Kraków, 2023

Program do odczytywania hieroglifów

Autor:

R. W.

Cel projektu

Projekt miał na celu napisanie programu zdolnego do rozpoznawania i odczytywania inskrypcji zapisanych pismem hieroglificznym. Najważniejszym aspektem tworzonego programu jest rozpoznawanie i interpretacja obrazów, a także zdolność do zapisu fonetycznego rozszyfrowanych piktogramów.

Zakres pracy

Aby zrealizować zadanie, należało znaleźć sposób, który umożliwi odnalezienie w pliku graficznym konkretnych piktogramów, a także posortowanie ich w pożądanej kolejności, by następnie odczytać znaczenie znaków.

Wykorzystane narzędzia

Do realizacji zadania skorzystano z języka Python i bibliotek Scikit-Image i NumPy.

Charakterystyka biblioteki Scikit-Image

Biblioteka scikit-image (skimage) to popularna biblioteka do przetwarzania obrazów w języku Python. Zapewnia wiele funkcji i narzędzi do manipulacji, analizy i wizualizacji obrazów. Oto kilka kluczowych cech i możliwości biblioteki scikit-image:

Wczytywanie i zapisywanie obrazów: scikit-image umożliwia łatwe wczytywanie i zapisywanie obrazów w różnych formatach, takich jak PNG, JPEG, TIFF itp. Można używać funkcji io.imread() do wczytywania obrazów i io.imsave() do zapisywania obrazów.

Przetwarzanie obrazów: biblioteka scikit-image zawiera szeroki zakres funkcji do przetwarzania obrazów, takich jak zmiana rozmiaru, przycinanie, obracanie, skalowanie, korekcja jasności i kontrastu, filtracja, segmentacja, detekcja krawędzi, ekstrakcja cech itp. Można zastosować je do obrazów za pomocą prostych i intuicyjnych interfejsów.

Porównywanie obrazów: scikit-image oferuje narzędzia do porównywania obrazów, takie jak dopasowywanie szablonu (match_template), które można używać do znalezienia podobieństwa między dwoma obrazami. Funkcje te są przydatne do wykrywania wzorców, dopasowywania obiektów i lokalizacji podobnych regionów w obrazach.

Transformacje geometryczne: biblioteka umożliwia stosowanie różnych transformacji geometrycznych na obrazach, takich jak translacja, rotacja, skalowanie, odwracanie itp. Te transformacje są przydatne przy manipulacji obrazami i analizie geometrii obiektów.

Wizualizacja obrazów: scikit-image zapewnia narzędzia do wizualizacji obrazów i wyników przetwarzania obrazów. Można używać funkcji imshow() do wyświetlania obrazów, a także dostosowywać różne aspekty wizualizacji, takie jak mapa kolorów, skalowanie, etykiety osi itp.

Operacje na pikselach: biblioteka umożliwia wykonanie różnych operacji na pikselach obrazu, takich jak dostęp do wartości pikseli, manipulacja wartościami pikseli, obliczanie histogramu, progowanie, binaryzacja itp.

Segmentacja obrazu: scikit-image dostarcza metody segmentacji obrazów, które pozwalają na identyfikację i podział obrazów na różne regiony na podstawie cech, takich jak kolor, tekstura, intensywność itp. Można zastosować segmentację do rozpoznawania obiektów, analizy danych i innych zastosowań.

Charakterystyka biblioteki NumPy

Biblioteka NumPy (Numerical Python) to popularna biblioteka do obliczeń naukowych i numerycznych w języku Python. Jest jednym z podstawowych narzędzi wykorzystywanych w ekosystemie naukowym Pythona. Oto kilka kluczowych cech i możliwości biblioteki NumPy:

Tablice wielowymiarowe: NumPy wprowadza nowy obiekt tablicy wielowymiarowej o nazwie ndarray. To jest elastyczna i wydajna struktura danych, która pozwala przechowywać i przetwarzać dane numeryczne w jednym lub więcej wymiarach. Tablice NumPy mogą mieć różne kształty (np. 1D, 2D, 3D itd.) i przechowywać dane jednego typu (np. liczby całkowite, liczby zmiennoprzecinkowe).

Efektywne obliczenia na tablicach: NumPy oferuje zestaw potężnych funkcji do wykonywania matematycznych operacji na tablicach, takich jak dodawanie, odejmowanie, mnożenie, dzielenie, potęgowanie itp. Te operacje są zoptymalizowane pod kątem wydajności i mogą być stosowane na całych tablicach zamiast wykonywać pętle na pojedynczych elementach.

Szeroki zakres funkcji matematycznych: NumPy zapewnia wiele funkcji matematycznych, takich jak funkcje trygonometryczne, funkcje logarytmiczne, funkcje statystyczne, funkcje algebraiczne itp. Można z nich korzystać bezpośrednio na tablicach, co ułatwia przetwarzanie danych numerycznych.

Indeksowanie i wycinanie: NumPy oferuje elastyczne mechanizmy indeksowania i wycinania tablic. Można używać indeksów całkowitych, zakresów indeksów oraz zaawansowanych mechanizmów indeksowania, takich jak maski logiczne, do wyodrębniania i modyfikowania konkretnych elementów tablicy lub ich fragmentów.

Transmisja (broadcasting): Transmisja w NumPy to mechanizm automatycznego dopasowywania kształtu tablic, który pozwala na wykonywanie operacji na tablicach o różnych kształtach. Dzięki temu można wygodnie przeprowadzać operacje, które normalnie wymagałyby ręcznego powielania tablic lub pętli.

Rodzaje hieroglifów

Wedle polskiej Wikipedii występują trzy podstawowe rodzaje hieroglifów:

- Znaki fonetyczne – służyły do zapisywania dźwięków występujących w języku egipskim.
- Znaki ideograficzne – określały całe słowa określające przedmioty, które przedstawiały.
- Znaki determinatywne – służyły do pokazania, do jakiej klasy należy dany przedmiot. Na przykład siedząca figura bóstwa była używana przed zapisem imion królów i bogów.

Opis rozwiązania

- Na początku program pobiera z folderu source pliki w formacie png przedstawiające hieroglify i zapisuje je w słowniku, w którym kluczem jest nazwa pliku będąca zarazem znaczeniem piktogramu, a wartością jest obraz.
- Następnie pobierana jest inskrypcja, która ma zostać przetłumaczona.
- W kolejnym kroku program w pętli wyszukuje fragmenty inskrypcji, które są najbardziej podobne do obrazów w słowniku. Do kolejnego słownika zapisywany jest obraz i jego pozycja w osi x na inskrypcji.
- Zawartość drugiego słownika jest sortowana rosnąco wedle pozycji x.
- Program wyświetla na ekranie kolejne okna przedstawiające znaleziony piktogram, jego pozycję na inskrypcji zaznaczoną czerwoną obwódką i przedstawienie dopasowania hieroglifu na całej inskrypcji. Aby przejść do kolejnego okna należy zamknąć poprzednie okno.
- Na koniec program wypisuje w terminalu zapis fonetyczny ciągu. Jeśli występuje w tekście znak determinatywny, jest on zapisywany po znaku „-”.

```python
import numpy as np
import matplotlib.pyplot as plt
import os

from skimage import data, io
from skimage.feature import match_template

folder_path = 'source'

images = {}
hierogliphs_in_line = {}

for filename in os.listdir(folder_path):
    if filename.endswith(".PNG"):
        file_path = os.path.join(folder_path, filename)

        image = io.imread(file_path)
        image = image[:, :, 0]
        name = filename
        last_dot_index = name.rfind(".")
        name = name[:last_dot_index]
        images[name] = image

image = io.imread('inscription.PNG')
image = image[:, :, 0]

for element in images:
    result = match_template(image, images[element])
    ij = np.unravel_index(np.argmax(result), result.shape)
    hierogliphs_in_line[ij[1]] = element
    x, y = ij[::-1]

sorted_items = sorted(hierogliphs_in_line.items(), key=lambda x: x[0])
print("These hieroglyphs mean: ", end="")
for element in sorted_items:
    print(element[1], end="")
    result = match_template(image, images[element[1]])
    ij = np.unravel_index(np.argmax(result), result.shape)
    x, y = ij[::-1]

    fig = plt.figure(figsize=(8, 3))
    ax1 = plt.subplot(1, 3, 1)
    ax2 = plt.subplot(1, 3, 2)
    ax3 = plt.subplot(1, 3, 3, sharex=ax2, sharey=ax2)

    ax1.imshow(images[element[1]], cmap=plt.cm.gray)
    ax1.set_axis_off()
    ax1.set_title('template')

    ax2.imshow(image, cmap=plt.cm.gray)
    ax2.set_axis_off()
    ax2.set_title('image')

    hcoin, wcoin = images[element[1]].shape
    rect = plt.Rectangle((x, y), wcoin, hcoin, edgecolor='r', facecolor='none')
    ax2.add_patch(rect)

    ax3.imshow(result)
    ax3.set_axis_off()
    ax3.set_title('`match_template`\nresult')

    ax3.autoscale(False)
    ax3.plot(x, y, 'o', markeredgecolor='r', markerfacecolor='none', markersize=10)
    plt.show()
```

Dane użyte do prezentacji działania programu

Jako przykładową inskrypcję wykorzystano zapis z książki Marka Colliera i Billa Manley'a „How to read Egyptian hieroglyphs" oznaczający słowo „zły".

Składa się on z następujących piktogramów:

Noga odpowiada fonetycznemu zapisowi litery „b".

Pióro jest odpowiedzialne za zapis litery „i".

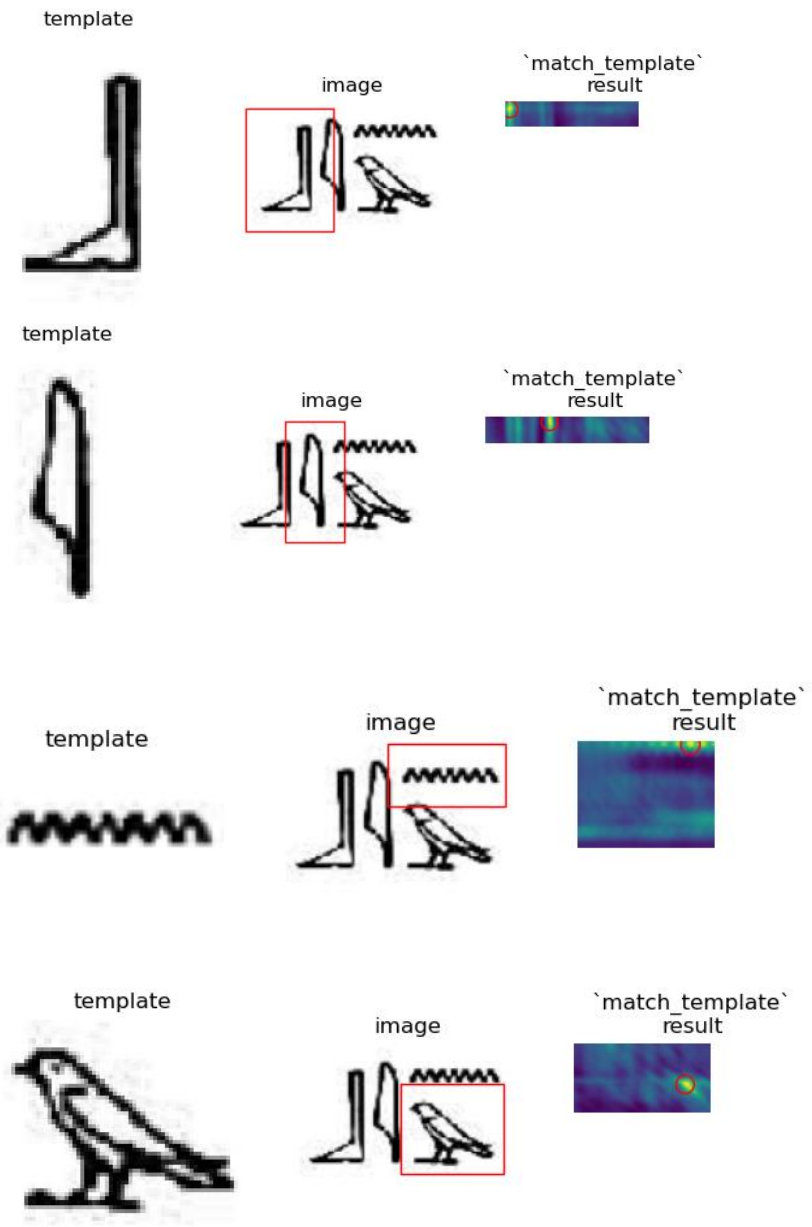
Fala oznacza fonetyczny zapis „n".

Mały ptak jest znakiem determinatywnym i jest używany by pokazać, że słowo odpowiada czemuś złemu, słabemu lub małemu.

Inskrypcja w zapisie fonetycznym powinna być czytana jako „bin".

Wynik działania programu

Program wyświetla kolejne okna przedstawiające rozpoznanie piktogramów w podanej inskrypcji. Zapis jest czytany z lewej do prawej.



Prócz tego, w terminalu jest wyświetlany zapis fonetyczny inskrypcji wraz z podaniem znaczenia znaku determinatywnego (po pauzie).

```
These hieroglyphs mean: bin-bad,weak
```

Możliwości rozbudowy programu

Obecnie program nie radzi sobie w sytuacji, w której dany znak występuje kilkukrotnie w inskrypcji. Należałoby zmodyfikować go tak, by można było odczytać zapis, który zawiera dowolną liczbę powtarzających się piktogramów.

Kolejną bolączką programu jest próba wyszukania obszaru najbardziej odpowiadającego znakowi nawet w sytuacji, gdy dany piktogram nie występuje w inskrypcji. Potrzebne jest rozsądne

oszacowanie, przy jakim poziomie podobieństwa należy zakwalifikować znak jako występujący w sprawdzanym obrazie.

Innym udogodnieniem, jakie należałoby zaimplementować jest możliwość przekazywania nowych plików graficznych do wyszukiwania w nich hieroglifów bez potrzeby ponownego uruchamiania programu.

Bibliografia

- Mark Collier, Bill Manley „How to read Egyptian hieroglyphs”, 1999
- https://pl.wikipedia.org/wiki/Hieroglify
- https://scikit-image.org/
- https://numpy.org/