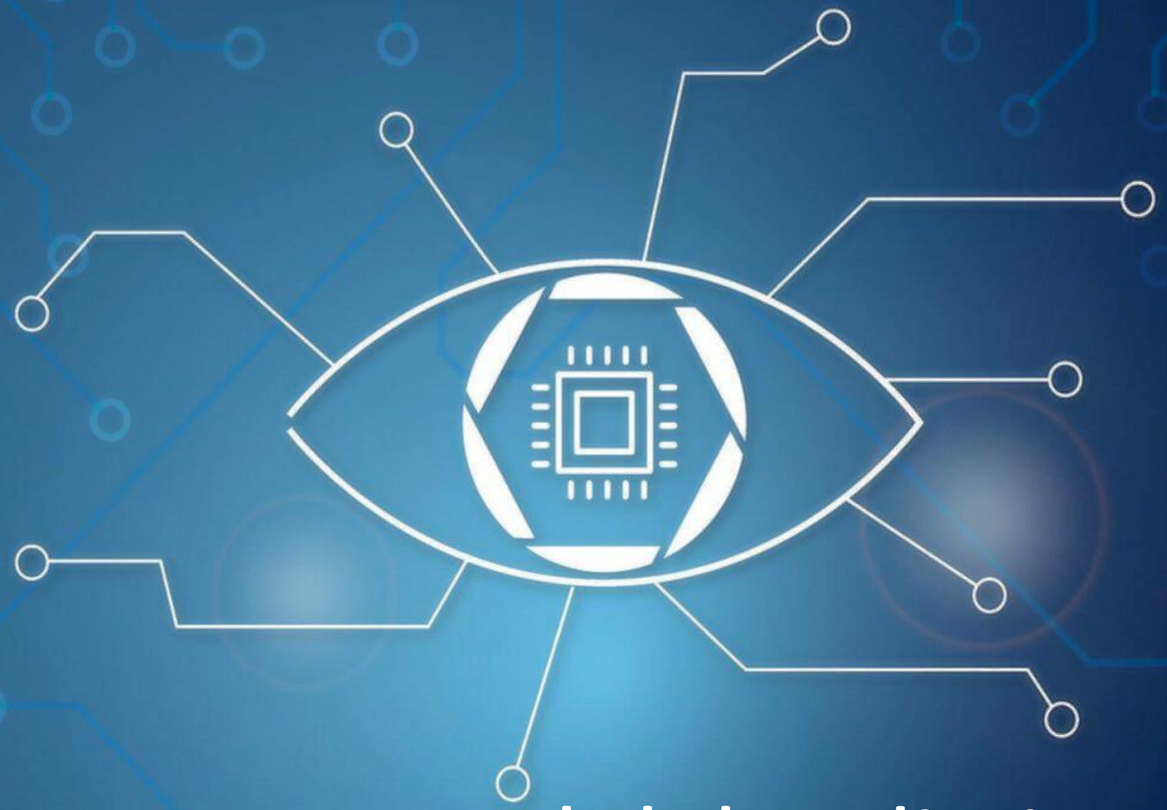


Segmentarea celulelor din imagini medicale

Agavriloaei Marina

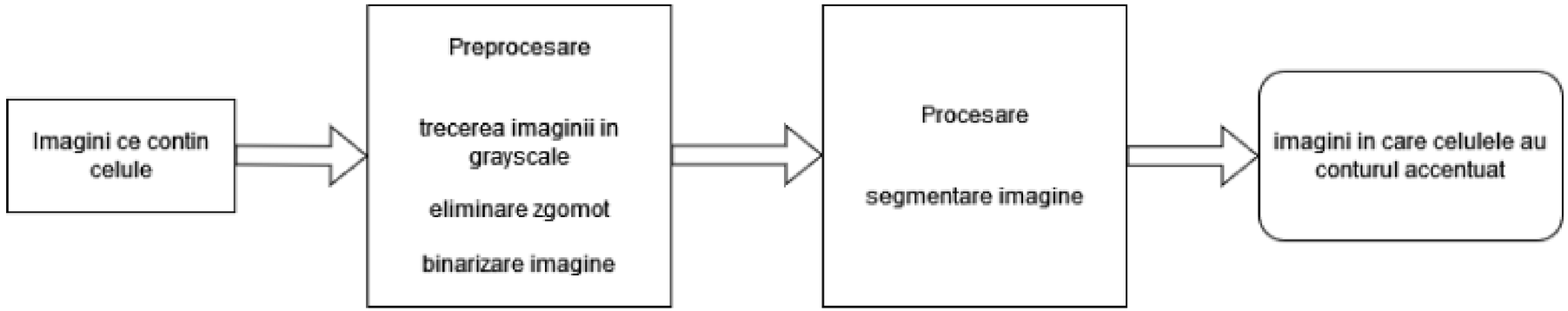
Moisuc Raluca-Elena

1. Context & Motivație

- **Context:** Cerința proiectului este realizarea unui algoritm care poate segmenta cu o acuratețe cât mai ridicată celulele din imagini medicale
- **Motivație:** Segmementarea automatizată a celulelor economisește timp, si poate ajuta la detecția timpurie a anomaliilor, ajuta la reducerea erorilor umane și poate ajuta la monitorizarea evoluției bolilor
- **Obiectivul proiectului:** De a evidenția forma celulelor , prin conturarea marginilor acestora și de a crea măști care suprapuse peste o imaginea initială ne vor arata regiuni de interes (în cazul dat doar celulele, sau doar fundalul zonei in care se afla celulele)

2. Arhitectura preliminară a soluției

- **Schema arhitecturii:**



- **Descrierea componentelor:**
- Cei doi algoritmi prezentati in cod au in comun:
- În etapa de preprocesare utilizam metoda Otsu pentru gasirea unui threshold optim pentru binarizarea imaginii.
- Facem egalizarea de histograma pentru distibuirea intensitatilor luminioase în cazul imaginilor cu un interval limitat de valori

2. Arhitectura preliminară a soluției

Descrierea componentelor:

- 1. gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
- 2. clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8, 8))
- 3. enhanced_img = clahe.apply(gray_img)

2. Arhitectura preliminară a soluției

Descrierea componentelor:

- **4.blurred_img = cv2.GaussianBlur(input_img, (5, 5), 0)**

- **5. _, binary_img = cv2.threshold(blurred_img, 0, 255, cv2.THRESH_BINARY + cv2.THRESH_OTSU)**

- **6. kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (3, 3))**

- **7. opened_img = cv2.morphologyEx(binary_img, cv2.MORPH_OPEN, kernel, iterations=2)**

2. Arhitectura preliminară a soluției

Descrierea componentelor:

- **8. dist_transform = cv2.distanceTransform(opened_img, cv2.DIST_L2, 5)**

- **9. sure_bg = cv2.dilate(opened_img, kernel, iterations=3)**

- **10. unknown = cv2.subtract(sure_bg, sure_fg)**

- **11. _, markers = cv2.connectedComponents(sure_fg)**

2. Arhitectura preliminară a soluției

Descrierea componentelor:

- 8. cv2.watershed(img_for_watershed, markers)
- 9. sure_bg = cv2.dilate(opened_img, kernel, iterations=3)
- 10. pixelAccuracy = (TP + TN) / (TP + TN + FP + FN)
- print(f"Pixel accuracy is {pixelAccuracy * 100:.2f}%")
- # Jaccard Index
- IoU = TP / (TP + FN + FP) if (TP + FN + FP) != 0 else 0
- print(f"Area of overlap/Area of Union (IoU) is {IoU * 100:.2f}%")

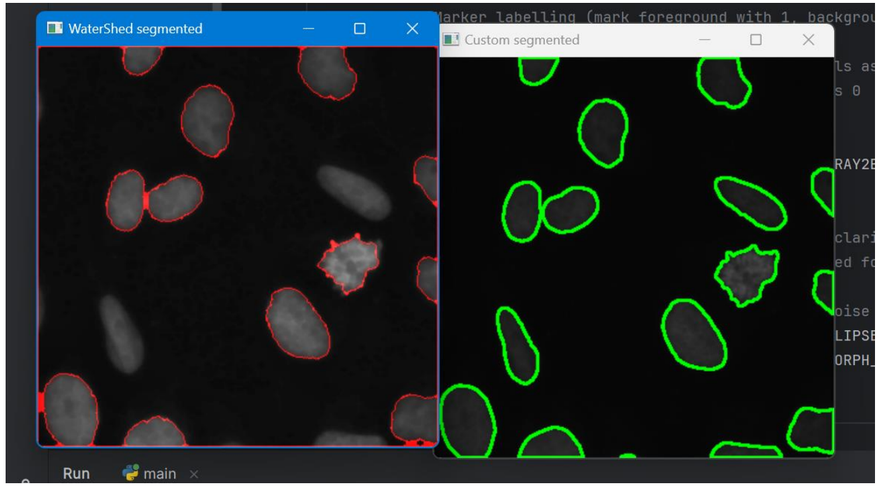
2. Arhitectura preliminară a soluției

Descrierea componentelor:

- 10. # Precision
- precision = TP / (TP + FP) if (TP + FP) != 0 else 0
- print(f"Precision is {precision * 100:.2f}%")
- # Recall
- recall = TP / (TP + FN) if (TP + FN) != 0 else 0
- print(f"Recall is {recall * 100:.2f}%")
- # Dice similarity coefficient
- F_measure = (2 * recall * precision) / (recall + precision) if (recall + precision) != 0 else 0
- print(f"F_measure is {F_measure:.2f}")

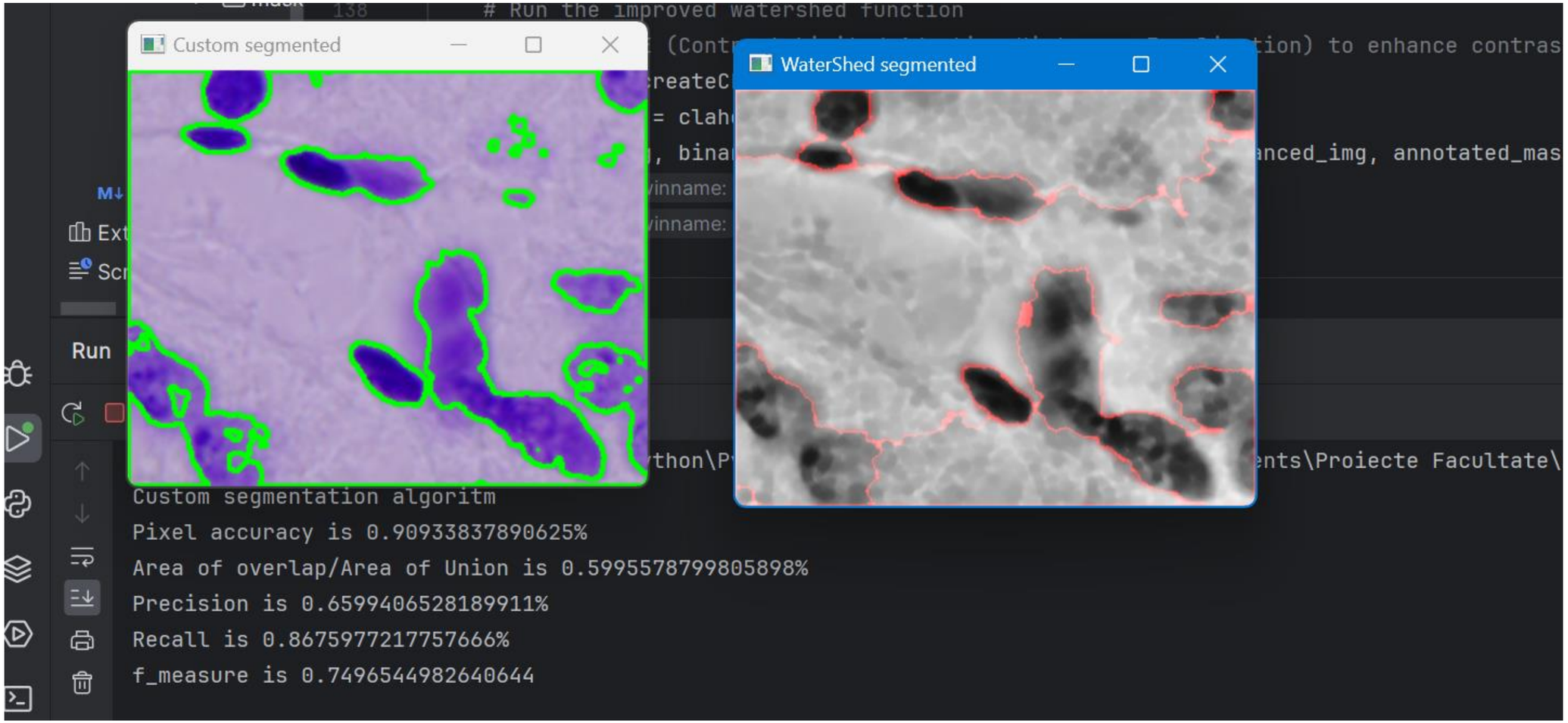
3. Evaluarea Preliminară a Soluției

- Metodologia de evaluare: accuratete,Jaccard Index, precizie,recall, F_measure
- Setul de date: Celule de diferite marimi si forme(RGB sau greyscale) care au venit cu un set de masti annotate
- Exemple de cazuri de test:



4. Rezultate Preliminare

- **Rezultate obținute:**



5. Concluzii Preliminare

- **Rezumatul progresului: Un algoritm creat intuitiv despre ce ar presupune segmentarea imaginii in mod eficient si algoritmul classic watershed**
- **Limitările soluției actuale: pentru imagini RGB, nivelurile de intensitate in masca sunt inversate , celulele se suprapun**
- **Potențiale îmbunătățiri: Adaugarea si compararea cu alti algoritmi de segementarea in cautarea unuia cu accuratete mai ridicata**

6. Direcții Viitoare

- **Pași următori: Pentru algoritmi deja implementati watershed trebuie sa poata recunoste mai concret marginile unei celule si in algoritmul custom putem aborda mai eficient zgomotele**
- **Plan de implementare: Adaugarea de transformari morfologice Obiectivele finale: ne-am dori sa putem obtine o serie de algoritmi din care sa putem alege care este cel mai eficient in functie de statistici**