Uniwersytet Zielonogórski
Instytut Sterowania i Systemów Informatycznych

Rozpoznawanie Obrazów

Semestr VII, Informatyka

mgr inż. Marcin Skobel

2021

Projekt część 2: Przygotowanie paczek do sieci neuronowej.

I. Część teoretyczna

Wstęp

Bieżąca cześć projektu polega na implementacji systemu wycinania obrazów treningowych i testowych o rozmiarze 256x256, które stanowić będą wsad do uczenia i testowania sztucznej splotowej sieci neuronowej.

Plan systemu

1. Zbiór ROI + Obrazki.
2. Generujemy etykiety z plików ROI:
- Wczytujemy do systemu pliki ROI i generujemy z nich obrazki.
- Krawędź jądra komórkowego powstaje w wyniku połączenia podwójnej erozji obiektu pierwotnego za pomocą elementów strukturalnych: [1 1 1, 1 1 1, 1 1 1] oraz [0 1 0, 1 1 1, 0 1 0] oraz pojedynczej dylatacji obiektu pierwotnego za pomocą elementu strukturalnego: [1 1 1, 1 1 1, 1 1 1]
- Ostatecznie w celu sprawdzenia działania algorytmu dajemy poszczególnym obiektom następujące etykiety: 0 dla tła, 128 dla krawędzi jądra komórkowego oraz 255 dla wnętrza.
- Kolejne obiekty ROI mają się nakładać na siebie, nowa wartość piksela zastępuje istniejącą
3. Po sprawdzeniu poprawności powstałych obrazów zmieniamy wartości etykiet:
- 0 dla tła, 1 dla krawędzi jądra komórkowego oraz 2 dla wnętrza.
4. Wycinanie obrazów o rozmiarze 256x256:
Przygotowujemy obrazki o rozmiarze 256x256 które muszą spełniać kilka zasad: - gdy pętla dojdzie do krawędzi obrazka przesuwamy współrzędne o wartość którą wycięcie przekracza obraz wejściowy (zarówno horyzontalnie jak i wertykalnie).
- wycinamy jednocześnie obraz RGB jak i jego obraz odpowiednik z etykietami.

- odrzucamy wszystkie obrazki w których zawartość wnętrz jąder komórkowych nie przekracza 20% powierzchni obrazka.

II. Przykład praktyczny

```
#Konfiguracja pakietow
#!pip install read_roi - instalujemy pakiet read_roi tylko raz

from read_roi import read_roi_zip
from PIL import Image
import numpy as np
from matplotlib.path import Path
import cv2
import matplotlib.pyplot as plt
from google.colab import drive
drive.mount('/content/drive')
#Deklaracja plikow - ostatecznie trzeba utworzyc odpowiednia petle do przetwarzania wszyst
FileName = '4477_01'
#wczytanie obrazu
im = Image.open('drive/My Drive/Colab Notebooks/Images/'+FileName+'.tif')
w, h = im.size
#wczytanie pliku zip z ROI
ROIS = read_roi_zip('drive/My Drive/Colab Notebooks/Images/'+FileName+'.zip')
#odczyt wspolrzednych ze slownika ROIS do tablicy numpy
j=0
for k in ROIS.values():
    x = k.get('x')
    y = k.get('y')
    #liczba wspolrzednych k-tego ROI
    n = len(x)
    #generowanie wspolrzednych poligonu
    i=0
    ListOfCorners = []
    for i in range(i,n):
        ListOfCorners.append((int((y[i])), int((x[i]))))
    #generowanie masek poligonow
    poly_path = Path(ListOfCorners)
    Nx, Ny = np.mgrid[:h, :w]
    coordinates = np.hstack((Nx.reshape(-1, 1), Ny.reshape(-1, 1)))
    mask = poly_path.contains_points(coordinates)
    mask = mask.reshape(h, w)
    mask = np.array(mask, dtype=bool)
    mask = 255 * mask
    mask = np.array(mask, dtype='uint8')
    # algorytm jest bardzo zlozony obliczeniowo bedzie bardzo dlugo trwal
    # na tym etapie generuja sie maski binarne oznaczonych jader komorkowych
    # dalsze zadania do wykonania to:
    #1. Podwojna erozja maski, raz przy uzyciu elementu [[1,1,1],[1,1,1],[1,1,1]]
    # za drugim razem przy uzyciu elementu [[0,1,0],[1,1,1],[0,1,0]], Po erozji
    # otrzymujemy wnetrze jadra komorkowego.
```

```
#2. Odjecie od pierwotnej maski nowej zerodowanej maski w celu utworzenia
# pierwszej czesci krawedzi jadra.
#3. Pojedyncza dylatacja maski pierwotnej a nastepnie odjecie maski pierwotnej od
# nowej maski w celu uzyskania drugiej czesci krawedzi jadra komorkowego
#4. Zsumowanie obu czesci krawedzi do jednej maski.
#5. Na tym etapie mozna rozpoczac nakladanie sie obrazow masek na siebie
# z uwzglednieniem wartosci 0-tlo, 1-krawedz, 2 wnetrze jadra komorkowego
```

III. Uwagi

Plik z przykładami można pobrać ze strony: `http://staff.uz.zgora.pl/mskobel/RO/Projekt/RoiZipExtractor.ipynb`