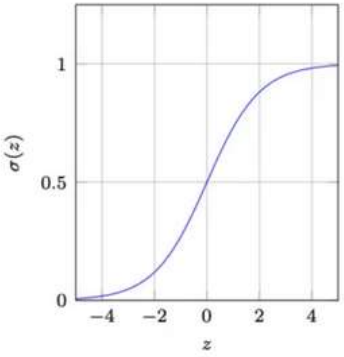
Schritt 3: Modell aufbauen

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

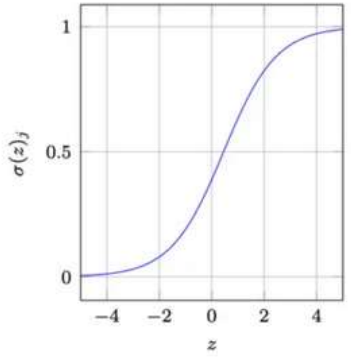
Neuronales Netz – Aktivierungsfunktionen

```python
# Write the activation functions
def ReLU(z):
    return np.maximum(z, 0)

def ReLU_derivative(z):
    return np.where(z > 0, 1, 0)

def Softmax(z):
    exp_z = np.exp(z - np.max(z)) # Avoid overflow
    return exp_z / np.sum(exp_z, axis=0)

def Sigmoid (x):
    return 1/(1 + np.exp(-x))

def Sigmoid_derivative(x):
    return x * (1 - x)
```





(a) Sigmoid activation function.



(b) Softmax activation function.

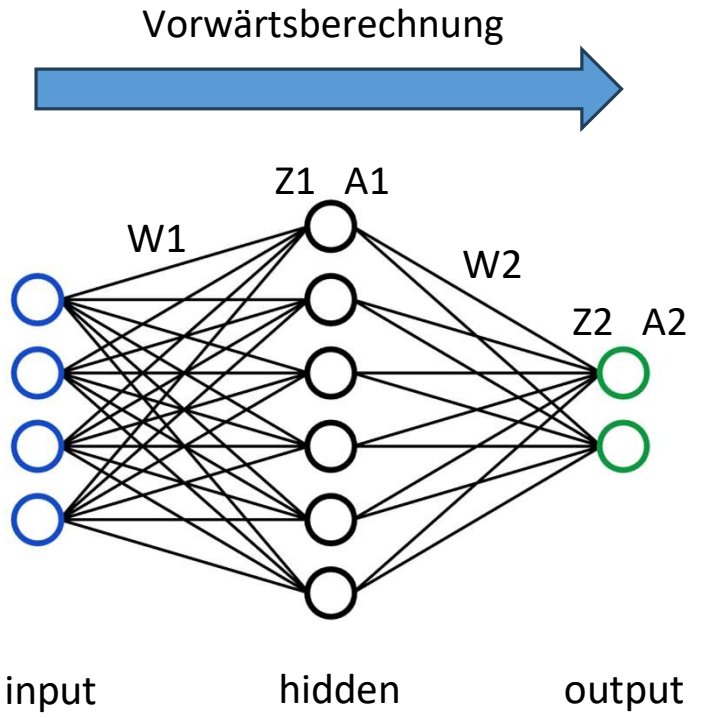
1
2
3
4
5

Schritt 3: Modell aufbauen

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz - Vorwärtsberechnung

```
def forward_prop(W1, b1, W2, b2, X):
    Z1 = W1.dot(X) + b1
    A1 = ReLU(Z1)
    Z2 = W2.dot(A1) + b2
    A2 = Softmax(Z2)
    return Z1, A1, Z2, A2
```



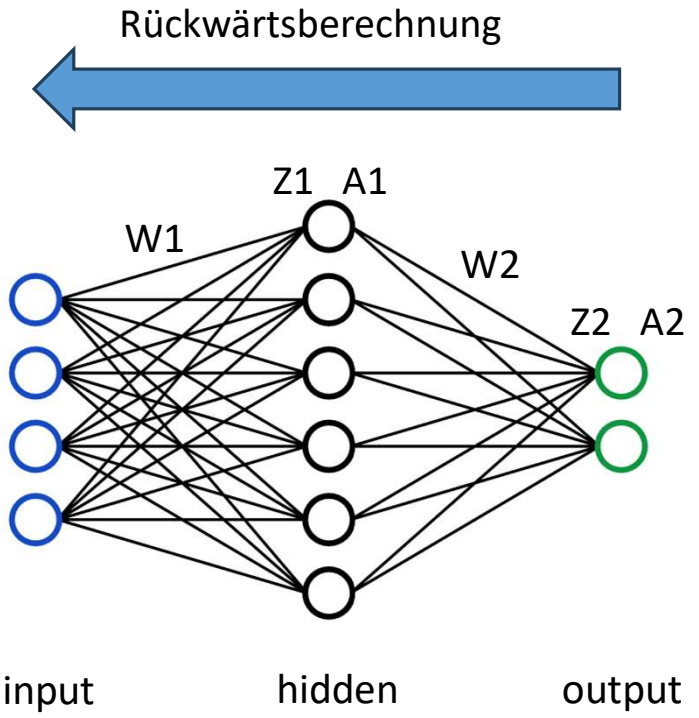
1 2 3 4 5

Schritt 3: Modell aufbauen

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz - Rückwärtsberechnung

```python
def backward_prop(Z1, A1, Z2, A2, W1, W2, X, Y):
    one_hot_Y = one_hot(Y)

    dZ2 = A2 - one_hot_Y
    dW2 = 1 / m * dZ2.dot(A1.T)
    db2 = 1 / m * np.sum(dZ2)
    dZ1 = W2.T.dot(dZ2) * ReLU_derivative(Z1)
    dW1 = 1 / m * dZ1.dot(X.T)
    db1 = 1 / m * np.sum(dZ1)
    return dW1, db1, dW2, db2
```



1
2
3
4
5

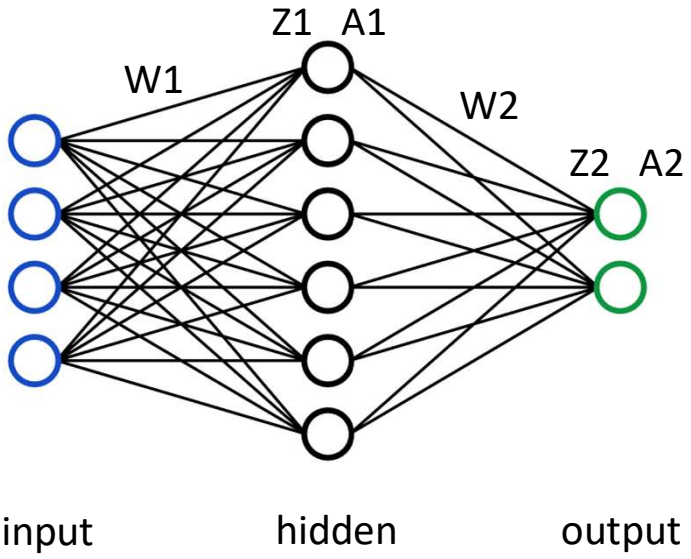
Schritt 3: Modell aufbauen

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Gewichte anpassen

```
def update_params(W1, b1, W2, b2, dW1, db1, dW2, db2, alpha):
    W1 = W1 - alpha * dW1
    b1 = b1 - alpha * db1
    W2 = W2 - alpha * dW2
    b2 = b2 - alpha * db2
    return W1, b1, W2, b2
```

UPDATE GEWICHTE!



1
2
3
4
5

Schritt 3: Modell aufbauen

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Auswertungsfunktionen

```python
def get_predictions(A2):
    return np.argmax(A2, 0)

def get_accuracy(predictions, Y):
    print(predictions, Y)
    return np.sum(predictions == Y) / Y.size
```

Zeile	0	1	2	3	4	5	6	7	8	9
0	0	**1**	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	**1**	0
2	0	**1**	0	0	0	0	0	0	0	0

$$\text{Genauigkeit [\%]} = \frac{Richtige\ Schätzung}{Totale\ Schätzungen}$$

1
2
3
4
5

Schritt 3: Modell aufbauen

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Auswertungsfunktionen

```python
def get_predictions(A2):
    return np.argmax(A2, 0)

def get_accuracy(predictions, Y):
    print(predictions, Y)
    return np.sum(predictions == Y) / Y.size
```

Zeile	0	1	2	3	4	5	6	7	8	9
0	0	**1**	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	**1**	0
2	0	**1**	0	0	0	0	0	0	0	0

$$\text{Genauigkeit [\%]} = \frac{Richtige\ Schätzung}{Totale\ Schätzungen}$$

1 2 3 4 5

Schritt 3: Modell aufbauen

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Auswertungsfunktionen

```python
def compute_cost(A2, Y):
    one_hot_Y = one_hot(Y)
    logprobs = -np.log(A2) * one_hot_Y
    cost = np.mean(logprobs)
    return cost
```

Schritt 3: Modell aufbauen

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Auswertungsfunktionen

```
def plot_cost_graph(cost_history):
    iterations = len(cost_history)
    plt.figure(figsize=(22, 8))
    plt.plot(range(iterations), cost_history)
    plt.title('Cost vs. Iterations')
    plt.xlabel('Iterations')
    plt.ylabel('Cost')
    plt.grid(True)
    plt.show();
```

Schritt 3: Modell aufbauen

```python
def gradient_descent(X, Y, alpha, iterations):
    W1, b1, W2, b2 = init_params()
    cost_history = []  # List to store cost values at each iteration

    for i in range(iterations):
        Z1, A1, Z2, A2 = forward_prop(W1, b1, W2, b2, X)
        dW1, db1, dW2, db2 = backward_prop(Z1, A1, Z2, A2, W1, W2, X, Y)
        W1, b1, W2, b2 = update_params(W1, b1, W2, b2, dW1, db1, dW2, db2, alpha)
        if i % 10 == 0:
            print("Iteration: ", i)
            predictions = get_predictions(A2)
            print(get_accuracy(predictions, Y))

        cost = compute_cost(A2,Y)
        cost_history.append(cost)

    return W1, b1, W2, b2, cost_history
```

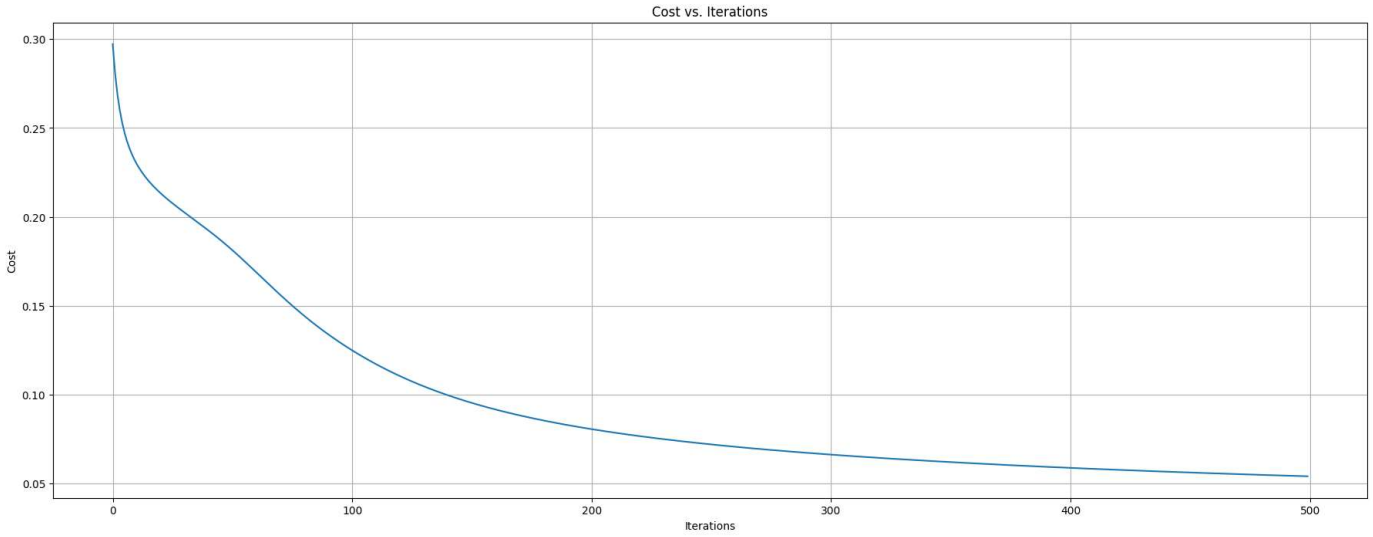
Schritt 3: Modell aufbauen

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Trainingszeit!!! 🤩

```
W1, b1, W2, b2, cost_history= gradient_descent(X_train, Y_train, 0.10, 500)

plot_cost_graph(cost_history)
```

```
Iteration:  0
0.08035714285714286
Iteration:  10
0.1480357142857143
Iteration:  20
0.23791666666666667
Iteration:  30
0.28401785714285716
Iteration:  40
0.3273214285714286
Iteration:  50
0.3794345238095238
```



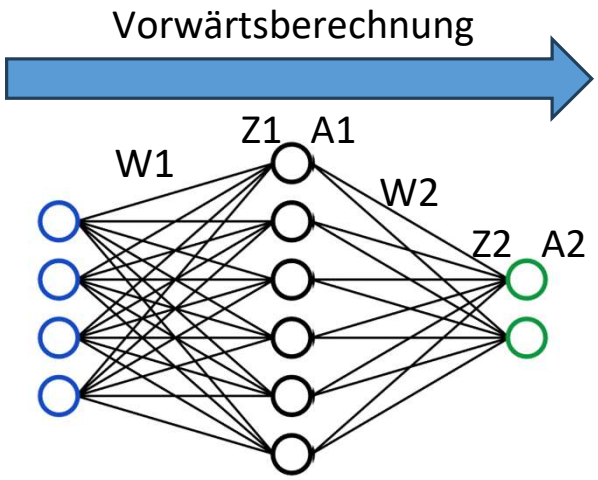
1
2
3
4
5

Schritt 4: Modell evaluieren

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Validierung 📊

```
def make_predictions(X, W1, b1, W2, b2):
    _, _, _, A2 = forward_prop(W1, b1, W2, b2, X)
    predictions = get_predictions(A2)
    return predictions
```



1
2
3
4
5

Schritt 4: Modell evaluieren

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Validierung 📊

Die Genauigkeit mit den Testdaten überprüfen:

```
dev_predictions = make_predictions(X_dev, W1, b1, W2, b2)
get_accuracy(dev_predictions, Y_dev)
```

Die Genauigkeit liegt bei rund 80%.

```
0.8084523809523809
```

1
2
3
4
5

Schritt 4: Modell evaluieren

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Validierung 📊

```python
def test_prediction(index, W1, b1, W2, b2):
    current_image = X_train[:, index, None]
    prediction = make_predictions(X_train[:, index, None], W1, b1, W2, b2)
    label = Y_train[index]
    print("Prediction: ", prediction)
    print("Label: ", label)

    current_image = current_image.reshape((28, 28)) * 255
    plt.gray()
    plt.imshow(current_image, interpolation='nearest')
    plt.show()
```

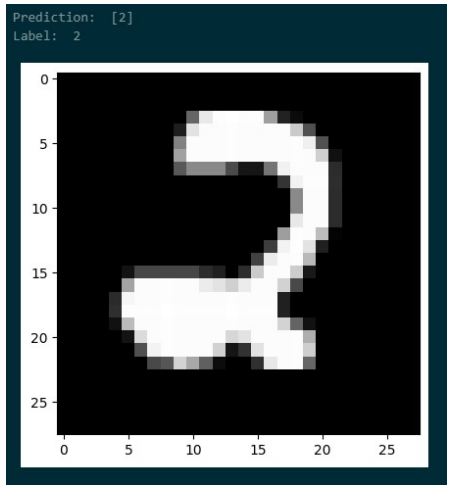
1
2
3
4
5

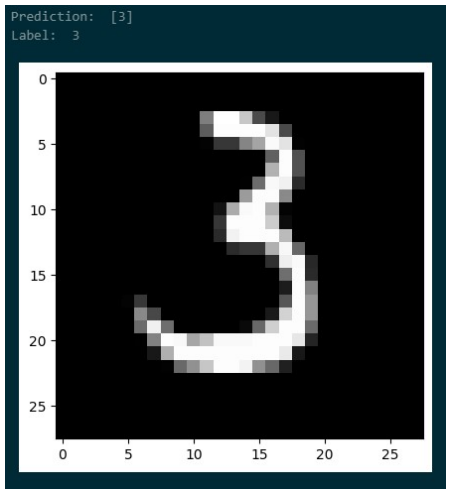
Schritt 4: Modell evaluieren

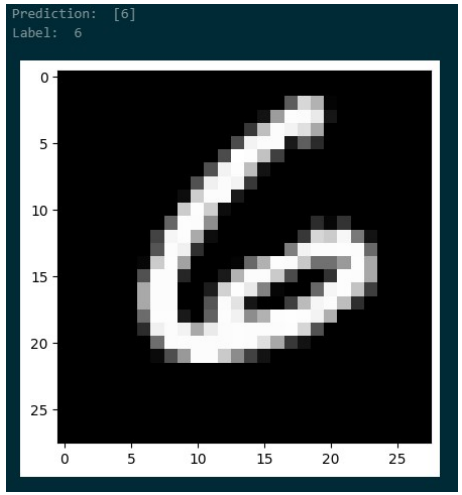
FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

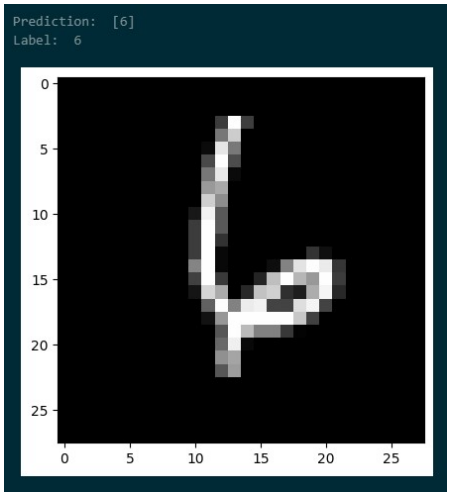
Neuronales Netz – Validierung 🤗

```
test_prediction(0, W1, b1, W2, b2)
test_prediction(1, W1, b1, W2, b2)
test_prediction(2, W1, b1, W2, b2)
test_prediction(3, W1, b1, W2, b2)
```







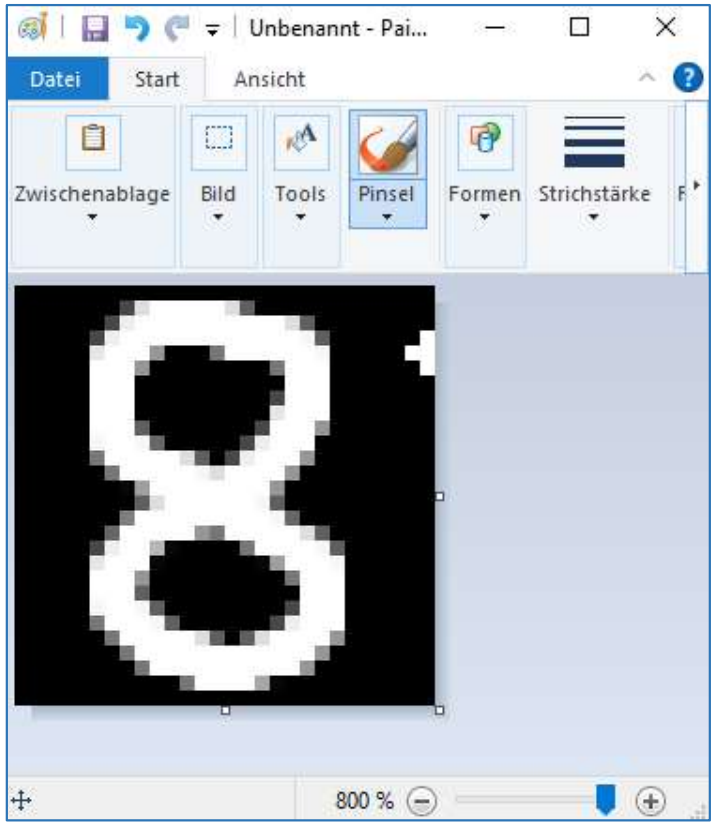


1
2
3
4
5

Schritt 5: Modell anwenden

Neuronales Netz – Meine Ziffer

- Paint öffnen
- Bildgröße auf 28x28 ändern
- Hintergrund auf schwarz ändern
- Eigene Ziffer mit weiß zeichnen
- Abspeichern



Schritt 5: Modell anwenden

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Meine Ziffer 🎨

```
import matplotlib.image as mpimg
img = mpimg.imread('drei.png')
imgplot = plt.imshow(img)
plt.show()
x_my = img
x_my = np.squeeze(x_my[:,:,0])
x_my = np.squeeze(x_my)
x_my = x_my.reshape(-1,1)

y_my = make_predictions(x_my,W1,b1,W2,b2)
print(y_my)
```

1
2
3
4
5

Schritt 5: Modell anwenden

FACHHOCHSCHULE
WIENER NEUSTADT
Austrian Network for Higher Education

Neuronales Netz – Meine Ziffer 🎨

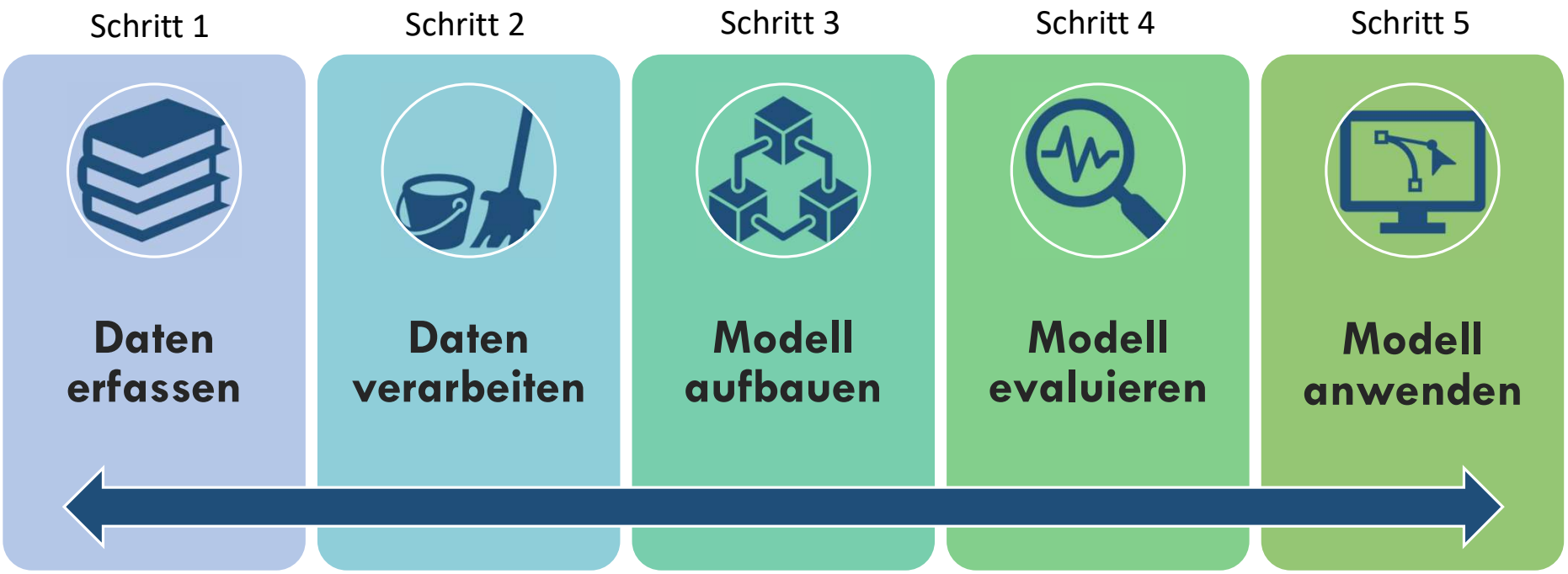
```
import matplotlib.image as mpimg
img = mpimg.imread('drei.png')
imgplot = plt.imshow(img)
plt.show()
x_my = img
x_my = np.squeeze(x_my[:,:,0])
x_my = np.squeeze(x_my)
x_my = x_my.reshape(-1,1)

y_my = make_predictions(x_my,W1,b1,W2,b2)
print(y_my)
```

1
2
3
4
5

Conclusio

FACHHOCHSCHULE WIENER NEUSTADT
Austrian Network for Higher Education

Eine kleine Zusammenfassung vom Workshop…



1 2 3 4 5