Praktikum 04 - NumPy und ASCII-Art

Die Bibliothek NumPy

NumPy ist eine Programmbibliothek für die Programmiersprache Python, die eine einfache Handhabung von Vektoren, Matrizen oder generell großen mehrdimensionalen Arrays ermöglicht. Neben den Datenstrukturen bietet NumPy auch effizient implementierte Funktionen für numerische Berechnungen an. (https://de.wikipedia.org/wiki/NumPy, Aufruf 24.11.2022)

1. NumPy verwenden

Minimalbeispiel zu Erstellung eines NumPy-Arrays. Es hat sich durchgesetzt bei der Nutzung von NumPy immer den Alias "np" zu nutzen.

```
import numpy as np

array_as_list = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

my_first_numpy_array = np.array(array_as_list)
```

$$\text{my\_first\_numpy\_array} = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

1.1 Arrays erstellen

Gegeben seien folgende Matrizen:

$$A = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad C = \begin{pmatrix} 5 & 5 & 5 \\ 5 & 5 & 5 \\ 5 & 5 & 5 \end{pmatrix}$$

Erstellen Sie die Matrizen als NumPy-Arrays ohne Listen zu benutzen. Nutzen Sie stattdessen folgende Funktionen: `np.ones()`, `np.eye()`, `np.full()`. Informationen zu den benötigten Parametern der Funktionen finden Sie in diesem Abschnitt der NumPy Dokumentation (Aufruf 24.11.2022)

Darüber hinaus soll eine weitere Matrix erstellt werden, welche alle ungeraden Zahlen von 1 bis 127 enthalten soll. Die Matrix soll 8 Spalten und 8 Zeilen besitzen. Nutzen Sie dafür ausschließlich die Funktionen `np.arange()` (Dokumentation, Aufruf 24.11.2022) und `np.reshape()` (Dokumentation, Aufruf 24.11.2022).

1.2 Informationen auslesen

Erstellen Sie eine Funktion `get_numpy_array_infos(array)`, welche folgende Informationen über das Array in der Konsole ausgibt:

- `dtype` : Datentyp der Array-Elemente
- `size` : Größe des Arrays; Anzahl Spalten mal Anzahl Zeilen
- `shape` : Dimensionen des Arrays als Tupel (Zeilen, Spalten)
- `min` : kleinster vorkommender Wert im Array
- `max` : größter vorkommender Wert im Array

Die Parameter `dtype`, `size`, `shape` können direkt aus einem NumPy-Objekt ausgelesen werden. Der kleinste und größte Wert des Arrays können über die Funktionen `np.max()` bzw. `np.min()` ermittelt werden.

Geben Sie die Informationen des 8x8-Arrays aus Aufgabe 1.1 aus.

1.3 Array normieren

Erstellen Sie eine Funktion `min_max_scaling(array)`, welche das Array auf den Wertebereich 0 bis 1 normalisiert. Das normierte Array X_{norm} lässt sich mit folgender Formel berechnen:

$$X_{norm} = \frac{X - x_{min}}{x_{max} - x_{min}}$$

- X = eingegebenes Array
- X_{norm} = normiertes Array
- x_{max} = größter Wert des Arrays
- x_{min} = kleinster Wert des Arrays

Lassen Sie sich die Informationen vom ursprünglichen Array und das normierte Array ausgeben. Nutzen Sie dafür die selbst erstellte Funktion aus Aufgabe 1.2. Welche Unterschiede gibt es?

1.4 Mittelwerte bilden

Mit der Funktion `np.mean()` (Dokumentation, Aufruf 28.11.2022) lassen sich unterschiedliche Mittelwerte eines Arrays berechnen. Berechnen Sie von `my_first_numpy_array` :

- den Mittelwert aller Koeffizienten
- die Mittelwerte der Zeilenelemente
- die Mittelwerte der Spaltenelemente

Nutzen Sie dafür den Input-Parameter `axis` der `np.mean()`-Funktion. Dokumentation lesen!

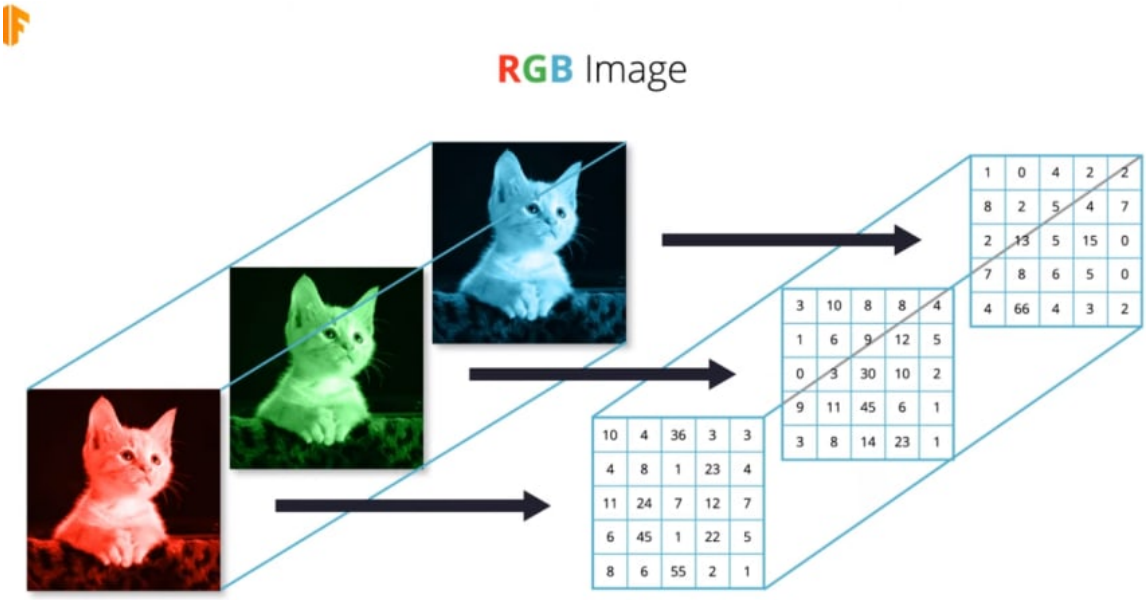
2. ASCII-Art

In dieser Aufgabe geht es darum ein Bild in Python zu laden, in ASCII-Art in Form eines Strings umzuformen und in der Konsole auszugeben. Dafür werden die Bibliotheken Pillow und NumPy verwendet.

Die Struktur einer Bilddatei entspricht dabei einem dreidimensionalem Array in folgender Form:

- Die ersten beiden Dimensionen stehen für die Pixelbreite und die Pixelhöhe
- Bei einem Bild in Full HD-Auflösung entspräche das 1920 × 1080
- Die dritte Dimension steht für die drei Farbkanäle Rot, Grün und Blau (RGB)
- Das bedeutet, dass bei jedem Pixel drei Werte hinterlegt sind
- Die Struktur eines Bildes in Form eines Arrays wird von NumPy mit (Höhe, Breite, Farbkanal) angegeben, ein Full HD-Bild hätte also die Form (1080, 1920, 3)

In folgender Grafik wird diese Struktur veranschaulicht, mit der Annahme, dass es sich um ein Bild der Auflösung 5 x 5 handele:



https://dev.to/sandeepbalachandran/machine-learning-going-furthur-with-cnn-part-2-41km, Aufruf 28.11.2022

Hätten wir das Bild aus der Grafik als NumPy-Array vorliegen könnten wir exemplarisch den ersten Pixel des blauen Kanals ansprechen mit `image_array[0, 0, 2]`. Wie bereits in Aufgabe 1.4 erwähnt werden die Index-Plätze bei NumPy `axis` genannt. Im Fall eines RGB-Bildes gibt es also drei Achsen bzw. drei Dimensionen im Array.

2.1 RGB zu Graustufenbild

Schreiben Sie eine Funktion `image_to_gray_scale(array)`, die als Input das NumPy-Array des Bildes erhält und ein Graustufenbild zurückgibt. Dafür müssen die drei Farbkanäle zu einem Kanal zusammenfasst werden, indem der Mittelwert entlang der Farbkanal-Achse gebildet wird.

2.2 Konvertierung von Array zu ASCII

Folgende Schritte sind durchzuführen:

- Lade das Bild und speichere es in einer Variablen vom Typ NumPy-Array (ist bereits vorbereitet)
- Informationen zu dem Array ausgeben lassen über die Funktion aus Aufgabe 1.2
- Konvertiere zu einem Graustufenbild über die Funktion aus Aufgabe 2.1
- Normieren des Arrays über die Funktion aus Aufgabe 1.3
- Erneut die Informationen des vorbearbeitenden Arrays ausgeben und vergleichen
- Konvertierung jedes Pixelwerts zu einem bestimmten Zeichen
- Die zu nutzenden Zeichen sind in einer Liste vorgegeben