System do odczytywania hieroglifów przy użyciu scikit-image

KK
TK

Cel projektu

Problemem rozpatrywanym w zadaniu było stworzenie programu, który jest w stanie rozpoznać hieroglif widoczny na wczytanym obrazku. Rozpoznawanie hieroglifu miało być realizowane przy użyciu biblioteki scikit-image.

Oznaczenia

W programie używaliśmy hieroglifów z listy znaków Gardinera. Zawiera ona około 800 znaków hieroglificznych. Została sporządzona przez Alana Gardinera. Każdy element tej listy jest oznaczony poprzez literę oraz numer. Litera oznacza kategorie, którą przedstawia hieroglif, a numer oznacza określony wariant wyglądowy. Hieroglify mają często wiele znaczeń. Dużo hieroglifów nie zostało przetłumaczonych do dziś.

Wynik programu

Program uczy się rozpoznawać hieroglify na danych dostarczonych ze zbioru danych pozyskanego z repozytorium Morrisa Frankena. W celu przetłumaczenia hieroglifu z obrazka należy wpisać odpowiednią ścieżkę do zdjęcia. Program przy pomocy biblioteki scikit-image oraz wcześniej wytrenowanego modelu zwraca oznaczenie hieroglifu, który najbardziej pasuje do odczytanego z obrazu.

Zaimportowane biblioteki

```
import os
import numpy as np
import joblib
from skimage.io import imread
from skimage.color import rgb2gray
from skimage.transform import resize
from sklearn.preprocessing import LabelBinarizer
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

Wczytywanie i przetwarzanie obrazów

```
dataset_path = directory_path
image_files = [os.path.join(dataset_path, f) for f in os.listdir(dataset_path) if f.endswith('.png')]

X = []
y = []
for image_file in image_files:
    image = imread(image_file)
    image = resize(image, (55, 55))
    label = image_file.split('_')[1].split(".")[0]
    if(label != 'UNKNOWN'):
        X.append(image)
        y.append(label)

X = np.array(X)
y = np.array(y)
```

Przygotowanie danych i zapis do plików

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.15)

np.save('hieroglyphs_train_data.npy', X_train)
np.save('hieroglyphs_train_labels.npy', y_train)

# Zapisanie danych testowych i etykiet do plików .npy
np.save('hieroglyphs_test_data.npy', X_test)
np.save('hieroglyphs_test_labels.npy', y_test)
```

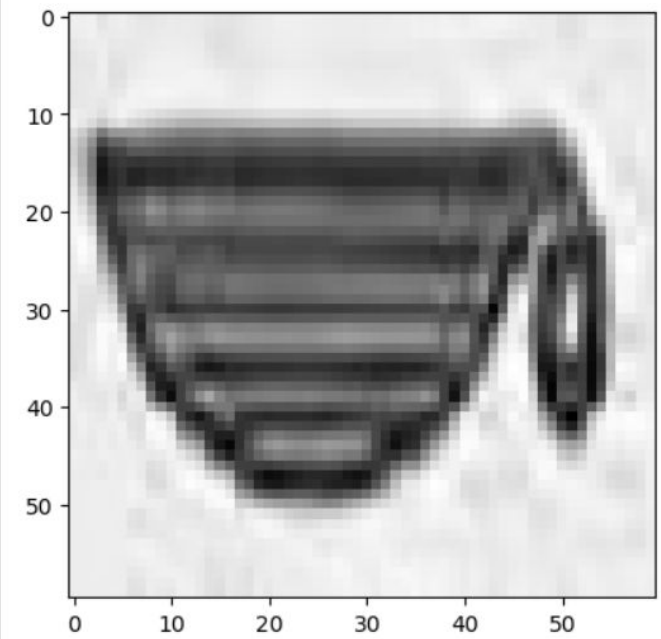
Trenowanie modelu

```
X_train = np.load('hieroglyphs_train_data.npy')
y_train = np.load('hieroglyphs_train_labels.npy')

X_test = np.load('hieroglyphs_test_data.npy')
y_test = np.load('hieroglyphs_test_labels.npy')

model = MLPClassifier(hidden_layer_sizes=(148, 148), max_iter=25)

X_train = X_train.reshape(X_train.shape[0], -1)

model.fit(X_train, y_train)

score = model.score(X_test.reshape(X_test.shape[0], -1), y_test)

joblib.dump(model, 'hieroglyphs_recognition_model.pkl')
```

Rozpoznawanie hieroglifu

```
# Wczytaj obraz, przekonwertuj go na szarość i zmień rozmiar
image = imread(image_path)
image = resize(image, (55, 55))
image = rgb2gray(image)
# Wyświetl obraz
plt.imshow(image, cmap='gray')
plt.show()
image = image.reshape(1, 3025)

predicted_label = model.predict(image)
print(predicted_label)
```

Wynik programu




```
Dopasowanie:

['V31']
```

Podsumowanie

Zaimplementowany przez nas program rozpoznaje hieroglify z wczytanego obrazu i wyświetla ich oznaczenie jako wynik. Założenia początkowe systemu zostały spełnione.

Bibliografia

https://github.com/morrisfranken/glyphreader

https://pl.wikipedia.org/wiki/Lista_znak%C3%B3w_Gardinera