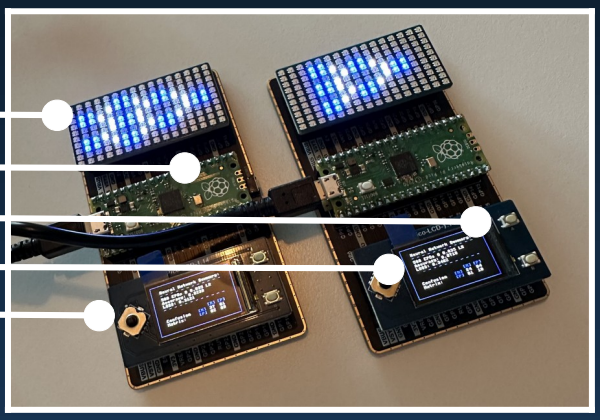
KÜNSTLICHE INTELLIGENZ: ENNA

(E)in (N)euronales (N)etz zum (A)usprobieren

---------- GRUNDLAGEN / BASISFUNKTIONEN ----------

ENNA ermöglicht Schülerinnen und Schülern ab der 10. Klasse den **intuitiven Einblick** in die **Funktionsweise von Neuronalen Netzen** als Grundlage der Künstlichen Intelligenz. Auf Basis eines vorprogrammierten **Raspberry Pi Pico** lassen sich über ein **LCD Display** mit zwei **Buttons** und einem **Joystick** zwei unterschiedliche Neuronale Netze auswählen, deren Aufbau und Lernverhalten anschließend auf einer **RGB Matrix** visualisiert werden. Die Neuronalen Netze, entweder mit **8 Neuronen in 4 Layern** oder **6 Neuronen in 3 Layern**, weisen ein unterschiedliches Lernverhalten auf, welches abschließend auf dem LCD Display ebenso wie der zugrundeliegende **Datensatz** visualisiert werden.



---- LERNZIELE ----

a) Aufbau eines **Microcontrollers**

b) **MicroPython Code** auf Microcontroller **installieren**

c) **Daten** für Klassifikationsprobleme verstehen

d) Architektur **Neuronaler Netze**

---------- ERWEITERTE FUNKTIONEN ----------

In **ENNA** können selbstverständlich auch **eigene Daten** für Klassifikationsprobleme eingespielt werden. Hierzu wird mit diesen in **Python** ein Neuronales Netz trainiert und dessen **Parameter** in die für **Microcontroller** geeignete Programmiersprache **MicroPython** überführt. In MicroPython lernen die Schülerinnen und Schüler schließlich die Funktionsweise von **Neuronen** und deren **Aktivierungsfunktionen** kennen. Fortgeschrittene können dadurch **flexibel** die Anzahl an Layern und Neuronen anpassen und **eigene Neuronale Netze programmieren**.

---- ERWEITERTE LERNZIELE ----

e) Grundlagen der Programmiersprachen **Python** mit TensorFlow und **MicroPython**

f) **Thonny** und **Anaconda** als kostenlose on- und offline Entwicklungsumgebungen

g) **Datenaufbereitung**, Architektur, **Training** und **Aktivierungsfunktionen**

ReLu

```python
# ReLU function
def relu(x):
    y = []
    for i in range(len(x)):
        if x[i] >= 0:
            y.append(x[i])
        else:
            y.append(0)

    return y
```

Sigmoid

```python
# Sigmoid function
def sigmoid(x):
    import math
    z = [1 / (1 + math.exp(-x[kk])) for kk in range(len(x))]
    return z
```

---------- HARD- / SOFTWARE ----------

Hardware:
* Raspberry Pi Pico H (mit Headern)
* Waveshare Dual GPIO Expander (SKU 19343)
* Waveshare 1.14 Pico Touch Display (SKU 19340)
* Waveshare 16x10 LED Matrix (SKU 20170)

Software:
* https://thonny.org/
* https://anaconda.cloud/

Idee und Umsetzung:
* Dennis Klinkhammer (2024)

ANLEITUNG & CODES