Uniwersytet Zielonogórski
Instytut Sterowania i Systemów Informatycznych

Rozpoznawanie Obrazów

Semestr VII, Informatyka

mgr inż. Marcin Skobel

2022

Laboratorium nr 13: Metody ewaluacji wyników segmentacji

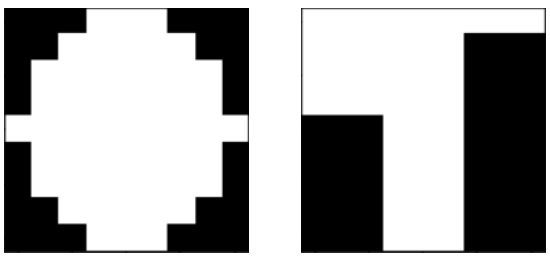
I. Zagadnienia teoretyczne

Wstęp

Segmentacja obrazu jest procesem w wyniku, którego otrzymujemy obiekty lub grupy obiektów jednorodnych np. jąder komórkowych na obrazie cyfrowym. Wynik segmentacji możemy jednocześnie uprościć przyjmując tylko interesującą nas klasę obiektów oraz tło. Sprowadzenie segmentacji do poziomu dwuklasowego pozwala na zastosowanie pewnych współczynników dokładności segmentacji. Jakościowa metoda ewaluacji w tym przypadku może polegać na wzrokowej ocenie jakości segmentacji, jednakże powstały pewne metody ilościowej ewaluacji pozwalające na określenie dokładności segmentacji w formie liczbowej. W niniejszym laboratorium poznamy podstawowe mierniki i współczynniki stosowane do oceny ilościowej dokładności segmentacji.

Indeks Jaccarda

Najbardziej intuicyjnym współczynnikiem, który możemy zastosować do porównanie dopasowania dwóch obiektów jest Indeks Jaccarda. Zakładamy, że te dwa obiekty o których mowa to ręcznie oznaczony obiekt oraz obiekt po segmentacji automatycznej. Porównanie tych dwóch obiektów da nam odpowiedź na pytanie jak dobrze została została wyznaczona segmentacja.

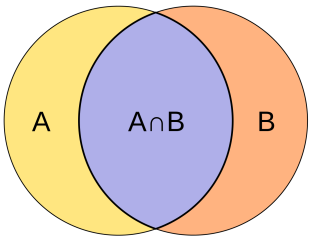


Rysunek 1: Przykładowe obiekty które możemy ze sobą porównać

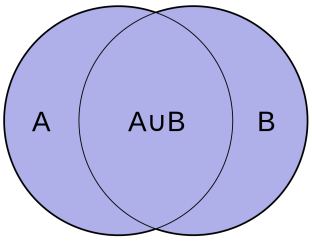
Indeks Jaccarda bazuje na algebrze zbiorów i przyjmuje poniższy wzór:

$$JI = \frac{|A \cap B|}{|A \cup B|}, \tag{1}$$

gdzie $|M \cap N|$ oznacza moc części wspólnej obu zbiorów Rys.2, natomiast $|M \cup N|$ moc ich sumy Rys.3.



Rysunek 2: Część wspólna zbiorów, źródło: wikipedia.org



Rysunek 3: Suma (Unia) zbiorów, źródło: wikipedia.org

Indeks Jaccarda przyjmuje wartości od 0 do 1. Im niższa wartość Indeksu Jaccarda tym niższe dopasowanie dwóch obiektów, natomiast wartość 1 oznacza pełne dopasowanie.

Indeks Dice-Sørensena

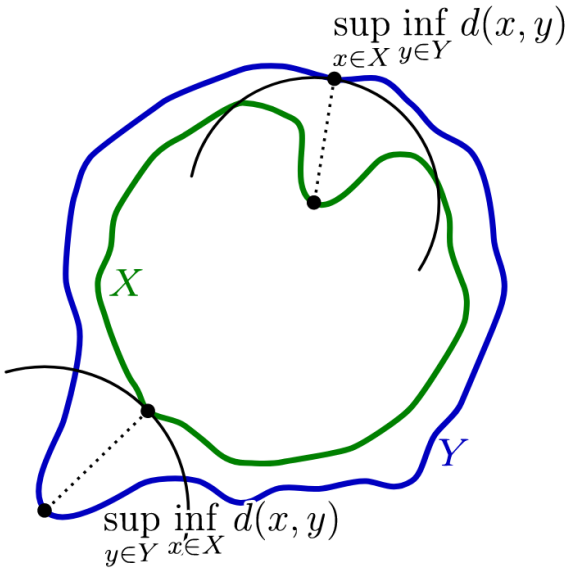
Kolejnym stosunkowo podobnym miernikiem jest Współczynnik Dice-Sørensena. Współczynnik ten jest równie intuicyjny jak Indeks Jaccarda. Wzór na obliczenie wartości współczynnika Dice-Sørensena wygląda następująco:

$$WDS = \frac{2|A \cap B|}{|A| + |B|}, \tag{2}$$

gdzie $|M \cap N|$ oznacza moc części wspólnej obu zbiorów Rys.2, natomiast $|A| + |B|$ stanowi dodawanie do siebie mocy dwóch testowanych zbiorów. Współczynnik Dice-so rensena przyjmuje również wartości od 0 do 1.

Metryka Hausdorffa

Metryka Hausdorffa jest najbardziej wyrafinowaną spośród przedstawianych metod. Najprościej ujmując odległość Hausdorffa jest to największa ze wszystkich odległości od punktu w jednym zestawie do najbliższego punktu w drugim zestawie.
Definicja matematyczny mówi natomiast, że jeżeli A i B będą dwoma niepustymi podzbiorami



Rysunek 4: Graficzna interpretacja Metryki Hausdorffa, źródło: wikipedia.org

przestrzeni metrycznej (M, d) to definiujemy ich odległość Hausdorffa HD(A,B) za pomocą wzoru:

$$HD(A,B) = \max\left\{\sup_{a\in A}\inf_{b\in B} \mathrm{d}(a,b), \sup_{b\in B}\inf_{a\in A} \mathrm{d}(a,b)\right\}, \tag{3}$$

gdzie sup reprezentuje supremum i inf na infimum. Dystans Hausdorffa może przyjmować wartości od 0 do ∞. Im niższa wartość metryki tym lepsze dopasowaniu dwóch porównywanych obiektów.

II. Przykład praktyczny

Konfiguracja wstępna

Na początku zaimportujemy wszystkie przydatne w tym zadaniu biblioteki:

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
from scipy.spatial.distance import cdist
```

Przygotowanie i przegląd danych do testów

W kolejnym kroku utworzymy kilka przykładowych obiektów do analizy:

```
A = np.array([[0,0,0,1,1,1,0,0,0],
              [0,0,1,1,1,1,1,0,0],
              [0,1,1,1,1,1,1,1,0],
              [0,1,1,1,1,1,1,1,0],
              [1,1,1,1,1,1,1,1,1],
              [0,1,1,1,1,1,1,1,0],
              [0,1,1,1,1,1,1,1,0],
              [0,0,1,1,1,1,1,0,0],
              [0,0,0,1,1,1,0,0,0],
])
B = np.array([[1,1,1,1,1,1,1,1,1],
              [1,1,1,1,1,1,0,0,0],
```

```
                [1,1,1,1,1,1,0,0,0],
                [1,1,1,1,1,1,0,0,0],
                [0,0,0,1,1,1,0,0,0],
                [0,0,0,1,1,1,0,0,0],
                [0,0,0,1,1,1,0,0,0],
                [0,0,0,1,1,1,0,0,0],
                [0,0,0,1,1,1,0,0,0],
])
C = np.array([[0,0,0,0,0,0,0,0,0],
                [0,0,0,0,0,0,0,0,0],
                [0,0,0,0,0,0,0,0,0],
                [0,0,0,0,0,0,0,0,0],
                [0,0,0,0,0,0,0,0,0],
                [0,1,1,1,1,1,1,1,0],
                [0,1,1,1,1,1,1,1,0],
                [0,0,1,1,1,1,1,0,0],
                [0,0,0,1,1,1,0,0,0],
])
D = np.array([[1,1,0,0,0,0,0,0,0],
                [1,1,1,0,0,0,0,0,0],
                [0,1,1,1,0,0,0,0,0],
                [0,0,1,1,1,0,0,0,0],
                [0,0,0,1,1,1,0,0,0],
                [0,0,0,0,1,1,1,0,0],
                [0,0,0,0,0,1,1,1,0],
                [0,0,0,0,0,0,1,1,1],
                [0,0,0,0,0,0,0,1,1],
])
ImageA = Image.fromarray(np.uint8(A*255))
ImageB = Image.fromarray(np.uint8(B*255))
ImageC = Image.fromarray(np.uint8(C*255))
ImageD = Image.fromarray(np.uint8(D*255))
plt.subplot(141), plt.imshow(ImageA, "gray"), plt.title("A")
plt.subplot(142), plt.imshow(ImageB, "gray"), plt.title("B")
plt.subplot(143), plt.imshow(ImageC, "gray"), plt.title("C")
plt.subplot(144), plt.imshow(ImageD, "gray"), plt.title("D")
plt.show(
```

Implementacja Indeksu Jaccarda

```
def JI(M,N):
  Size1 = np.size(M)
  Size2 = np.size(N)
  if Size1!=Size2:
      error('Matrices should be the same size!')
  Int = M & N
  Union = M | N
  I = sum(sum(Int))/sum(sum(Union))
  return I
#Test
JI(A,D)
```

Implementacja Współczynnika Dice-Sørensena

```
def DI(M,N):
    Size1 = np.size(M)
    Size2 = np.size(N)
    if Size1!=Size2:
        error('Matrices should be the same size!')
    Int = sum(sum(M&N))
    Sum = sum(sum(M))+sum(sum(N))
    I = 2*Int/Sum
    return I
#Test
DI(A,D)
```

Implementacja Metryki Hausdorffa

```
def HD(M,N):
    Size1 = np.size(M)
    Size2 = np.size(N)
    if Size1!=Size2:
        error('Matrices should be the same size!')
    XY_M1 = np.column_stack(np.where(M > 0))
    XY_N1 = np.column_stack(np.where(N > 0))
    D1 = cdist(XY_M1,XY_N1)
    D2 = cdist(XY_N1,XY_M1)
    min1 = D1.min(axis=1)
    min2 = D2.min(axis=1)
    D = np.max([np.max(min1), np.max(min2)]);
    return D
#Test
HD(A,D)
```

III. Uwagi

Plik z przykładami można pobrać ze strony: `http://staff.uz.zgora.pl/mskobel/lab13.py`

IV. Lista zadań

1. Wykonaj obliczenia wszystkich współczynników dla każdej pray obiektów (A, B, C, D). 2. Porównaj wyniki i wskaż, które obiekty są najbardziej podobne do siebie według poszczególnych współczynników.