

AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

DZIEDZINA NAUK ŚCISŁYCH I PRZYRODNICZYCH

DYSCYPLINA NAUKI FIZYCZNE

ROZPRAWA DOKTORSKA

Komputerowe metody analizy i przetwarzania obrazów medycznych

Autor: mgr inż. Mateusz Buczkowski

Promotor rozprawy: prof. dr hab. inż. Khalid Saeed

Praca wykonana: Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie, Wydział Fizyki i Informatyki Stosowanej

Kraków, 2024

Oświadczenie autora rozprawy:

Oświadczam, świadomy odpowiedzialności karnej za poświadczenie nieprawdy, że niniejszą pracę doktorską wykonałem osobiście i samodzielnie i że nie korzystałem ze źródeł innych niż wymienione w pracy.

data, podpis autora

Oświadczenie promotora rozprawy:

Niniejsza rozprawa jest gotowa do oceny przez recenzentów.

data, podpis promotora rozprawy

Rozprawa doktorska powstała przy udziale finansowania pochodzącego ze stypendium doktoranckiego IUVENES – KNOW, Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie, Wydział Fizyki i Informatyki Stosowanej.

Promotor niniejszej pracy, Pan prof. dr hab. Inż. Khalid Saeed, okazywał mi wsparcie i zrozumienie przez te wszystkie lata pracy oraz poświęcał swój czas, za co jestem mu ogromnie wdzięczny i składam podziękowania.

Szczególnie pragnę podziękować mojej Żonie Martynie za wyrozumiałość oraz wsparcie. Dziękuję za te wszystkie soboty, niedziele i wieczory kiedy zajmowałaś się dziećmi, a ja pracowałem nad doktoratem. Dziękuję moim dzieciom: Adrianie, Brunowi i Gabrielowi. Przez całe wasze dotychczasowe życie tata zajmował się doktoratem jednocześnie pracując, a każda godzina pracy była godziną mniej z wami. Dziękuję również moim rodzicom, Mamie Danucie i Tacie Tadeuszowi za okazane wsparcie.

SPIS TREŚCI

| 1 | WPROWADZENIE | 4 |
|---|------------------------|---|
| | 1.1 Cel i teza pracy | 4 |
| | 1.2 Struktura rozprawy | 5 |

CZĘŚĆ TEORETYCZNA

| 2 | PODS | STAWY | TEOR | ETYCZNE | CYFROWEGO | PRZETWARZANIA |
|---|----------|----------|---------------|---------------|-----------------------|-------------------------|
| C | BRAZÓ | W | | | ••••• | 7 |
| | 2.1 Wp | rowadze | enie do cyfro | owego przetwa | arzania obrazów | 7 |
| | 2.2 Rep | rezenta | cja obrazów | cyfrowych | | 9 |
| | 2.3 Kor | nwersja | do skali szar | rości | | |
| | 2.4 Hist | togram. | | | | |
| | 2.4.1 | Wyrów | wnanie histo | gramu | | |
| | 2.5 Zale | eżności | pomiędzy | pikselami - | pojęcie sąsiedztwa, | przylegania, łączności, |
| | regionów | v oraz g | ranice | | | |
| | 2.6 Filt | rowanie | e przestrzenn | ie | | |
| 3 | MET | ODY FI | ILTRACJI | WSTĘPNEJ | OBRAZÓW | 40 |
| | 3.1 Wp | rowadze | enie | | | |
| | 3.2 Met | tody kla | syczne | | | |
| | 3.2.1 | Filtr m | nedianowy | | | |
| | 3.2.2 | Filtr u | średniający . | | | |
| | 3.2.3 | Filtr b | ilateralny | | | |
| | 3.2.4 | Odejm | nowanie tła | | | |
| | 3.2.5 | Limito | owane kontra | astem adaptac | yjne wyrównanie histo | gramu (CLAHE) 45 |
| 4 | MET | ODY SI | EGMENTA | CJI OBRAZ | ÓW | |
| | 4.1 Wp | rowadze | enie | | | |
| | 4.2 Prze | ekształc | enia morfolo | ogiczne | | |
| | 4.2.1 | Mapa | odległości | | | |
| | 4.2.2 | Erozja | ι | | | |

| 4.2.3 | Dylatacja | . 48 |
|---------|--|------|
| 4.3 Met | tody bazujące na intensywności pikseli | . 49 |
| 4.3.1 | Algorytm Otsu | . 49 |
| 4.3.2 | Progowanie z histerezą | . 49 |
| 4.3.3 | Algorytm wododziałowy | . 50 |
| 4.3.4 | Statystyczne łączenie obszarów (ang. Statistical Region Merging - SRM) | . 50 |
| 4.4 Met | tody bazujące na wykrywaniu krawędzi | . 51 |
| 4.4.1 | Detektor krawędzi Canny–Deriche | . 51 |
| 4.5 Met | tody bazujące na modelach | . 53 |
| 4.5.1 | Modele deformowalne - Aktywne kontury (ang. Active Contours, Snakes). | . 53 |
| 4.5.2 | GVF (ang. Gradient Vector Flow) | . 54 |
| 4.6 Met | tody bazujące na sztucznych sieciach neuronowych | . 55 |
| 4.6.1 | Podstawy teoretyczne | . 55 |

CZĘŚĆ DOŚWIADCZALNA

| 5.1 Wprowadzenie | 64 |
|---|---------|
| 5.2 Opis Algorytmu | 67 |
| 5.2.1 Przetwarzanie indywidualnych obiektów | 71 |
| 5.2.2 Dostosowanie wartości parametrów | 75 |
| 5.3 Wyniki | 77 |
| 5.3.1 Ewaluacja oraz porównanie wyników | |
| 5.4 Podsumowanie | 83 |
| 6 SEGMENTACJA ERYTROCYTÓW Z OBRAZÓW Z MIKR | OSKOPU |
| OPTYCZNEGO WRAZ Z OBLICZANIEM STOSUNKU ICH WYSOKO | DŚCI DO |
| SZEROKOŚCI | |
| 6.1 Wprowadzenie | |
| 6.2 Opis Algorytmu | 85 |
| 6.2.1 Weterne przetwarzanie i segmentacja | |

| 6.2.2 Indywidualne przetwarzanie i segmentacja obiektów |
|--|
| 6.2.3 Prezentowana technika segmentacji jako procesor map prawdopodobieństwa |
| uzyskanych przy użyciu głębokiego uczenia |
| 6.2.4 Opisywana technika segmentacji połączona z uczeniem maszynowym w celu |
| kategoryzacji wyników |
| 6.3 Wyniki 101 |
| 6.3.1 Porównanie wyników ewaluacji do podejścia zgodnego ze stanem wiedzy. 101 |
| 6.4 Dyskusja wyników 106 |
| 6.5 Podsumowanie |
| 7 OPRACOWANIE OBRAZÓW POCHODZĄCYCH Z MIKROSKOPU |
| SIŁ ATOMOWYCH W CELU UWIDOCZNIENIA POWIERZCHNI |
| ERYTROCYTÓW 110 |
| 7.1 Wprowadzenie |
| 7.2 Opis Algorytmu |
| 7.3 Wyniki |
| 7.4 Podsumowanie |
| 8 WNIOSKI KOŃCOWE 118 |
| 8.1 Podsumowanie wyników pracy 118 |
| 8.2 Oryginalne elementy pracy 119 |
| 8.3 Rozważania na temat przyszłości i kierunków rozwoju algorytmów analizy |
| i przetwarzania obrazów w tym obrazów medycznych120 |
| BIBLIOGRAFIA |
| SPIS UŻYWANYCH SKRÓTÓW 154 |
| SPIS RYSUNKÓW 155 |
| SPIS TABEL 162 |

1 Wprowadzenie

1.1 Cel i teza pracy

Cele niniejszej rozprawy to:

- Przedstawienie opracowanych przez autora niniejszej rozprawy skutecznych metod cyfrowego przetwarzania i analizy obrazów, a w szczególności segmentacji obrazów medycznych bazujących na algorytmach klasycznych, oraz algorytmach klasycznych wspartych sztucznymi sieciami neuronowymi, a następnie ocena wyników i ich porównanie pomiędzy różnymi metodami.
- Połączenie różnych podejść w celu stworzenia algorytmów wykorzystujących zalety każdego z nich np. wykorzystanie klasycznych algorytmów segmentacji obrazów w modelach opartych na sztucznych sieciach neuronowych.
- Nakreślenie własnego spojrzenia na prawdopodobne kierunki rozwoju dziedziny komputerowych metod cyfrowego przetwarzania i analizy obrazów w związku z dynamicznym rozwojem metod opartych na sztucznych sieciach neuronowych.

Tezy niniejszej rozprawy to:

- Klasyczne metody segmentacji pozwalają uzyskać satysfakcjonujące wyniki segmentacji w zastosowaniach rozpatrywanych w niniejszej pracy.
- Klasyczne metody segmentacji mogą stanowić uzupełnienie algorytmów bazujących na głębokim uczeniu i odwrotnie.
- W rozpatrywanych zastosowaniach indywidualne przetwarzanie obiektów znajdujących się na obrazie pozwala zwiększyć jakość segmentacji kosztem złożoności obliczeniowej.

1.2 Struktura rozprawy

Niniejsza rozprawa została podzielona na dwie części: teoretyczną oraz doświadczalną. Część teoretyczna zawiera informacje o podstawach związanych z cyfrowym przetwarzaniem i analizą obrazów w zakresie pozwalającym na zrozumienie metod użytych w części doświadczalnej. Zawiera ona również krótkie opisy tych metod. Część doświadczalna przedstawia rozwiązania problemów z dziedziny cyfrowego przetwarzania i analizy obrazów ze szczególnym uwzględnieniem segmentacji obrazów czyli wyodrębnienia obszarów będących obiektami na obrazie. Ostatni rozdział zawiera rozważania o możliwych kierunkach rozwoju dziedziny w kontekście dynamicznego rozwoju metod opartych na sztucznych sieciach neuronowych, który to rozwój znacznie przyspieszył w okresie ostatniego roku przed ukończeniem niniejszej rozprawy.

W wielu miejscach w rozprawie oprócz nazw spolszczonych podawane są nazwy angielskie. Wynika to z faktu, że w dziedzinie cyfrowego przetwarzania i analizy obrazów powszechnym jest stosowanie angielskich określeń opisujących wiele pojęć.

Zdecydowana większość rysunków zamieszczonych w rozprawie zostało przygotowanych przez jej autora. W przypadkach gdy były one tworzone na podstawie rysunków innych autorów jest to odpowiednio oznaczone.

CZĘŚĆ TEORETYCZNA

2 Podstawy teoretyczne cyfrowego przetwarzania obrazów

2.1 Wprowadzenie do cyfrowego przetwarzania obrazów

Najbardziej ogólną definicją obrazu cyfrowego jest przedstawienie go w postaci dwuwymiarowej funkcji f(x, y) gdzie x oraz y oznaczają koordynaty przestrzenne. Wartość funkcji z = f(x, y) dla każdej pary skończonych i dyskretnych koordynat (x, y)nazywamy intensywnością lub poziomem szarości dla obrazu monochromatycznego [1]. Obraz cyfrowy składa się więc ze skończonej liczby elementów zwanych pikselami z przypisanymi koordynatami (x, y) oraz wartością z spełniającą te same warunki co koordynaty przestrzenne [1, 2]. Obraz można również przedstawić jako macierz, gdzie wartości jej poszczególnych elementów odpowiadają intensywności poszczególnych pikseli, a jej wymiar rozmiarowi obrazu [3]. Źródła cyfrowych obrazów mogą być różne. Głównym źródłem nasuwającym się jako pierwsze są wszelkiego rodzaju kamery cyfrowe operujące w zakresie światła widzialnego ale również mogące wykonywać zdjęcia w paśmie podczerwieni. Kamery takie można również łączyć z odpowiednią optyką jak ma to miejsce np. przy zastosowaniu kamery cyfrowej w celu przechwycenia obrazu z mikroskopu optycznego. Patrząc szerzej, takimi źródłami moga być różne metody bazujące na rejestracji promieniowania elektromagnetycznego akwizycji obrazu jak np. tomografia czy mikro-tomografia operujące w zakresie promieniowania rentgenowskiego. Obraz cyfrowy uzyskać można również przy pomocy aparatu USG gdzie źródłem obrazu jest odpowiednio przetworzony sygnał uzyskany z wykorzystaniem ultradźwięków. Innym przykładem jest mikroskopia sił atomowych (ang. atomic force microscope - AFM) gdzie obraz powstaje w wyniku skanowania powierzchni próbki z wykorzystaniem sił oddziaływań międzyatomowych. Obrazy mogą mieć również od razu pochodzenie cyfrowe. Mogą być wygenerowane przez algorytmy lub stworzone przy pomocy programów do tworzenia grafiki komputerowej. Ogólnie w formie obrazu cyfrowego można przedstawić każdą funkcję dwuwymiarową ze skończonymi i dyskretnymi koordynatami oraz wartościami zgodnie z definicją przedstawioną na początku niniejszego paragrafu [1, 2, 4]. Tworząc obraz cyfrowy z danych o ciągłej naturze należy przeprowadzić proces próbkowania przestrzennego oraz kwantyzacji wartości [1, 5].

Nie została ustanowiona ścisła granica pomiędzy przetwarzaniem obrazów (ang. image processing), a analiza obrazów (ang. image analysis) czy wizją komputerową (ang. computer vision) wiec rozsądnym wydaje się spojrzenie na te obszary jako przetwarzanie obrazów nisko, średnio i wysoko poziomowe [1]. Do kategorii przetwarzania niskopoziomowego zaliczyć można proste operacje jak np. odszumianie różnego rodzaju filtrami czy różnego rodzaju maski konwolucyjne mające na celu np. wyostrzenie obrazu [1, 2, 3, 4]. Pod pojęciem przetwarzania średniopoziomowego kryją się bardziej skomplikowane algorytmy które ekstrahuja atrybuty z obrazu jak np. segmentacja obiektów znajdujących się na obrazie [1, 2, 3, 4]. Przetwarzanie wysokopoziomowe mieści w sobie z kolei wnioskowanie na podstawie obiektów wykrytych na obrazie oraz ich zależności [1]. Przykładowo zalicza się tu algorytm mówiący jakie emocje okazuje osoba przedstawiona na obrazie [6]. Algorytmy opisane w niniejszej pracy można zaliczyć głównie do dwóch pierwszych grup czyli przetwarzania nisko i średniopoziomowego z elementami przetwarzania wysokopoziomowego. Dla zobrazowania algorytm opisany w rozdziale 6 łączy w sobie wszystkie trzy poziomy przetwarzania obrazu. Etap filtracji wstępnej i segmentacji erytrocytów zalicza do przetwarzania niskopoziomowego się i średniopoziomowego, a etap kategoryzujący te erytrocyty jako normalne, anormalne i wykrywający błędną segmentację przynależy do grupy trzeciej czyli przetwarzania wysokopoziomowego. Trudność z wyznaczeniem ostrej linii podziału kategoryzującej algorytmy można zobrazować sposobem przetwarzania obrazu wewnątrz konwolucyjnych sieci neuronowych, które przecież wzorowane są na procesie przetwarzania obrazu w ludzkim mózgu. Kolejne warstwy sieci wykrywają coraz to bardziej abstrakcyjne i złożone wzorce poczawszy od warstw wykrywających np. krawędzie a skończywszy na warstwach wykrywających coraz to bardziej złożone wzorce (rozdział 4.6). Przejście pomiędzy kolejnymi warstwami w kontekście przypisania do wyżej wymienionych kategorii jest płynne co wydaje się stanem naturalnym. Tenże rozdział przedstawia kilka podstawowych pojęć potrzebnych aby zrozumieć operacje mieszczące się w wymienionych kategoriach.

2.2 Reprezentacja obrazów cyfrowych

Obrazy cyfrowe reprezentowane są jako zbiór pikseli zawierający M wierszy oraz N kolumn z dyskretnymi koordynatami oznaczanymi jako (x, y). Numeracja koordynat rozpoczyna się od zera: x = 0,1,2,..., M - 1 dla wierszy oraz y = 0,1,2,..., N - 1 dla kolumn. W ten sposób obraz reprezentowany jest jako dwuwymiarowa tablica (równanie (2.1) [1]). Zwyczajowo przyjęto konwencję umiejscowienia koordynat (0,0) w lewym górnym rogu obrazu oraz kartezjański układ współrzędnych obrócony o 90° (Rysunek 2.1) [1, 2].

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1,N-1) \end{bmatrix}$$
(2.1)

Najczęściej używana ilość poziomów kwantyzacji wartości to 256 [7]. Taki obraz jest zwany ośmiobitowym (8-bit). Dla takiego obrazu wartości intensywności mieszczą się w zakresie od 0 do 255. W ten sposób reprezentowany jest obraz w skali odcieni szarości. Obraz kolorowy najczęściej reprezentowany jest przez zestaw trzech takich wartości odpowiednio dla kanałów czerwonego, zielonego i niebieskiego (RGB) [1, 2, 7].



Rysunek 2.1 Układ współrzędnych stosowany do opisu obrazu cyfrowego (układ kartezjański obrócony o 90°) gdzie centra pikseli posiadają współrzędne opisane liczbami całkowitymi.

2.3 Konwersja do skali szarości

Najpopularniejszymi metodami konwersji obrazu RGB do skali szarości czyli wyznaczenia pojedynczej wartości na podstawie trzech odpowiadających poszczególnym kolorom są uśrednianie (Równanie (2.2)) oraz uśrednianie z wykorzystaniem średniej ważonej (Równanie (2.3)) [8].

$$g(x,y) = \frac{f(x,y,R) + f(x,y,G) + f(x,y,B)}{3}$$
(2.2)

$$g(x,y) = \alpha_R \cdot f(x,y,R) + \alpha_G \cdot f(x,y,G) + \alpha_B \cdot f(x,y,B)$$
(2.3)

gdzie:

g(x, y) – dwuwymiarowa funkcja opisująca intensywność piksela w skali szarości o zadanych koordynatach x, y

f(x, y, z) – trójwymiarowa funkcja opisująca intensywność piksela w modelu RGB o zadanych koordynatach przestrzennych *x*, *y* oraz kanale koloru $z \in R, G, B$

 $\alpha_R, \alpha_G, \alpha_B$ – nieujemne współczynniki dla odpowiednich kanałów koloru spełniające zależność opisaną równaniem (2.4) [8]

$$\alpha_R + \alpha_G + \alpha_B = 1 \tag{2.4}$$

Współczynniki dla kanałów poszczególnych kolorów w metodzie uśredniania za pomocą średniej ważonej mają za zadanie odzwierciedlić sposób w jaki ludzkie oko postrzega poszczególne kolory. Popularne wartości współczynników to w zaokrągleniu:

 $\alpha_R = 0.30$, $\alpha_G = 0.59$, $\alpha_B = 0.11$ [8]. Można więc zapisać równanie opisujące sposób wyznaczenia intensywności dla piksela (*x*, *y*) (Równanie (2.5)).

$$I_{x,y} = 0.30 \cdot R_{x,y} + 0.59 \cdot G_{x,y} + 0.11 \cdot B_{x,y}$$
(2.5)

gdzie:

 $I_{x,y}$ – intensywność piksela o koordynatach (x, y)

 $R_{x,y}$ – intensywność kanału czerwonego dla piksela o koordynatach (x, y)

 $G_{x,y}$ – intensywność kanału zielonego dla piksela o koordynatach (x, y)

 $B_{x,y}$ – intensywność kanału niebieskiego dla piksela o koordynatach (x, y)

2.4 Histogram

Histogram jest sposobem prezentacji danych o rozkładzie intensywności pikseli rozważanego obrazu w postaci jednowymiarowej funkcji. Przedstawia on liczbę pikseli posiadających poszczególne wartości intensywności. Histogram może być wyznaczony dla obrazu w skali szarości (Rysunek 2.2, Rysunek 2.3) lub obrazu kolorowego (Rysunek 2.5). Przykładowo obraz kolorowy przedstawiony w modelu RGB może mieć wyznaczone trzy histogramy (dla każdego kanału koloru osobno) (Rysunek 2.5). Liczba poziomów wartości intensywności pikseli obrazu wynika z jego głębi bitowej (ilości poziomów kwantyzacji). Przykładowo obraz w skali szarości z głębią 8-bit posiada 256 możliwych wartości intensywności piksela (od 0 do 255) [7]. Podobnie obraz kolorowy w modelu RGB posiadający głębię 8-bit na kanał posiada 256 możliwych wartości na kanał co daje 16 777 216 kombinacji kolorów [7]. Definiując nieznormalizowany histogram $h(r_k)$ przyjmujemy liczbę możliwych poziomów intensywności piksela obrazu jako L oraz r_k jako poszczególne wartości intensywności, gdzie k = 0, 1, 2, ..., L - 1. Prowadzi to do następującego równania [1]:

$$h(r_k) = n_k \tag{2.6}$$

gdzie n_k jest ilością pikseli obrazu posiadających intensywność r_k . Znormalizowany histogram $p(r_k)$ możemy zdefiniować wykorzystując równanie (2.6) w postaci [1]:

$$p(r_k) = \frac{h(r_k)}{M \cdot N} = \frac{n_k}{M \cdot N}$$
(2.7)

gdzie:

M – ilość wierszy obrazu

N – ilość kolumn obrazu

Suma $p(r_k)$ dla wszystkich możliwych wartości k (możliwych poziomów intensywności) wynosi 1, co zapisano równaniem (2.8) [1, 2].

$$\sum_{k=0}^{L-1} p(r_k) = 1$$
 (2.8)

Tworząc histogram można zastosować grupowanie wartości intensywności w przedziały (ang. *bins*) (Rysunek 2.4) [1, 2]. Operacje przeprowadzane na histogramie są jednym z podstawowych narzędzi przetwarzania obrazów. Z tego względu, że są niezbyt kosztowne i proste obliczeniowo używane są często do przetwarzania obrazów w czasie rzeczywistym. Przykładem algorytmu bazującego na histogramie jest progowanie metodą Otsu (rozdział 4.3.1).



Rysunek 2.2 Przykładowy obraz zawierający trzy obiekty posiadające różne poziomy intensywności (a) oraz odpowiadający mu histogram (b) na którym można zaobserwować, że na obrazie występują cztery poziomy intensywności (trzy dla obiektów i jeden największy odpowiadający tłu).



Rysunek 2.3 Przykładowy rzeczywisty obraz pochodzący z mikroskopu optycznego w skali szarości (8-bit) zawierający wiele obiektów posiadających różne poziomy intensywności (a) oraz odpowiadający mu histogram (b) na którym można zaobserwować, że na obrazie występują dwa maksima liczby pikseli (jedno lokalne i jedno globalne) dla konkretnej intensywności: jedno w okolicy poziomu intensywności równego 26, a drugie w okolicy 212.



Rysunek 2.4 Histogramy z grupowaniem w przedziały ilości pikseli dla zadanego zakresu wartości intensywności: 64 przedziały (a), 32 przedziały (b), 16 przedziałów (c).



Rysunek 2.5 Przykładowy rzeczywisty kolorowy obraz pochodzący z mikroskopu optycznego, przedstawiający erytrocyty w modelu RGB (8-bit na kanał) zawierający wiele obiektów posiadających różne poziomy intensywności (a) oraz dopowiadające mu histogramy dla kanału R (b), G (c) oraz B (d) na których można zaobserwować różnice pomiędzy histogramami dla poszczególnych kanałów.

2.4.1 Wyrównanie histogramu

Wyrównanie histogramu jest przekształceniem mającym na celu zwiększenie kontrastu obrazu. Dla przypadku obrazu o małym kontraście (Rysunek 2.6 a) oraz jego wariantów w postaci obrazu jasnego (Rysunek 2.7 a) i ciemnego (Rysunek 2.7 b) można pokazać, że obrazy z małym kontrastem posiadają wartości pikseli skupione w wąskim przedziale możliwych wartości. Zwiększenie bądź zmniejszenie jasności obrazu nie zmienia jego kontrastu. Zmiana jasności obrazu objawia się na histogramie jako jego przesunięcie. Wyrównanie histogramu powoduje rozproszenie wartości intensywności pikseli w taki sposób aby wykorzystać cały dostępny zakres poziomów intensywności tak aby liczba pikseli w każdym zakresie intensywności na histogramie była podobna [4]. Inaczej mówiąc rozkład intensywności pikseli obrazu dąży do jednorodności (Rysunek 2.6 d) [1, 2, 4, 5].



Rysunek 2.6 Przykładowy obraz o małym kontraście (a) i jego histogram (c) oraz obraz o dużym kontraście (b) i jego histogram (d). Na podstawie [1].



Rysunek 2.7 Przykładowy obraz o dużej jasności (a) i jego histogram (c) oraz obraz o małej jasności (b) i jego histogram (d). Na podstawie [1].

Wyrównanie histogramu przeprowadzone jest dla poziomów intensywności piksela będących wartościami dyskretnymi co w takim przypadku powoduje, że równania zapisujemy na bazie prawdopodobieństwa i sumowania. Prawdopodobieństwo wystąpienia piksela z poziomem intensywności r_k dla danego obrazu zapisujemy jako [1]:

$$p_r(r_k) = \frac{n_k}{MN} \tag{2.9}$$

gdzie:

MN – całkowita liczba pikseli obrazu

 n_k – liczba pikseli posiadających wartość r_k

 $p_r(r_k)$ – prawdopodobieństwo wystąpienia piksela o wartości r_k , zwane często znormalizowanym histogramem. Dla założenia: $r_k \in [0, L-1]$, gdzie: L jest liczbą możliwych poziomów intensywności piksela obrazu.

Równanie pozwalające zmapować każdy piksel obrazu wejściowego o intensywności r_k na odpowiadający mu piksel obrazu wyjściowego z wartością intensywności s_k zapisujemy jako [1]:

$$s_k = T(r_k) = (L-1) \sum_{j=0}^k p_r(r_j)$$
 (2.10)

gdzie k = 0, 1, 2, ..., L - 1

Zakładając obraz 3-bit o rozdzielczości 32×32 pikseli mamy L = 8, k = 0,1,2,...,7 oraz MN = 1024. Zakładamy wartości pikseli:

| r_k | n_k | $p_r(r_k)$ |
|---------------------------|-------|------------|
| $r_0 = 0$ | 199 | 0.19 |
| $r_{1} = 1$ | 250 | 0.24 |
| <i>r</i> ₂ = 2 | 210 | 0.21 |
| r ₃ = 3 | 170 | 0.17 |
| $r_4 = 4$ | 80 | 0.08 |
| r ₅ = 5 | 60 | 0.06 |
| $r_{6} = 6$ | 30 | 0.03 |
| r ₇ = 7 | 25 | 0.02 |

Tabela 2.1 Ilość pikseli dla zakładanego obrazu oraz prawdopodobieństwo wystąpienia piksela o zadanej wartości w zależności od poziomu intensywności. Na podstawie [1].

W wyniku zastosowania równania (2.10) otrzymujemy:

 $s_0 = 1.33 \approx 1, \ s_1 = 3.01 \approx 3, \ s_2 = 4.48 \approx 4, \ s_3 = 5.67 \approx 6, s_4 = 6.23 \approx 6,$ $s_5 = 6.65 \approx 7, \ s_6 = 6.86 \approx 7, \ s_7 = 7.00 \approx 7$ Jak można zauważyć kilka poziomów intensywności oryginalnego obrazu mapowane jest na ten sam poziom intensywności w obrazie wynikowym. Powoduje to bardziej jednorodny oraz rozciągnięty na większy zakres możliwych wartości intensywności rozkład intensywności pikseli (Rysunek 2.8) [1, 2, 4, 5].



Rysunek 2.8 Histogram oryginalnego obrazu (a) oraz histogram tego samego obrazu po zastosowaniu metody wyrównania histogramu (b).

2.5 Zależności pomiędzy pikselami - pojęcie sąsiedztwa, przylegania, łączności, regionów oraz granice

Obraz cyfrowy składa się z pikseli, które ułożone są w postaci dwuwymiarowej tablicy z wierszami i kolumnami (Rysunek 2.9).



Rysunek 2.9 Przykładowy obraz o rozmiarze 7 na 7 pikseli (7 wierszy i 7 kolumn) w skali szarości (8-bit) (a) oraz odpowiadający mu obraz z zaznaczonymi pikselami (b).

Każdy piksel *p* posiada koordynaty (x, y) i bardzo rzadko jest przetwarzany oddzielnie. Przeważnie wykorzystywany jest też jego kontekst czyli sąsiedztwo. Rozpatrywany piksel posiada dwóch poziomych i pionowych sąsiadów z koordynatami: (x + 1, y), (x - 1, y), (x, y + 1), (x, y - 1), gdzie takie sąsiedztwo oznaczane jest jako $N_4(p(x,y))$ (Rysunek 2.10) [1]. Na przekątnych znajdują się kolejne cztery sąsiednie piksele o koordynatach: (x + 1, y + 1), (x + 1, y - 1), (x - 1, y + 1), (x - 1, y - 1), gdzie tego typu diagonalne sąsiedztwo oznaczone jest jako $N_D(p(x,y))$ (Rysunek 2.10) [1]. Obie kategorie sąsiedztwa tworzą razem sąsiedztwo oznaczone jako $N_8(p(x,y))$, a zbiór pikseli sąsiadujących z wybranym pikselem p(x, y) zwany jest sąsiedztwem tego piksela [1, 2].



Rysunek 2.10 Piksel p(x, y) oraz jego sąsiedztwo $N_4(p(x, y))$ oznaczające sąsiadów poziomych i pionowych (a) oraz sąsiedzi po przekątnej $N_D(p(x, y))$ (b) (należy pamiętać, że kartezjański układ współrzędnych zastosowany dla obrazów cyfrowych jest obrócony o 90°).

Przyleganie pikseli definiuje się w odniesieniu nie tylko do położenia przestrzennego ale również do ich wartości (poziomu intensywności), a więc w celu zdefiniowania przyległości pikseli określamy zbiór V zawierający wartości intensywności pikseli użytych do zdefiniowania przyległości [1]. W najczęściej rozważanym przypadku obrazu binarnego (zdefiniowanego jako zawierający dwie wartości intensywności: 1 oraz 0) zbiór V definiuje się jako $V = \{1\}$ w przypadku gdy chcemy określić przyleganie pikseli o wartości 1 [1]. Analogicznie w przypadku chęci określenia przylegania pikseli o wartości 0 zbiór V definiuje się jako V = $\{0\}$ [1]. Podobnie dla obrazu w skali szarości można zdefiniować taki zbiór np. dla obrazu 8-bitowego w postaci poziomów intensywności większych od 128, a uogólniając dla przedziału możliwych wartości od 0 do 255 zbiór V można zdefiniować jako dowolny jego podzbiór [1]. Określając przyleganie piksela wyróżnia się trzy typy przylegania, gdzie wybierając poszczególne z nich wynik określenia przylegania może ulec zmianie [1]:

- 4-przyleganie (ang. 4-adjacency) piksele p i q posiadające wartości intensywności należące do zbioru V użytego do zdefiniowania przylegania są 4-przyległe jeżeli piksel q spełnia zależność: q ∈ N₄(p) (Rysunek 2.11)
- 8-przyleganie (ang. 8-adjacency) piksele p i q posiadające wartości intensywności należące do zbioru V użytego do zdefiniowania przylegania są 8-przyległe jeżeli piksel q spełnia zależność: q ∈ N₈(p) (Rysunek 2.11)
- m-przyleganie (ang. *m-adjacency*) zwane również przyleganiem mieszanym (ang. *mixed adjacency*) piksele p i q posiadające wartości intensywności należące do zbioru V użytego do zdefiniowania przylegania są m-przyległe jeżeli
 - piksel q spełnia zależność: $q \in N_4(p)$ lub
 - o piksel q spełnia zależność: q ∈ N_D(p) oraz zbiór pikseli zdefiniowany jako
 N₄(p) ∩ N₄(q) nie posiada pikseli z wartościami należącymi do zbioru
 V (Rysunek 2.11)

Powodem wprowadzenia m-przylegania jest niejednoznaczność 8-przylegania, ponieważ występuje tam przyleganie wielokrotne, a zastosowanie m-przylegania usuwa tę niejednoznaczność (Rysunek 2.11) [1].



Rysunek 2.11 Przykładowy układ pikseli gdzie założono zbiór $V = \{1\}$ (a), 4-przyleganie oznaczone liniami (b), 8-przyleganie oznaczone liniami (c), m-przyleganie oznaczone liniami (d). Na podstawie [1].

Ścieżka (ang. *path*) łącząca piksel $p(x_0, y_0)$ oraz piksel $q(x_n, y_n)$ jest definiowana jako sekwencja pikseli: $(x_0, y_0), (x_1, y_1), ..., (x_n, y_n)$, przy założeniu, że piksele (x_i, y_i) oraz (x_{i-1}, y_{i-1}) są przyległe dla $1 \le i \le n$, gdzie n jest długością ścieżki, a samą ścieżkę można określić przy wykorzystaniu różnych typów przylegania [1].

Definiując zbiór S będący podzbiorem zbioru A (spełniający zależność $S \subseteq A$) zawierającego wszystkie piksele rozpatrywanego obrazu oraz dwa piksele p oraz q możemy powiedzieć, że są one połączone w zbiorze S jeżeli istnieje pomiędzy nimi ścieżka składająca się tylko i wyłącznie z pikseli zawartych w zbiorze S [1]. Dla każdego piksela $p: p \in S$ zbiór pikseli połączonych z nim w zbiorze S nazywa się spójną składową (ang. *connected component*) zbioru *S*, a jeżeli w zbiorze *S* istnieje tylko jedna taka spójna składowa to wtedy zbiór *S* zwany jest zbiorem spójnym (ang. *connected set*) [1].

Przyjmijmy zbiór *R* będący podzbiorem zbioru *A* (spełniający zależność $R \subseteq A$) zawierającego wszystkie piksele rozpatrywanego obrazu [1]. Jeżeli zbiór *R* jest zbiorem spójnym nazywamy go regionem obrazu, a jeżeli suma dwóch rozważanych regionów R_i oraz R_j tworzy zbiór spójny to nazywamy je regionami przylegającymi [1]. W celu zdefiniowania regionów rozpatrywane jest 4-przyleganie oraz 8-przyleganie (

Rysunek 2.12), a regiony które nie są przyległe określamy jako rozłączne [1].

| $R \int 1$ | 1 | 1 |
|-------------|---|---|
| $n_i \ge 0$ | 1 | 0 |
| R.∫1 | 0 | 0 |
| $(1)^{(n)}$ | 1 | 0 |

Rysunek 2.12 Dwa regiony R_i oraz R_j z zaznaczonymi na niebiesko pikselami, które decydują o przyleganiu regionów. Jeżeli rozpatrujemy 4-przyleganie to regiony są rozłączne, a w przypadku 8-przylegania są regionami przylegającymi. Na podstawie [1].

Zakładając, że obraz zawiera zestaw regionów rozłącznych będących interesującymi nas obiektami to ich sumę określamy jako pierwszy plan (ang. *foreground*), a wszystkie pozostałe piksele obrazu określamy jako tło (ang. *background*) [1].

Granica regionu R (zwana również konturem) jest zbiorem pikseli należących do zbioru R które są jednocześnie przylegające do zbioru będącego dopełnieniem zbioru R, a ujmując to krócej granica regionu jest zbiorem pikseli regionu R, które posiadają przynajmniej jednego sąsiada należącego do tła [1]. Do wyznaczenia tak zdefiniowanej granicy regionu należy również przyjąć założenie co do rozpatrywanego typu łączności w celu ustalenia przylegania, gdzie najczęściej wybierana jest 8-łączność [1]. Przy wyborze 4-łączności

piksele, które przylegają do tła tylko diagonalnie nie będą traktowane jako piksele należące do granicy regionu (Rysunek 2.13), gdzie tak zdefiniowana granica regionu może być nazwane granicą wewnętrzną, a analogiczna granica wyznaczona w obszarze tła zwana jest granicą zewnętrzną [1]. W szczególnym przypadku gdy region *R* zajmuje cały obraz to jego granicę definiujemy jak zbiór pikseli leżący w pierwszym i ostatnim wierszu oraz kolumnie obrazu [1].

| 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 |

Rysunek 2.13 Piksel oznaczony kolorem niebieskim w przypadku 8-łączności jest pikselem należącym do granicy regionu, a w przypadku użycia 4-łączności nie jest. Na podstawie [1].

W wielu przypadkach potrzebny jest pomiar odległości pomiędzy pikselami obrazu z wykorzystaniem odpowiedniej metryki. Aby przeprowadzić taki pomiar należy zacząć od zdefiniowania funkcji, która posłuży do jego przeprowadzenia. Funkcja *D* służąca pomiarowi odległości jest funkcją rzeczywistą, która jednocześnie spełnia poniższe warunki dla pikseli p(x, y), q(u, v), s(w, z) [1, 2]:

• identyczność nierozróżnialnych oraz nieujemność:

$$D(p,q) \ge 0 \text{ or } az \ D(p,q) = 0 \iff p = q \tag{2.11}$$

• symetria:

$$D(p,q) = D(q,p) \tag{2.12}$$

nierówność trójkąta:

$$D(p,s) \le D(p,q) + D(q,s)$$
 (2.13)

Istnieje wiele funkcji spełniających powyższe kryteria i mogą one posłużyć do określenia odległości pomiędzy pikselami. Można jednak wyróżnić kilka takich funkcji, które są używane najczęściej. Najpopularniejszą funkcją używaną do określania odległości pikseli jest odległość euklidesowa D_e zdefiniowana jako [1, 2]:

$$D_e(p,q) = \sqrt{(x-u)^2 + (y-v)^2}$$
(2.14)

Dla tak zdefiniowanej metryki piksele które mają odległość mniejszą bądź równą r od punktu (x, y) zawierają się w okręgu o promieniu r ze środkiem znajdującym się w punkcie (x, y) [1, 2]. W celu uogólnienia sposobu wyznaczania odległości euklidesowej na n wymiarów dla punktów, które posiadają koordynaty $A(x_{1A}, x_{2A,...,} x_{nA})$ oraz $B(x_{1B}, x_{2B,...,} x_{nB})$ równanie zapisujemy jako [1, 2]:

$$D_e(A,B) = \sqrt{(x_{1A} - x_{1B})^2 + (x_{2A} - x_{2B})^2 + \dots + (x_{nA} - x_{nB})^2}$$
(2.15)

oraz zapisując w postaci sumy:

$$D_e(A,B) = \sqrt{\sum_{i=1}^{n} (x_{iA} - x_{iB})^2}$$
(2.16)

Kolejną metryką często używaną do określania odległości w kontekście przetwarzania obrazów jest odległość D_4 zwana też metryką taksówki (ang. *taxicab metric*) czy odległością Manhattan (Rysunek 2.14), gdzie zakładając punkty p i q metrykę definiujemy jako [1, 2]:

$$D_4(p,q) = |x - u| + |y - v|$$
(2.17)

W takim przypadku dla piksela D_4 odległość od punktu (x, y), która jest mniejsza bądź równa *d* formuje kształt przypominający diament ze środkiem w punkcie (x, y), a uogólniając dla punktów $A(x_{1A}, x_{2A,...}, x_{nA})$ oraz $B(x_{1B}, x_{2B,...}, x_{nB})$ [1, 2]:

$$D_4(A,B) = \sum_{i=1}^n |x_{iA} - x_{iB}|$$
(2.18)

| 0 | 0 | 2 | 0 | 0 |
|---|---|---|---|---|
| 0 | 2 | 1 | 2 | 0 |
| 2 | 1 | 0 | 1 | 2 |
| 0 | 2 | 1 | 2 | 0 |
| 0 | 0 | 2 | 0 | 0 |

Rysunek 2.14 Zobrazowanie metryki D_4 gdzie cyfry odpowiadają odległościom. Na podstawie [1].

Inną popularną metryką jest odległość D_8 zwana też odległością Czebyszewa lub odległością szachownicy (ang. *chessboard distance*) (Rysunek 2.15) [1, 2]. Definiujemy ją jako:

$$D_8(p,q) = \max(|x-u|, |y-v|)$$
(2.19)

W takim przypadku dla piksela D_8 odległość od punktu (x, y), która jest mniejsza bądź równa *d* formuje kwadrat ze środkiem w punkcie (x, y) [1, 2].

| 2 | 2 | 2 | 2 | 2 |
|---|---|---|---|---|
| 2 | 1 | 1 | 1 | 2 |
| 2 | 1 | 0 | 1 | 2 |
| 2 | 1 | 1 | 1 | 2 |
| 2 | 2 | 2 | 2 | 2 |

Rysunek 2.15 Zobrazowanie metryki D₈ gdzie cyfry odpowiadają odległościom. Na podstawie [1].

2.6 Filtrowanie przestrzenne

Jednym z głównych narzędzi przetwarzania obrazów jest filtrowanie przestrzenne. Ten typ filtrowania polega na modyfikacji obrazu poprzez przypisanie nowej wartości dla każdego piksela obrazu na podstawie funkcji tego piksela oraz pikseli w jego sąsiedztwie [1, 3, 7]. Jeżeli to przekształcenie jest liniowe to taki filtr nazywamy liniowym filtrem przestrzennym, a w przypadku gdy jest nieliniowe – nieliniowym filtrem przestrzennym [1, 2, 3, 7].

W przypadku liniowych filtrów przestrzennych mamy do czynienia z przekształceniem polegającym na sumie wyników operacji pomiędzy jądrem filtra ω , a obrazem f [1, 2]. Jądro (ang. *kernel*) filtra przestrzennego nazywane jest również często maską (ang. *mask*) [1, 4, 7]. Najczęściej wykorzystywaną maską jest maska o rozmiarze 3×3 . Dla każdego piksela obrazu o koordynatach (x, y) po zastosowaniu filtra wyznaczana jest odpowiedź g(x, y), która jest sumą działania współczynników maski na piksele obrazu objęte przez tę maskę (Równanie (2.20)) [1, 2]. Maska przesuwa się po całym obrazie w ten sposób,

że koordynaty centrum maski wyznaczają koordynaty piksela na nowym obrazie wynikowym do którego przypisywany jest wynik operacji [1]. Tworzony jest nowy obraz, ponieważ przypisywanie nowych wartości do obrazu oryginalnego podczas filtracji powodowałoby zmianę obrazu oryginalnego podczas filtrowania więc maska przesuwająca się po obrazie operowałaby również na zmienionych pikselach [1, 2].

$$g(x,y) = \omega(-1,-1)f(x-1,y-1) + \omega(-1,0)f(x-1,y) + \cdots$$

+ $\omega(0,0)f(x,y) + \cdots + \omega(1,1)f(x+1,y+1)$ (2.20)

W trakcie procesu filtrowania maska przesuwana jest od piksela do piksela [1, 2]. Centrum maski $\omega(0,0)$ pokrywa się z przetwarzanym w danej chwili pikselem o koordynatach (x, y)oraz zakładamy maskę posiadającą rozmiar $m \times n$, gdzie dodatkowo spełnione są zależności dla *a* oraz *b* będących nieujemnymi liczbami całkowitymi: m = 2a + 1 oraz n = 2b + 1, więc zakładamy, że maska może przyjmować rozmiary będące liczbami nieparzystymi oraz niemniejsze niż 1 [1]. Dla obrazu o rozmiarach $M \times N$ oraz maski $m \times n$ gdzie *x* oraz *y* zmieniają się tak, że centrum maski znajduje się na pozycji każdego piksela obrazu *f* tylko raz zapisujemy równanie [1, 2]:

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s,t) f(x+s,y+t)$$
(2.21)

Mówiąc o filtrowaniu przestrzennym należy wyjaśnić dwa pojęcia: korelacja przestrzenna (ang. *spatial correlation*) oraz konwolucja przestrzenna (ang. *spatial convolution*). Mechanizm korelacji przestrzennej jak i konwolucji przestrzennej jest identyczny [1]. Polega na przesuwaniu maski tak aby jej środek dokładnie raz znalazł się na pozycji każdego piksela obrazu oraz na obliczeniu w każdej z tych pozycji sumy wyników działania poszczególnych współczynników maski na piksele obrazu odpowiadające kolejnym współczynnikom maski [1]. Korelacja przestrzenna opisana jest równaniem (2.21) oraz przedstawiona na ilustracji (Rysunek 2.16). Różnica pomiędzy korelacją przestrzenną a konwolucją przestrzenną sprowadza się do tego, że maska w przypadku konwolucji przestrzennej jest obrócona o 180° [1, 2]. W przypadku gdy maska używana do konwolucji przestrzennej i korelacji przestrzennej jest symetryczna względem jej centrum to oba filtry przestrzenne dadzą ten sam wynik ponieważ obrócenie maski o 180° nie będzie miało na niego wpływu [1].



Rysunek 2.16 Przykład obrazu po którego wszystkich pikselach przesuwa się maska $\omega(s,t)$ o rozmiarach 3×3 gdzie w powiększeniu widoczne są współrzędne pikseli obrazu znajdujących się pod maską gdy jej środek przypada w punkcie (x, y) oraz współczynniki maski. Na podstawie [1].
Pewien problem pojawia się gdy chcemy obliczyć wynik korelacji bądź konwolucji używając maski o rozmiarze większym niż 1, mianowicie jeżeli chcemy obliczyć którąś z tych wartości dla piksela leżącego w takiej odległości od przynajmniej jednej z krawędzi obrazu, że maska wycentrowana na tym pikselu w części wykracza poza obszar obrazu (Rysunek 2.17) to dla części współczynników maski brakuje odpowiadających im wartości pikseli obrazu do obliczeń [1, 2]. Istnieją dwa najczęściej stosowane rozwiązania. Pierwsze to przesuwanie maski po obrazie tylko w miejsca gdzie w całości znajduje się ona wewnątrz obrazu. Dla maski o rozmiarze $m \times m$ oraz obrazu źródłowego o rozmiarze $M \times N$ gdzie $m \leq M$ oraz $n \leq N$ obraz wynikowy będzie pomniejszony o wyniki dla tych pikseli dla których przyłożona do nich maska wykracza poza obraz. Obraz wynikowy w takim wypadku będzie miał rozmiar $(M - m + 1) \times (N - n + 1)$. Wadą takiego rozwiązania jest to, że obraz wynikowy posiada mniejszy rozmiar niż obraz wyjściowy, a w przypadku zastosowania na nim sekwencyjnie kilku filtrów z których każdy pomniejsza obraz wynikowy różnice te mogą być znaczne. Powoduje to utratę informacji z obrazu. Innym sposobem nie powodującym zmniejszania rozmiaru obrazu wynikowego, a więc i utraty informacji jest powiększenie obrazu poprzez dodanie odpowiedniej ilości wierszy i kolumn z pikselami o wartości zero dookoła obrazu. Tak zakładając należy dodać $\frac{m-1}{2}$ wierszy oraz $\frac{n-1}{2}$ kolumn z każdej strony obrazu tworząc obraz o wymiarach (M + m-1 × (N + n - 1) (Rysunek 2.18) [1, 2].



Rysunek 2.17 Maska o rozmiarze 3 × 3 zastosowana dla pikseli skrajnych w części wykracza poza obraz co powoduje, że wynik działania maski na obrazie jest w tych miejscach nieokreślony.

| 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 |

Rysunek 2.18 Obraz o rozmiarze 3×3 do którego planuje się zastosować maskę 3×3 powiększony poprzez dodanie wierszy i kolumn wypełnionych zerami z każdej strony (niebieski kolor).

Dla przypadku założenia obrazu z dyskretnym impulsem jednostkowym w centrum obrazu możemy pokazać, że wyznaczenie korelacji przestrzennej takiego obrazu z użyciem maski ω daje jako wynik kopię maski ω lecz obróconą o 180° i z centrum znajdującym się w miejscu wystąpienia impulsu na obrazie [1]. Dla analogicznego przypadku różniącego się zastosowaniem konwolucji przestrzennej zamiast korelacji przestrzennej różnicą jest wykonanie rotacji maski ω przed wyznaczeniem konwolucji co prowadzi do wyniku w postaci dokładnej kopii maski ω z centrum znajdującym się w miejscu wystąpienia impulsu (Rysunek 2.19) [1, 2].

| Ory | gina | ılny | obr | az f | |] | Pos | zerz | ony | obr | az f | | | | | | |
|-----|------|------|-----|------|-----|-----|------|------|-----|------|------|-------|---|------|------|------|------|
| | | | | | (|) | 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 | 0 | 0 | (|) | 0 | 0 | 1 | 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 | | | | | |
| | | (a) | | | | | | | (b) | | | | | | | | |
| | M | aska | ω | | Рос | cza | ątko | owa | poz | ycja | mas | ski ω | W | ynil | c ko | rela | acji |
| | | | | | 1 | L | 2 | 3 | 0 | 0 | 0 | 0 | | | | | |
| | | | | | | | E | 6 | Ο | Ο | 0 | 0 | 0 | Ο | 0 | Ο | 0 |

| | 1 | - 4 | Э | U | U | U | U | | | | | | |
|-------|---|-----|---|-----|---|---|---|--|---|---|-----|---|---|
| | 4 | 5 | 6 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 |
| 1 2 3 | 7 | 8 | 9 | 0 | 0 | 0 | 0 | | 0 | 9 | 8 | 7 | 0 |
| 4 5 6 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | | 0 | 6 | 5 | 4 | 0 |
| 789 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 3 | 2 | 1 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 0 | 0 | 0 |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | | | | |
| | | | | | | | | | | | | | |
| (c) | | | | (d) | | | | | | | (e) | | |
| | | | | | | | | | | | | | |

| Maska ω obrócona o 180° | | Początkowa pozycja obróconej | | | | | | | Wynik konwolucji | | | | | |
|--------------------------------|-----|------------------------------|---|---|---|------|---|---|------------------|---|---|-----|---|---|
| | | | | | m | aski | ω | | | | | | | |
| | | | 9 | 8 | 7 | 0 | 0 | 0 | 0 | | | | | |
| | | | 6 | 5 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 9 | 8 | 7 | 3 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 3 | 0 |
| 6 | 5 | 4 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 4 | 5 | 6 | 0 |
| 3 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 8 | 9 | 0 |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | | | |
| | (f) | | | | | (g) | | | | | | (h) | | |

Rysunek 2.19 Korelacja (c, d, e) oraz konwolucja (f, g, h) obrazu f z wykorzystaniem dwuwymiarowej maski ω oraz jej obróconej o 180° wersji. Na podstawie [1].

Ostatecznie korelacje przestrzenną $(\omega \not\approx f)(x, y)$ maski ω o rozmiarze $m \times n$ oraz obrazu f(x, y) zapisujemy równaniem:

$$(\omega \not\approx f)(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s, t) f(x + s, y + t)$$
(2.22)

gdzie $a = \frac{m-1}{2}$, $b = \frac{n-1}{2}$, a *m* oraz *n* są parzystymi nieujemnymi liczbami całkowitymi [1].

Ostatecznie konwolucje przestrzenną $(\omega \star f)(x, y)$ maski ω o rozmiarze $m \times n$ oraz obrazu f(x, y) zapisujemy równaniem:

$$(\omega \star f)(x, y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s, t) f(x - s, y - t)$$
(2.23)

gdzie znaki odejmowania wyrównują koordynaty obrazu f oraz obróconej o 180° maski ω oraz $a = \frac{m-1}{2}$, $b = \frac{n-1}{2}$, a m oraz n są parzystymi nieujemnymi liczbami całkowitymi [1].

Tabela 2.2 Fundamentalne własności korelacji oraz konwolucji. Na podstawie [1, 5].

| Własność | Konwolucja | Korelacja |
|------------------------|---|---|
| Przemienność | $f \star g = g \star f$ | - |
| Łączność | $f \star (\mathbf{g} \star \mathbf{h}) = (f \star \mathbf{g}) \star \mathbf{h}$ | - |
| Rozdzielność działania | $f + (g \star h) = (f \star g) + (f \star h)$ | $f + (g \stackrel{\text{tr}}{\Rightarrow} h) = (f \stackrel{\text{tr}}{\Rightarrow} g) + (f \stackrel{\text{tr}}{\Rightarrow} h)$ |

Najprostszym przykładem maski działającej jak filtr dolnoprzepustowy jest maska zwana jądrem pudełkowym (ang. *box kernel*) (Rysunek 2.20) [1, 7]. Konwolucja z użyciem takiego filtra powoduje uśrednienie wartości obrazu. Im większy będzie rozmiar maski tym obszar uśredniania będzie większy więc obraz będzie bardziej rozmyty.

| 1 | [1 | 1 | [1 |
|---|----|---|----|
| - | 1 | 1 | 1 |
| 9 | 1 | 1 | 1 |

Rysunek 2.20 Jądro pudełkowe wraz ze współczynnikiem normalizującym.

Inną maską bardzo często używaną w celu rozmycia obrazu (usunięcia szczegółów) jest maska oparta na próbkowaniu dwuwymiarowej funkcji Gaussa (Rysunek 2.21, Rysunek 2.24), gdzie wartości współczynników opisane są wzorem (2.24), dla K = 1 oraz $\sigma = 1$ [1]:

$$\omega(s,t) = G(s,t) = Ke^{-\frac{s^2 + t^2}{2\sigma^2}}$$
(2.24)

| 1 | [0.3679 | 0.6065 | 0.3679] |
|--------|---------|--------|---------|
| 1007(| 0.6065 | 1.0000 | 0.6065 |
| 4.89/6 | L0.3679 | 0.6065 | 0.3679 |

Rysunek 2.21 Jądro oparte na funkcji Gaussa wraz ze współczynnikiem normalizującym. Na podstawie [1].

Wartości współczynników w takim przypadku maleją wraz ze wzrostem odległości od centrum maski. Powoduje to, że piksele z sąsiedztwa rozpatrywanego w danej chwili piksela mają malejący wraz z odległością (Rysunek 2.23) wpływ na ostateczną wartość

odpowiadającą temu pikselowi po filtrowaniu. Można powiedzieć, że piksele leżące bliżej rozpatrywanego piksela są ważniejsze niż te bardziej oddalone.

W celu wyostrzenia obrazu czyli wydobycia detali stosuje się często maski wyostrzające. Uwidaczniają one detale obrazu lecz mogą również zwiększać widoczność szumu. Przykładem takiej maski jest:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 12 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Rysunek 2.22 Przykładowa maska wyostrzająca.



Rysunek 2.23 Euklidesowe odległości od centrum dla kwadratowych jąder (masek) różnych rozmiarów *m*. Na podstawie [1].



Rysunek 2.24 Dwuwymiarowa funkcja Gaussa użyta w celu stworzenia maski do konwolucji przestrzennej.

Przykładem filtrów nieliniowych są filtry oparte na statystyce kolejności (ang. *order-statistic filters*) [1, 7]. Działanie tego typu filtrów opiera się na ułożeniu pikseli sąsiedztwa odpowiadającego obszarowi działania filtra w określonej przez zdefiniowane reguły kolejności [1, 7]. Jest to stworzenie swoistego rankingu tych pikseli. Wartość wynikowa dla rozpatrywanego piksela otrzymywana jest poprzez przypisanie mu wartości piksela wybranego z utworzonego rankingu [1, 7]. Najczęściej wykorzystywanym filtrem tego typu jest filtr medianowy, którego wynikiem dla rozpatrywanego piksela jest mediana wartości pikseli z rozpatrywanego obszaru (sąsiedztwa) tego piksela [1, 7]. Można również wybrać przykładowo wartość minimalną bądź maksymalną otrzymując odpowiednio filtr minimalny oraz filtr maksymalny [7]. Sam rozpatrywany piksel również brany jest pod uwagę. Filtr medianowy świetnie sprawdza się w zadaniu usuwania szumu impulsowego zwanego czasami szumem sól i pieprz (ang. *salt-and-pepper noise*) [1, 7]. Szum taki składa się z pojedynczych czarnych lub białych pikseli na obrazie [7]. Filtry bazujące na uśrednianiu wartości w obszarze maski filtrowania gorzej radzą sobie z takim zadaniami ponieważ taki impulsowy szum znacznie zaburza średnią [1, 2].

3 Metody filtracji wstępnej obrazów

3.1 Wprowadzenie

Niniejsza rozprawa kładzie nacisk na zagadnienie segmentacji obrazów jako jedno z ważniejszych zagadnień w dziedzinie analizy i przetwarzania obrazów. Poprawna segmentacja ułatwia również zadania związane z klasyfikacją obiektów na obrazach. Obrazy dla których wykonuje się zadania segmentacji i klasyfikacji obiektów często zawierają zaszumienie czy różnego rodzaju artefakty wynikające np. z metody użytej do stworzenia obrazu lub wykorzystanego sprzętu. Usunięcie takiego szumu zaburzającego oryginalną informację zawartą w obrazie pozwala polepszyć wyniki otrzymywane w zadaniach segmentacji i klasyfikacji. W tej dziedzinie na szeroką skalę używane są algorytmy klasyczne jak np. filtry lecz ostatnimi laty coraz szerzej wykorzystywane jest głębokie uczenie. W kolejnych podrozdziałach przedstawiono metody usuwania szumów, odejmowania tła czy wyrównania histogramu wykorzystywane w algorytmach opisanych w tej rozprawie.

3.2 Metody klasyczne

3.2.1 Filtr medianowy

Filtr medianowy jest często używanym przedstawicielem nieliniowych metod przetwarzania obrazów [7]. Algorytm opiera się na iteracji po wszystkich pikselach obrazu w celu obliczenia mediany z wartości pikseli w jego sąsiedztwie. W wynikowym przefiltrowanym obrazie wartość rozważanego piksela zastępowana jest przez tak wyznaczoną medianę. Jeżeli założymy, że punktowe odstępstwa od normy w postaci jasnych lub ciemnych pikseli są szumem na badanym obrazie to filtr medianowy jest bardzo dobry w usuwaniu tego typu szumu [2, 7, 9, 10]. Dla przykładu przyjmując rozmiar maski filtra jako 3x3 bierzemy pod uwagę 9 pikseli. Zakładając wartości pikseli 60, 60, 60, 60, 60, 60, 60, 60, 240 po zastosowaniu filtra medianowego nowa wartość piksela wyniesie 60 [5]. Punktowy szum zostaje w ten sposób usunięty i nie ma wpływu na nową wartość piksela. Gdyby dla takich przykładowych wartości użyto np. filtra

uśredniającego wartość piksela po przefiltrowaniu wyniosłaby 80. Szum zostałby usunięty częściowo. Szum punktowy wpłynąłby na wartość po przefiltrowaniu.

$$h(x, y) = \underset{(s,t)\in S_{xy}}{\operatorname{mediana}} \{g(s, t)\}$$
(3.1)

gdzie:

- h(x, y) mediana obliczona dla wartości z obszaru s * t oryginalnego obrazu
- g(s, t) obszar oryginalnego obrazu z centrum w punkcie (x, y)
- S_{xy} Zbiór współrzędnych dla maski o rozmiarze m * n

3.2.2 Filtr uśredniający

Filtr uśredniający jest jednym z przedstawicieli liniowych metod przetwarzania obrazów [7]. Po filtracji nowa wartość dla każdego rozważanego piksela jest średnią wartością wszystkich sąsiadujących pikseli należących do zadanego sąsiedztwa. Taki rodzaj filtracji powoduje równomierne rozmycie obrazu [1, 2, 7].

$$h(x,y) = \frac{1}{m*n} \sum_{(s,t) \in S_{xy}} g(s,t)$$
(3.2)

gdzie:

- h(x, y) średnia obliczona dla wartości z obszaru s * t oryginalnego obrazu
- g(s,t) obszar oryginalnego obrazu z centrum w punkcie (x, y)
- S_{xy} Zbiór współrzędnych dla maski o rozmiarze m * n

3.2.3 Filtr bilateralny

Filtr bilateralny jest techniką która ma zastosowanie gdy celem jest usunięcie niechcianych detali z obrazu jak tekstury czy szum przy jednoczesnym zachowaniem ostrych krawędzi. Czyni go to dobrym wyborem gdy np. w kolejnym kroku po odszumianiu będzie wykonane wykrywanie krawędzi Ta cecha jest dużą przewagą tego filtra nad np. rozmyciem gaussowskim. Metoda wykorzystuje konwolucje z użyciem jądra gaussowskiego wraz z ważoną średnią wartości pikseli z zadanego sąsiedztwa (równanie (3.3)). Ten algorytm rozszerza działanie rozmycia gaussowskiego ponieważ oprócz ważonej zależności przestrzennej bierze również pod uwagę odległość pomiędzy wartościami pikseli w dziedzinie intensywności [11, 12]. W rozmyciu gaussowskim do obliczenia nowej wartości piksela wykorzystana jest ważona średnia wartości intensywności pikseli z zadanego sąsiedztwa. Wagi zmniejszają się wraz ze wzrostem dystansu przestrzennego rozpatrywanego piksela z sąsiedztwa od centralnego piksela. Bardziej odległe sąsiednie piksele mają mniejsze znaczenie podczas obliczania nowej wartości dla zadanego piksela. Ta zależność wyraża się równaniem [1, 11]:

$$G[I]_{p} = \frac{1}{W_{pG}} \sum_{q \in S} G_{\sigma}(\|p - q\|) I_{q}$$
(3.3)

gdzie $G_{\sigma}(x)$ jest gaussowskim jądrem konwolucji wyrażonym jak przedstawiono w równaniu [1, 11]:

$$G_{\sigma}(x) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2}{2\sigma^2}\right)$$
(3.4)

gdzie: S jest dziedziną przestrzenną, W_{pG} – sumą wszystkich wag, I – intensywnością piksela, ||p - q|| – odległością euklidesową pomiędzy rozważanym centralnym pikselem

p a innym pikselem *q* zawartym w zadanym sąsiedztwie piksela centralnego. Profil zmiany wag zależny od odległości przestrzennej wyrażony jest przy użyciu σ . Większe wartości tego parametru powodują większe rozmycie obrazu wynikowego powodując zbliżenie działania filtra do filtra uśredniającego gdzie wszystkie wagi mają równą wartość. Główną wadą filtra gaussowskiego jest rozmywanie krawędzi. Filtr bilateralny odpowiada na ten problem. Wyrażony jest równaniem [11]:

$$B[I]_{p} = \frac{1}{W_{pB}} \sum_{q \in S} G_{\sigma_{s}}(\|p - q\|) G_{\sigma_{r}}(\|I_{p} - I_{q}\|) I_{q}$$
(3.5)

gdzie czynnik normalizacyjny wyrażony jest jako [11]:

$$W_{pB} = \sum_{q \in S} G_{\sigma_{S}}(\|p - q\|) G_{\sigma_{r}}(\|I_{p} - I_{q}\|)$$
(3.6)

Tylko piksele, które są blisko piksela centralnego w przestrzeni i dziedzinie intensywności są brane pod uwagę. Dla dziedziny przestrzennej jądro gaussowskie wyrażone jest przez G_{σ_s} . Dla dziedziny intensywności jądro gaussowskie jest wyrażone jako G_{σ_r} . Wagi maleją wraz ze wzrostem odległości w dziedzinie intensywności czyli wraz ze wzrostem różnicy wartości poszczególnych pikseli. Tak ujęte jednoczesne filtrowanie w dziedzinie przestrzennej i intensywności pozwala filtrowi bilateralnemu wygładzanie obrazu przy jednoczesnym zachowaniu krawędzi [13, 14]. Powoduje to również, że filtr ten nie usunie np. szumu typu sól i pieprz z którym świetnie poradzi sobie z kolei filtr medianowy. Pokazuje to, że należy uważnie dobrać algorytmy filtracji do dziedziny problemu, który chcemy rozwiązać.

3.2.4 Odejmowanie tła

Odejmowanie tła jest metodą stosowaną w przypadku obrazów z niejednorodnym profilem tła. Umożliwia ona uzyskanie równomiernego rozkładu wartości tła dla całego obrazu. Pozwala to wyeliminować przesunięcia intensywności pikseli w różnych obszarach obrazu powstałe np. w wyniku zastosowanej techniki akwizycji obrazu. Przykładem mogą być obrazy pochodzące z mikroskopu optycznego gdzie niejednorodne podświetlenie preparatu powoduje większe wartości pikseli w okolicach centrum obrazu (Rysunek 3.1).



Rysunek 3.1 Obraz erytrocytów uzyskany przy pomocy mikroskopu optycznego w skali szarości oraz po przekształceniu typu WW/WL (ang. *Window width/Window level*) zastosowanym w celu uwypuklenia niejednorodności tła (a), wizualizacja tego samego obrazu w postaci wykresu 3d (b).

Przykładem takiego algorytmu jest algorytm toczącej się kuli (ang. *Rolling ball algorithm*). Wykorzystuje on sferyczny (lub o innym kształcie) element strukturalny, który przesuwa się po powierzchni otrzymanej poprzez potraktowanie obrazu jako trójwymiarowego wykresu gdzie rozmiary obrazu oraz wartości pikseli tworzą osie [1, 15]. Podczas przesuwania elementu strukturalnego nie może on znaleźć się poniżej tak utworzonej powierzchni [1, 15]. Taki mechanizm pozwala zdefiniować profil tła poprzez przesunięcie

elementu strukturalnego na pozycję wyznaczoną przez położenie każdego piksela obrazu [1, 15]. Tak uzyskany profil tła może być następnie odjęty od obrazu. Rozmiar elementu strukturalnego dobiera się tak aby był odpowiednio większy od szczytów i zagłębień odpowiadających obiektom na obrazie aby uzyskać profil tła niezaburzony przez obiekty znajdujące się na obrazie [1, 15].

3.2.5 Limitowane kontrastem adaptacyjne wyrównanie histogramu (CLAHE)

Algorytm limitowanego kontrastem adaptacyjnego wyrównania histogramu (ang. *contrast limited adaptive histogram equalization* - CLAHE) ma na celu wyrównanie histogramu (2.4.1). Jest to algorytm adaptacyjny ponieważ wyrównanie histogramu odbywa się na mniejszych fragmentach obrazu zamiast na całym jednocześnie [1, 2].

Standardowe wyrównanie histogramu polega na przekształceniu wszystkich pikseli obrazu przy pomocy funkcji uzyskanej na podstawie histogramu wyznaczonego dla wszystkich pikseli obrazu (2.4.1). Powoduje to, że jeżeli obraz nie jest jednorodny tzn. występują obszary jaśniejsze i ciemniejsze to występują miejsca gdzie algorytm nie zadziała w sposób dający oczekiwane wyniki. Ulepszeniem tego podejścia jest zastosowanie adaptacyjnego wyrównania histogramu (ang. adaptive histogram equalization - AHE), który polega na obliczeniu funkcji transformacji wartości pikseli obrazu na podstawie histogramu wyznaczonego dla obszarów będących mniejszymi fragmentami obrazu [16]. Powoduje to lepsze dopasowanie tej transformacji do lokalnego rozkładu intensywności pikseli. Algorytm daje wtedy dobre wyniki również przy występowaniu lokalnych jaśniejszych i ciemniejszych obszarów. Wadą tego rozwiązania jest to, że w przypadku występowania jednorodnych obszarów algorytm może znacznie wzmocnić szum [16]. Rozwiązaniem tego problemu jest algorytm limitowanego kontrastem adaptacyjnego wyrównania histogramu (CLAHE) [16, 17]. Wprowadza on limit kontrastu do oczekiwanego poziomu poprzez ograniczenie maksymalnego nachylenia funkcji z równania (2.10) przy pomocy wprowadzenia odpowiedniego współczynnika β [17]:

$$\beta = \frac{M}{N} \left(1 + \frac{\alpha}{100} (s_{max} - 1) \right) \tag{3.7}$$

gdzie:

- N ilość możliwych poziomów intensywności
- M ilość pikseli w rozpatrywanym regionie obrazu
- α współczynnik ograniczania o wartościach wyrażonych w procentach od 0 do 100
- s_{max} maksymalne dopuszczalne nachylenie

4 Metody Segmentacji obrazów

4.1 Wprowadzenie

W niniejszym rozdziale przedstawiono metody segmentacji obrazów czyli wyodrębnienia regionów będących obiektami z obrazu. Przedstawione są metody bezpośrednio wykonujące segmentację (np. algorytm Otsu) jak i takie biorące w niej udział pośrednio jako część procesu (np. Dylatacja).

4.2 Przekształcenia morfologiczne

4.2.1 Mapa odległości

Mapa odległości jest przekształceniem polegającym na obliczeniu odległości euklidesowej do tła dla każdego piksela należącego do obiektów na obrazie [1, 2]. Wynik można zaprezentować jako białe tło i ciemne obiekty gdzie kolor poszczególnych pikseli należących do tych obiektów staje się jaśniejszy wraz ze zbliżaniem się do tła i analogicznie ciemniejszy wraz ze wzrastającą odległością do pikseli należących do obszaru tła (Rysunek 4.1).



Rysunek 4.1 Przykładowy obraz binarny erytrocytów po segmentacji (a), oraz otrzymana na jego podstawie mapa odległości (b).

4.2.2 Erozja

Erozja jest operacją wykonywaną na obrazie binarnym z wykorzystaniem wybranego elementu strukturalnego jak np. kwadratowa lub radialna maska [1, 2, 5]. Algorytm polega na przeprowadzeniu testu dla każdego piksela czy wybrany element strukturalny umieszczony centralnie w obecnie przetwarzanym punkcie zawiera się w zbiorze punktów należących do obiektu (równanie (4.1)) [1, 2, 5]. W przypadku spełnienia tego warunku rozważany piksel jest traktowany w obrazie wynikowym jako piksel obiektu [1, 2, 5]. Równanie (4.1) wyraża matematyczne przedstawienie operacji erozji [1, 4].

$$A \ominus B = \{ z | (B)_z \subseteq A \}$$

$$(4.1)$$

gdzie:

- *A* zbiór pikseli pierwszoplanowych (należących do obiektów)
- *B* element strukturalny (przykładowo maska 3x3)
- $(B)_z B$ po translacji o z

4.2.3 Dylatacja

Dylatacja jest operacją wykonywaną na obrazie binarnym z wykorzystaniem wybranego elementu strukturalnego w podobny sposób jak erozja [1, 2, 5]. Algorytm polega na przeprowadzeniu testu dla każdego piksela czy element strukturalny umieszczony centralnie w obecnie przetwarzanym punkcie posiada jakąkolwiek część wspólną ze zbiorem punktów należących do obiektów [1, 2, 5]. Jeżeli powyższy warunek jest spełniony obecnie rozważany piksel jest traktowany w obrazie wynikowym jako piksel należący do zbioru pikseli obiektów [1, 2, 5]. Równanie (4.2) wyraża matematyczną reprezentację operacji dylatacji [1, 4].

$$A \bigoplus B = \left\{ z | (\hat{B})_z \cap A \neq \emptyset \right\}$$

$$(4.2)$$

gdzie:

- *A* zbiór pikseli pierwszoplanowych (należących do obiektów)
- *B* element strukturalny (przykładowo maska 3x3)
- $(\hat{B})_z$ odbite *B* po translacji o *z*

4.3 Metody bazujące na intensywności pikseli

4.3.1 Algorytm Otsu

Segmentacja przy użyciu algorytmu Otsu jest szybką metodą segmentacji dającą dobre wyniki dla obrazów gdzie obiekty są dobrze oddzielone od tła. Duża szybkość działania algorytmu jest zasługą użycia histogramu do obliczenia progów (patrz rozdział 2.4). Algorytm zaczyna działanie od obliczenia histogramu, a następnie znajduje progi z użyciem kryterium minimalizacji ważonej wariancji w otrzymanych klasach co jest równoważne do maksymalizacji wariancji pomiędzy klasami [1, 7, 18, 19]. Obliczenia mogą być przeprowadzone dla jednego lub większej ilości progów [18]. Segmentacja obrazu uzyskana jest poprzez wykorzystanie tak wyznaczonych progów do progowania obrazu oryginalnego [18, 19].

4.3.2 Progowanie z histerezą

Progowania z histerezą wykorzystuje dwa progi jako parametry T_1 , T_2 gdzie $T_1 < T_2$. Zastosowanie dwóch progów pozwala podzielić piksele na trzy grupy [14, 20, 21]:

 Piksele z intensywnością poniżej dolnego progu (T₁) – ta grupa pikseli jest klasyfikowana jako piksele nie należące do obiektu co prowadzi do ich usunięcia [14, 20, 21].

- Piksele z intensywnością powyżej górnego progu (T₂) ta grupa pikseli klasyfikowana jest jako piksele należące do obiektu co skutkuje ich pozostawieniem [14, 20, 21].
- Piksele z intensywnością pomiędzy progami T_1 i T_2 ta grupa pikseli klasyfikowana jest jako piksele należące do obiektu tylko jeżeli dany piksel jest umiejscowiony przy pikselu należącym do drugiej grupy (z intensywnością powyżej górnego progu T_2). Spełnienie tych warunków prowadzi do pozostawienia piksela [14, 20, 21].

4.3.3 Algorytm wododziałowy

Algorytm segmentacji wododziałowej traktuje dwuwymiarowy monochromatyczny obraz jako trójwymiarową funkcję, którą można zwizualizować jako mapę gdzie trzecim wymiarem jest intensywność pikseli [1, 2, 22]. Wyższe wartości intensywności odpowiadają grzbietom górskim, a mniejsze dolinom [1, 2]. Algorytm przeprowadza operację wzorowaną na zalewaniu takiej mapy przez ciecz np. wodę, gdzie wyimaginowany poziom wody podnosi się zalewając doliny powstałe wokół lokalnych minimów zwane zlewnią (ang. *catchment basins*) [1, 2]. Postępuje to iteracyjne. Gdy w następnym kroku zatapiania zlewnia przelałaby się powodując połączenie z inną zlewnią wyimaginowana tama powstaje na otaczających ją grzbietach w celu zapobiegnięcia łączeniu się basenów, a sam proces jest parametryzowalny co pozwala sterować poziomem segmentacji [1, 2].

4.3.4 Statystyczne łączenie obszarów (ang. Statistical Region Merging - SRM)

W metodach bazujących na łączeniu obszarów, obszary definiowane są jako zbiór pikseli posiadających jednorodne własności. Podczas iteracyjnego procesu obszary te są iteracyjnie

powiększane poprzez łączenie mniejszych obszarów. Elementarnymi obszarami są piksele. Szczegółowy opis metody znajduje się w oryginalnym artykule [23].

4.4 Metody bazujące na wykrywaniu krawędzi

4.4.1 Detektor krawędzi Canny–Deriche

Aby skutecznie rozwiązać zadany problem dobrze jest go najpierw zdefiniować. Canny w swojej publikacji w celu opisania algorytmu detekcji krawędzi zaproponował trzy ważne według niego kryteria cechujące wynik procesu efektywnej detekcji krawędzi [20]:

- dobra detekcja (ang. *Good detection*) kryterium to opisuje małe prawdopodobieństwo niewykrycia istniejącej krawędzi oraz małe prawdopodobieństwo wykrycia nieistniejącej krawędzi [20]
- dobra lokalizacja (ang. *Good localization*) kryterium to opisuje sytuację gdzie wykryte krawędzie powinny znajdować się możliwie najbliżej tych rzeczywistych [20]
- jedna odpowiedź dla jednej krawędzi (ang. One response to one edge) kryterium to oznacza, że nie powinny występować wielokrotne odpowiedzi dla pojedynczej rzeczywistej krawędzi

Następnie Canny wykorzystał opisane powyżej kryteria i połączył je w celu stworzenia jednego optymalnego operatora, który jest w przybliżeniu pierwszą pochodną funkcji Gaussa [20, 21]:

$$f(x) = -\frac{x}{\sigma^2} e^{-\frac{x^2}{2}\sigma^2}$$
(4.3)

Deriche wykorzystał operator stworzony przez Cannego lecz zmodyfikował jego podejście w celu poprawienia rezultatów i stworzenia bardziej optymalnego rozwiązania będącego detektorem krawędzi [21]. Tak otrzymany ulepszony detektor krawędzi przedstawił w postaci równania [21]:

$$f(x) = k \cdot e^{-\alpha |x|} \sin \omega x \tag{4.4}$$

Rozpatrując szczególny przypadek gdy ω dąży do 0 powyższe równanie można zapisać jako [21]:

$$g(x) = k \cdot x e^{-\alpha |x|} \tag{4.5}$$

Podejście zaproponowane przez Derichego pozwoliło uzyskać lepsze wyniki niż oryginalna wersja zaproponowana przez Cannego [20, 21]. Metoda składa się z kilku etapów. Jako pierwsze wykonywana jest detekcja wielkości i kierunku gradientu w celu stworzenia mapy gradientu, gdzie większe wartości gradientu pojawiają się w pobliżu krawędzi obiektów, ponieważ tam zmiany intensywności sąsiednich pikseli są największe [21]. Kolejnym etapem jest przeprowadzenie procesu tłumienia niemaksymalnych wartości gradientu (ang. *non-maximal suppression*) co pozwala otrzymać ścienioną krawędź poprzez wybranie pojedynczego najjaśniejszego piksela (największa wartość gradientu) w poprzek szerokości krawędzi [21]. Metody ścieniania krawędzi, do których należy tłumienie niemaksymalnych wartości gradientu to cała oddzielna kategoria algorytmów [3, 24]. Ostatni etap obejmuje progowanie z histerezą (patrz rozdział 4.3.2) stosowane w celu uzyskania ostatecznego binarnego obrazu reprezentującego wyniki detekcji krawędzi [14, 21].

4.5 Metody bazujące na modelach

4.5.1 Modele deformowalne - Aktywne kontury (ang. Active Contours, Snakes)

Model oparty na aktywnych konturach (ang. *Active contours*) [25, 26, 27] zwany wężem (ang. *Snake*) może być opisany jako krzywa parametryczna [27]:

$$x(s) = [x(s), y(s)], \quad s \in [0,1]$$
(4.6)

W trakcie iteracyjnego działania algorytmu tak zdefiniowany wąż porusza się w domenie przestrzennej obrazu, a jego ruchem steruje dążenie aby minimalizować jego energię wyrażoną wzorem [27]:

$$E = \int_{0}^{1} \frac{1}{2} [\alpha |x'(s)|^{2} + \beta |x''(s)|^{2}] + E_{ext}(x(s))ds$$
(4.7)

gdzie:

 α – parametr kontrolujący napięcie (pierwsza pochodna)

 β – parametr kontrolujący sztywność (druga pochodna)

 E_{ext} – siła zewnętrzna wyznaczona na podstawie mapy gradientów zadanego obrazu która przyjmuje mniejsze wartości w pobliżu krawędzi (patrz rozdział 4.5.2).

4.5.2 GVF (ang. Gradient Vector Flow)

W podejściu prezentowanym w rozdziale 4.5.1 wykorzystano metodę GVF (ang. *gradient vector flow*), która polega na dyfuzji wektorów gradientu [27]. Metoda ta stosowana może być przykładowo do obrazów w skali szarości lub binarnej mapy krawędzi. Pola GVF są polami wektorowymi wyznaczonymi na podstawie obrazów poprzez minimalizację funkcjonału energii [27]. Ten etap przeprowadzony jest przy pomocy rozwiązania pary niezależnych liniowych cząstkowych równań różniczkowych [27]. Model aktywnych konturów wykorzystujący pole GVF jako zewnętrzną siłę często jest nazwany wężem GVF (ang. *GVF snake*) [27]. Szczegółowy opis oraz numeryczna implementacja GVF znajduje się w oryginalnym artykule [27].

4.6 Metody bazujące na sztucznych sieciach neuronowych

4.6.1 Podstawy teoretyczne

Algorytmy bazujące na odgórnie zaprogramowanych regułach mają trudności z rozwiązywaniem zadań, które są skomplikowane. Twórca algorytmu musi zauważyć i odpowiednio sformułować reguły pozwalające poprawnie zadziałać algorytmowi. Dobrym przykładem jest algorytm progowania. Dla prostego zadania gdzie od algorytmu wymaga się znalezienia pikseli odpowiadających ciemnym obiektom na jasnym tle, gdzie tło jest bliskie jednorodnemu, a piksele obiektów mają podobne wartości łatwo wybrać wartość progu pozwalającą rozdzielić obiekty od tła. W miarę wzrastania poziomu skomplikowania problemu trzeba wprowadzać coraz więcej reguł i zwiększać stopień ich skomplikowania. Gdy powyżej przytoczony przykład zostanie zmodyfikowany tak, że tło jest niejednorodne i np. obiekty z jednego fragmentu obrazu są jaśniejsze niż tło w innym fragmencie to wybranie jednego progu w celu segmentacji obiektów będzie niemożliwe. np. wprowadzić bardziej skomplikowane reguły w postaci progowania Można adaptacyjnego (ang. adaptive thresholding) i w ten sposób rozwiązać bardziej skomplikowane zadanie lecz łatwo wyobrazić sobie kolejne utrudnienia wymagające wzrastania poziomu skomplikowania algorytmu, aż osiągnięty zostanie poziom skomplikowania problemu w praktyce uniemożliwiający człowiekowi dodawanie kolejnych reguł w celu pełnego rozwiązania zadania. Takim skomplikowanym zadaniem może być np. segmentacja obiektów na skomplikowanych obrazach. Tego typu podejście zwane jest bazującym na wiedzy (ang. knowledge base) [28].

Kolejną grupą algorytmów są algorytmy oparte na uczeniu maszynowym (ang. *machine learning*), które potrafią zdobywać własną wiedzę znajdując wzorce korzystając z surowych danych np. naiwny klasyfikator bayesowski (ang. *naive Bayes classifier*) czy regresja logistyczna (ang. *logistic regression*) [28]. Takie podejście pozwala przezwyciężyć problemy opisane w poprzednim akapicie, które napotykają algorytmy bazujące na wbudowanej wiedzy. Problemem tego typu algorytmów jest zależność ich skuteczności w rozwiązywaniu zadanego problemu od sposobu reprezentacji danych którymi są zasilane [28]. Każdy fragment informacji zawartej w reprezentacji danych zasilających system nazywamy cechą (ang. *feature*) [28]. Przykładowo (autorski przykład

55

bazujący na przykładzie opisanym w [28]) system sztucznej inteligencji bazujący na regresji logistycznej mający za zadanie odpowiedzieć czy pacjent ma jakiś typ choroby nowotworowej bazujący na wynikach badań morfologicznych krwi czy innych skategoryzowanych informacjach pochodzących od lekarzy będzie mieć istotnie lepsze rezultaty niż gdyby podobny system bazował na wartościach indywidualnych pikseli pochodzących z tomografii komputerowej. Algorytm regresji logistycznej uczy się jak poszczególne cechy korelują z wynikiem [28]. Pojedyncze piksele pochodzące z badania tomografii komputerowej posiadają pomijalną korelację z wynikiem, więc algorytm zasilony takimi danymi nie może dać dobrego wyniku [28]. Przeprowadzenie segmentacji obrazów pochodzących z tomografii komputerowej i odnalezienie np. guzów które mogą być wynikiem nowotworu mogłoby stanowić dobrze skorelowaną cechę. Problemem jest to, że czasami takie przetworzenie danych może być operacją bardziej skomplikowaną niż rozwiązanie oryginalnego problemu [28].

W wielu przypadkach prosty algorytm uczenia maszynowego zasilony dobrze dobranymi cechami może być wystarczający do uzyskania dobrych wyników [28]. Niestety często dobór odpowiednich cech jest bardzo skomplikowany i czasochłonny [28]. Jednym z rozwiązań jest uczenie reprezentacji (ang. *representation learning*) polegające na wykorzystaniu uczenia maszynowego nie tylko do przetworzenia reprezentacji danych na wynik ale również do jej wybrania [28]. Przykładem takiego podejścia jest autoenkoder (ang. *autoencoder*) [28]. Jest to kombinacja funkcji kodującej przetwarzającej dane wejściowe do innej ich reprezentacji oraz funkcji dekodującej przetwarzającej tę reprezentację z powrotem do oryginalnego przedstawienia [28]. Autoenkoder uczony jest tak by zachować jak najwięcej informacji po operacji kodowania i następnie dekodowania oraz tak by nowa reprezentacja danych uzyskana po operacji kodowania miała jakieś przydatne właściwości [28].

W rzeczywistych problemach znaczną komplikacją jest identyfikacja czynników zmienności (ang. *factors of variation*) wpływających na dane [28]. Przykładem takiego czynnika modyfikującego dane może być kąt wykonania zdjęcia, który wpływa na kształt obiektów znajdujących się na obrazie, czy warunki oświetleniowe wpływające na kolory [28]. Większość problemów wymaga wyodrębnienia czynników zmienności i pozbycia

się tych które nie są istotne dla danego problemu co może być bardzo skomplikowanym i czasochłonnym zadaniem [28].

Idea głębokiego uczenia (ang. *deep learning*) wykorzystuje budowanie złożonych, abstrakcyjnych konceptów z prostszych, co pozwala rozwiązać ten główny problem uczenia reprezentacji wprowadzając podejście przedstawienia reprezentacji wyrażonych poprzez inne, prostsze reprezentacje [28].

Podstawowym modelem w głębokim uczeniu jest perceptron wielowarstwowy (ang. *multilayer perceptron*, MLP) [29, 30]. Perceptron wielowarstwowy może być postrzegany jako funkcja mapująca wartości przekazane na wejściu (warstwa wejściowa) na wartości otrzymywane na wyjściu (warstwa wyjściowa) za pomocą jednej lub więcej warstw ukrytych [28, 29, 30] (Rysunek 4.2).



Rysunek 4.2 Architektura perceptronu wielowarstwowego.

Tematyka niniejszej pracy skupia się na problemach związanych z przetwarzaniem obrazów. Rewolucja w rozwiązywaniu tego typu problemów wiąże się z zastosowaniem konwolucyjnych sieci neuronowych (CNN ang. convolutional neural networks). Patrząc na coroczny konkurs ImageNet Large Scale Visual Recognition Challenge (ILSVRC) polegający na rozpoznawaniu obrazów należących do 1000 kategorii jako punkt odniesienia dla ewaluacji istniejących algorytmów można powiedzieć, że punktem przełomowym był rok 2012 gdzie poziom błędu klasyfikacji osiągnięty przez głęboką konwolucyjną sieć neuronową nazwaną AlexNet osiągnął 15.3% przy 25.8% osiągniętych rok wcześniej [28, 31]. Od tego momentu modele oparte na głębokim uczeniu zdominowały konkurencję i w roku 2017 bład klasyfikacji wyniósł 2.3% [28, 31]. Podobnie jak perceptron wielowarstwowy konwolucyjna sieć neuronowa przetwarza coraz bardziej abstrakcyjne koncepty wraz z kolejnymi warstwami (Rysunek 4.3). Jest to przykład modelu czerpiacego inspirację z neuronauki, a dokładnie ze sposobu przetwarzania informacji w pierwszorzędowej korze wzrokowej (ang. primary visual *cortex*, V1). Metody oparte na głębokim uczeniu (w tym CNN) w ostatnich latach zyskały na znaczeniu i popularności. Głównymi czynnikami tego wzrostu było zwiększenie mocy obliczeniowej komputerów, powstanie bardzo obszernych zbiorów danych oraz rozwój technik uczenia pozwalających wykorzystać coraz głębsze sieci [28]. Nie bez wpływu było również powstanie narzędzi pozwalających na szybkie i stosunkowo proste budowanie i trenowanie sieci neuronowych takich jak np. TensorFlow [32] czy Keras [33]. Spowodowało to zainteresowanie się tematem szerszej grupy ludzi, ponieważ do zbudowania sieci neuronowej dającej dobre wyniki dla wielu problemów nie potrzeba dogłębnej wiedzy teoretycznej czy tworzenia własnych algorytmów. Można użyć wysokopoziomowego API (interfejsu programistycznego aplikacji ang. Application Programming Interface) np. Keras [33] bez zagłębiania się w wiedzę teoretyczną. Powstało wiele architektur sieci neuronowych które można w ten sposób wykorzystać z sukcesem do wielu problemów bez modyfikacji lub z małymi zmianami nie wymyślając za każdym razem architektury od zera. Pozwoliło to wykorzystać głębokie uczenie do rozwiązywania realnych problemów m.in. w zastosowaniach komercyjnych np. w analizowani zwyczajów zakupowych konsumentów czy personalizowaniu i profilowaniu reklam przez wiele firm. Konwolucyjne sieci neuronowe (CNN) stały się główną metodą rozwiązywania problemów

klasyfikacji i segmentacji obrazów. Analizując przepływ sygnałów w pierwszorzędowej korze wzrokowej (ang. *primary visual cortex*, V1) i porównując go z zasadą działania i popularnymi architekturami CNN można zauważyć wiele podobieństw jak np. warstwowa struktura ale istnieją również kluczowe różnice. Przykładowo w CNN te same filtry operują na każdym fragmencie obrazu, a w biologicznym pierwowzorze "filtry" są różne dla różnych fragmentów pola widzenia. Ponadto konwolucyjne sieci neuronowe są sprzężone w przód (ang. *feedforward*), a w korze wizyjnej występuje również wiele połączeń realizujących sprzężenie zwrotne.

Powstała również platforma CUDA [34] pozwalająca trenować modele znacznie wydajniej z wykorzystaniem kart graficznych również na komputerach osobistych, czy usługi pozwalające wykonywać obliczenia w chmurze, również udostępniające moc obliczeniową za darmo jak robi to Google Colab [35]. Powstała również platforma zrzeszająca osoby zainteresowane tematem uczenia maszynowego Kaggle [36]. Agreguje ona znaczną liczbę zbiorów danych oraz pozwala na wymianę wiedzy. Zwiększenie zainteresowania uczeniem maszynowym przez sektor komercyjny jak i osoby prywatne, u których to zainteresowanie owocuje często wykorzystywaniem tej wiedzy w pracy zawodowej lub naukowej pozwala sądzić, że rozwój wiedzy akademickiej jak i oprogramowania związanego z uczeniem maszynowym będzie postępował co w połączeniu z ciągle wzrastającą mocą obliczeniową komputerów będzie napędzało dalszy wzrost znaczenia tej dziedziny i coraz szersze wykorzystanie jej w wielu obszarach.



Rysunek 4.3 Architektura konwolucyjnej sieci neuronowej na przykładzie sieci VGG16 [37]. Wraz z przesuwaniem się w stronę głębszych warstw sieci wzrasta ilość filtrów w poszczególnych warstwach, a zmniejsza się rozmiar warstwy ze względu na zastosowanie warstw łączenia maksimum.



Rysunek 4.4 Wizualizacja przykładowych filtrów konwolucyjnych o rozmiarze 3x3 na przykładzie architektury CNN nazwanej VGG16 [37]. Każdy filtr pozwala wyodrębnić inną cechę obrazu. Filtry z pierwszej warstwy operują na oryginalnym obrazie, a te z kolejnych, głębszych warstw na coraz bardziej abstrakcyjnych wynikach działania filtrów z poprzednich warstw.



(b)

(a)



Rysunek 4.5 Wizualizacja map cech (ang. *feature maps*) na przykładzie architektury CNN nazwanej VGG16 [37] (b) oraz przykładowy oryginalny obraz którym zasilono sieć neuronową (a). Można zauważyć, że każda mapa zawiera odpowiedź na inną cechę obrazu wejściowego.

CZĘŚĆ DOŚWIADCZALNA

5 Wieloetapowe podejście do segmentacji zaszumionych obrazów na przykładzie obrazów z mikro-tomografii

5.1 Wprowadzenie

Niniejszy rozdział powstał na podstawie wcześniejszych publikacji autora rozprawy (jako pierwszy autor) [14, 38, 39]. Dotyczy to zarówno samego algorytmu, jego opisu jak i rysunków. Celem algorytmu przedstawionego w tymże rozdziale jest przeprowadzenie segmentacji obrazów uzyskanych przy użyciu mikro tomografii (μCT), która jest techniką dającą duże możliwości w zakresie nieniszczącego obrazowania wewnętrznych struktur niewielkich obiektów [38]. Najlepsze urządzenia wykorzystujące tę technikę są w stanie osiągnąć rozdzielczość przekraczającą nawet jeden mikrometr [39]. Takie możliwości mikro tomografii pozwalają wykorzystać szeroko tę technikę w geologii, biologii, materiałoznawstwie oraz innych dziedzinach wymagających obrazowania niewielkich struktur [39]. Zasada działania mikro tomografii opiera się na generowaniu promieniowania elektromagnetycznego z zakresu promieniowania rentgenowskiego przy wykorzystaniu lampy rentgenowskiej, która dzięki zastosowaniu wysokiego napięcia przyspiesza elektrony [40, 41, 42]. Tak powstała wiązka elektronów podczas spotkania z materiałem anody lampy emituje promieniowanie hamowania (Bremsstrahlung) jako ciągłe widmo promieniowania elektromagnetycznego w zakresie promieniowania rentgenowskiego [40, 41, 42]. W celu uzyskania obrazu wiązka promieniowania rentgenowskiego przepuszczana jest przez obrazowany obiekt gdzie ulega absorbcji oraz rozproszeniu w różnym stopniu, zależnym od materiału (jego gęstości, grubości, rodzaju oraz długości fali promieniowania), przez który przechodzi [40, 41, 42]. Natężenie promieniowania rentgenowskiego po przejściu przez materiał wyraża się wzorem [40, 41, 42]:

$$I = I_0 e^{-\mu(\lambda)d} \tag{5.1}$$

gdzie:

I – natężenie promieniowania po przejściu przez materiał

 I_0 – początkowe natężenie promieniowania padającego na materiał

 $\mu(\lambda)$ – liniowy współczynnik osłabienia zależny od długości fali promieniowania, gęstości materiału oraz jego rodzaju

d – grubość materiału

Opisany powyżej efekt osłabienia promieniowania rentgenowskiego po przejściu przez formie materiał pozwala na zobrazowanie różnic w badanym materiale w dwuwymiarowego obrazu dzięki zastosowaniu dwuwymiarowego detektora umieszczonego po przeciwnej stronie materiału (przeciwnej od padającej wiązki promieniowania rentgenowskiego). Energia promieniowania deponowana jest w detektorze. Tak uzyskana dwuwymiarowa tablica intensywności promieniowania po przejściu przez badany materiał nazywana jest projekcja [43, 44]. Próbke można obracać w celu zebrania wielu (setek lub tysięcy) obrazów dwuwymiarowych. Można to porównać do rzutowania trójwymiarowego obiektu na płaszczyzne dwuwymiarową pod różnymi kątami. Z takiego zbioru projekcji przy pomocy algorytmów komputerowych rekonstruowany jest obraz trójwymiarowy. Obraz taki zawiera informacje o wewnętrznej strukturze obrazowanego obiektu. Jedna z popularniejszych metod takiej rekonstrukcji jest filtrowana projekcja wsteczna [43, 44]. Obiekt w przestrzeni trójwymiarowej reprezentowany jest jako trójwymiarowa tablica wokseli (ang. voxel). Woksel można traktować jako trójwymiarowy odpowiednik piksela. Dwuwymiarowa tablica wokseli reprezentująca przekrój obiektu trójwymiarowego może być traktowana jako obraz dwuwymiarowy. Tak uzyskany obraz dwuwymiarowy w każdym pikselu przechowuje informacje z odpowiadającego mu woksela. Jest to informacja o pewnym trójwymiarowym wycinku obiektu przedstawiona w postaci dwuwymiarowej.

Przetwarzane obrazy uzyskano w sposób opisany w powyższym akapicie. Obrazy udostępnił dr hab. inż., prof. AGH Jacek Tarasiuk. Opisywany algorytm skupia się na segmentacji dwuwymiarowych obrazów (przekrojów) struktur porowatych w celu segmentacji rozdzielającej obie fazy badanego materiału na przykładzie obrazów w postaci

65

przekrojów piany polimerowej. Tego typu piany znajdują przykładowo zastosowanie w medycynie regeneracyjnej jako wypełnienie ubytków kości [45, 46]. Segmentacja przekrojów takich struktur pozwala na analizę tego typu materiałów. Porowate struktury można opisać jako dwufazowe kompozyty gdzie występuje faza stała oraz druga faza w postaci próżni, gazu badź cieczy [39]. Gdy różnica pomiędzy liniowym współczynnikiem osłabienia promieniowania rentgenowskiego pomiędzy fazami jest duża prowadzi to do dużej różnicy natężenia promieniowania po przejściu przez próbkę dla każdej z faz co z kolei reprezentowane jest na dwuwymiarowych obrazach przekrojów próbki jako duża różnica w intensywności pikseli. Takie obrazy są stosunkowo łatwe do segmentacji ponieważ wariancja pomiędzy klasami reprezentującymi obie fazy jest duża, a wariancja w poszczególnych klasach mała więc proste algorytmy progowania jak np. algorytm Otsu, zastosowane po etapie prostej filtracji np. filtrem medianowym czy bilateralnym prowadzą do dobrej separacji faz. Obrazy przetwarzane przez opisywany algorytm posiadają małe różnice w intensywności pikseli pomiędzy fazami (mała różnica liniowego współczynnika osłabienia promieniowania rentgenowskiego pomiędzy fazami) oraz dodatkowo obrazy cechują się dużą ilością szumu więc takie proste podejście jak opisane powyżej nie da dobrych rezultatów (Rysunek 5.1). Po segmentacji wszystkich dwuwymiarowych przekrojów z trójwymiarowego stosu takich binarnych obrazów można np. stworzyć trójwymiarową wizualizację [14]. Problem opisany w niniejszym rozdziale wymaga bardziej rozbudowanego podejścia.



Rysunek 5.1 Fragment oryginalnego obrazu przekroju piany polimerowej.

5.2 Opis Algorytmu

Algorytm został użyty na dość skomplikowanych obrazach. Podobieństwo intensywności pikseli należących do obiektu i tła jest główną trudnością wpływającą na jakość segmentacji. Niektóre obiekty są tak trudne do segmentacji, że nawet człowiek ma trudność we wskazaniu położenia krawędzi. W opisywanym podejściu wykorzystano połączenie wielu metod segmentacji w celu uzyskania zadowalających rezultatów. Działanie takie jest konieczne, ponieważ proste metody segmentacji bazujące na intensywności pikseli jak progowanie nie radzą sobie dobrze z mocno zaszumionymi obrazami na których znajdują się niejednolite obiekty [47, 48, 49]. Opisywany algorytm skupia się na łączeniu kilku algorytmów takich jak detektor krawędzi Canny–Deriche [21], filtr bilateralny [13], GVF [27], aktywne kontury [25, 26], statystyczne łączenie obszarów [23] czy wieloprogowe progowanie Otsu [18]. Ma to na celu uzyskanie dobrych wyników przy pomocy stworzonego wieloetapowego podejścia do segmentacji zadanych obrazów.

Przetwarzane obrazy są obrazami w skali szarości z głębią 8-bit. Pierwszym krokiem jest normalizacja histogramu zastosowana do wejściowego obrazu w celu zwiększenia kontrastu. W przypadku przetwarzania silnie zaszumionych obrazów kluczowym etapem jest filtracja polegająca na ograniczeniu szumu co pozwala uzyskać lepsze wyniki segmentacji na kolejnych etapach. W opisywanym algorytmie ten krok zrealizowano wykorzystując filtr bilateralny (patrz rozdział 3.2.3). Możliwe do ustawienia parametry dla filtra bilateralnego to: rozmiar maski, zakres intensywności, sigma w dziedzinie przestrzennej oraz sigma w dziedzinie intensywności (patrz rozdział 5.2.2). Krok wygładzania obrazu filtrem bilateralnym został zastosowany iteracyjne w dwóch iteracjach (Rysunek 5.2). W pierwszym kroku z parametrami zapewniającymi większe wygładzanie, a w drugim przebiegu z mniejszym wygładzaniem. Krok ten ma na celu uproszczenie obrazu poprzez usunięcie niechcianych tekstur oraz szumu z jednoczesnym pozostawieniem informacji o obiektach.



(a)



Rysunek 5.2 Fragment oryginalnego obrazu po filtracji filtrem medianowym. Pierwsza iteracja (a) oraz druga iteracja (b).
Kolejnym etapem jest detekcja krawędzi z użyciem detektora krawędzi Canny-Deriche. Parametry możliwe do modyfikacji na tym etapie: alfa, górny próg oraz dolny próg (patrz rozdział 5.2.2). Detekcja krawędzi rozpoczyna się od obliczenia wielkości i kierunku gradientu, a następnie przeprowadzenia procesu tłumienia niemaksymalnych wartości (ang. *non-maximum suppression*) co powoduje ścienienie krawędzi (Rysunek 5.3 a, b, c). Ostatnim krokiem detekcji krawędzi jest progowanie z histerezą (patrz rozdział 4.3.2) stosowane w celu pozostawienia najważniejszych krawędzi (Rysunek 5.3 d).

Następnie przeprowadzane jest grupowanie wykrytych krawędzi w celu przypisania ich do konkretnego obiektu. Osiągane jest to poprzez grupowanie wszystkich pikseli należących do krawędzi znajdujących się na obrazie uzyskanym w poprzednim kroku. Piksele, które leżą w odległości mniejszej od siebie niż zadany próg traktowane są jako należące do jednego obiektu. Następne etapy polegają na przetwarzaniu każdego obiektu osobno.



(a)



Rysunek 5.3 Detektor krawędzi Canny-Deriche: fragment oryginalnego obrazu (a), mapa gradientu (b), mapa gradientu po ścienieniu krawędzi (c) oraz wyniki detekcji krawędzi po progowaniu z histerezą (d).

5.2.1 Przetwarzanie indywidualnych obiektów

Przetwarzanie indywidualnych obiektów rozpoczyna się od wyznaczenia mapy gradientu metodą GVF (ang. gradient vector flow) (Rysunek 5.4) (patrz rozdział 4.5.2), która pozwoli na dalsze przetwarzanie obrazu metodą aktywnych konturów (patrz rozdział 4.5.1). Model aktywnych konturów inicjalizowany jest na zewnątrz każdego z obiektów. Następnie model ewoluuje w celu jak najdokładniejszego dopasowania do krawędzi obiektu poprzez minimalizację energii w kolejnych iteracjach (Rysunek 5.5). Na energię modelu wpływ ma kilka składowych: energia wewnętrzna (pierwsza i druga pochodna), energia zewnętrzna (wyznaczona na podstawie mapy gradientu) oraz zewnętrzna energia nacisku. W prezentowanym podejściu do wyznaczenia całkowitej energii modelu użyto dyskretnego modelu deformowalnego opartego na skończonej ilości punktów. Przed każdą iteracją model podlegał ponownemu próbkowaniu, co przy skończonej ilości punktów zapewnia ich równomierne rozłożenie na konturze, a tym samym prawidłowe zachowanie modelu. W celu optymizacji ewolucji modelu wykorzystano algorytm Viterbiego.



Rysunek 5.4 Przykładowa mapa gradientu uzyskana przy pomocy metody GVF.



Rysunek 5.5 Przykłady mapy gradientów uzyskanych przy pomocy metody GVF oraz deformowalnych modeli które iteracyjnie wyewoluowały w celu odnalezienia krawędzi obiektów. Niebieskie punkty oznaczają dyskretne punkty użyte w celu wyznaczenia energii modelu, a czerwone linie są połączeniami pomiędzy nimi.

Krzywa uzyskana przy pomocy deformowalnego modelu (o znanych koordynatach) pozwala wyznaczyć obszar pikseli wewnątrz takiej krzywej. Nakładając ten wyznaczony obszar na oryginalny obraz (po etapie filtracji wstępnej) otrzymujemy jego wycinek zawierający obiekt (Rysunek 5.6). Statystyczne łączenie obszarów (patrz rozdział 4.3.4) jest stosowane do tak uzyskanego obrazu z intencją łączenia pikseli o podobnej intensywności w celu uproszczenia obrazu (Rysunek 5.7). Tak uzyskane obrazy są następnie poddawane segmentacji z użyciem wieloprogowego progowania bazującego na algorytmie Otsu (Rysunek 5.8) (patrz rozdział 4.3.1). Jeżeli obiekty mają małą wysokość bądź szerokość (mniejszą niż 60) lub wszystkie piksele na obrazie posiadają wartości większe niż arbitralnie ustawiony próg to używane jest zwykłe progowanie z jednym progiem. Ostatecznie wyniki segmentacji dla wszystkich obrazów przetwarzanych indywidualnie prezentowane są jako jeden obraz wynikowy (Rysunek 5.9).



Rysunek 5.6 Przykłady pikseli wyciętych z oryginalnego obrazu po filtracji wstępnej przy pomocy maski uzyskanej metodą deformowalnych modeli.



Rysunek 5.7 Przykłady pikseli wyciętych z oryginalnego obrazu po filtracji wstępnej przy pomocy maski uzyskanej metodą deformowalnych modeli oraz po wykorzystaniu metody statystycznego łączenia obszarów.



Rysunek 5.8 Przykłady binarnych obrazów uzyskanych przy pomocy wieloprogowego progowania Otsu.



Rysunek 5.9 Diagram przepływu opisanego algorytmu.

5.2.2 Dostosowanie wartości parametrów

W celu uzyskania dobrych wyników zarówno dla małych jak i dużych obiektów dostosowano następujące parametry dla poszczególnych etapów przetwarzania: filtr bilateralny (rozmiar maski, zakres intensywności, sigma w dziedzinie przestrzennej oraz intensywności), detektor krawędzi Canny-Deriche (alfa, górny próg, dolny próg) oraz próg dla etapu usuwania zbyt krótkich krawędzi.

Poniższa procedura zastosowana została do dwóch mniejszych fragmentów oryginalnych obrazów zawierających odpowiednio mniejsze i większe obiekty [38].



(a)

(b)

Rysunek 5.10 Fragmenty przetwarzanych obrazów w oryginalnym rozmiarze: zawierające małe obiekty (a) oraz duże obiekty (b).

W celu odnalezienia optymalnych parametrów, krawędzie zostały manualnie wyznaczone na testowanych fragmentach obrazów. Parametry zostały dostrojone poprzez porównanie wyników detekcji krawędzi do wyników oznaczenia manualnego. Zbiory parametrów dające wyniki najbliższe rzeczywistym zostały wykorzystane do przetworzenia oryginalnych obrazów (zarówno z dużymi jak i z małymi obiektami). W przypadku dostrajania parametrów na fragmentach obrazów górny i dolny próg (detektor krawędzi Canny-Deriche) mogą wymagać manualnych poprawek ponieważ mapy gradientów pełnego obrazu i jego fragmentu mogą mieć różne intensywności z powodu normalizacji wielkości gradientu. Problem jest zminimalizowany w przypadku przeprowadzenia dopasowania parametrów na pełnym obrazie ale takie podejście jest bardzo pracochłonne z powodu konieczności ręcznego oznaczenia krawędzi dla dużej ilości obiektów (trudność jest mniejsza gdy na obrazie jest mała ilość obiektów).

Obrazy przypisano do dwóch grup. Grupa pierwsza zawierające małe obiekty oraz grupa druga zawierające duże obiekty. Podział obrazów pomiędzy grupami osiagnieto poprzez przeprowadzenie segmentacji zgrubnej czyli odnalezienie krawędzi z wykorzystaniem opisywanego algorytmu z użytym wybranym ręcznie wcześniej zestawem parametrów skutkującym dającymi się interpretować wynikami dla obu grup. Następnie dla przetwarzanych obrazów wszystkie piksele wykryte jako krawędzie zostały zgrupowane w postaci obiektów. Piksele których odległość od siebie jest mniejsza niż zadany próg traktowane są jako jeden obiekt. Algorytm oblicza średnią ilość pikseli na obiekt dla danego obrazu. Zakłada się, że obiekt z większą ilością pikseli należących do krawędzi jest wiekszy niż ten zawierający ich mniej. Na tej podstawie przy zadany progu rozdziału pomiędzy grupami obrazy przyporządkowywane są do odpowiedniej grupy. Po takim podziale obrazy w ramach grup są przetwarzane jeszcze raz już z optymalnym zestawem parametrów dla danej grupy. W zależności od rodzaju badanych obrazów można stworzyć więcej niż dwie grupy w celu dokładniejszego dobrania parametrów dla każdej z nich. Parametry mogą być wtedy analogicznie dostosowane dla każdej z grup. Takie podejście pozwala zautomatyzować przetwarzanie zbiorów setek bądź tysięcy obrazów reprezentujących warstwy obiektu badanego z wykorzystaniem mikro-tomografii. Parametry są ustawiane przy wykorzystaniu fragmentu jednego obrazu z takiego zbioru, a pod warunkiem podobnych rozmiarów obiektów na wszystkich obrazach ze zbioru można z tymi parametrami przetworzyć je wszystkie. (Rysunek 5.10).

5.3 Wyniki

Zastosowany algorytm pozwolił przeprowadzić segmentację porowatych struktur znajdujących się na obrazach o słabym kontraście. Dobrze dobrana dwuetapowa filtracja wstępna zastosowana wraz z detektorem krawędzi Canny-Deriche pozwoliła uzyskać poprawnie zlokalizowane krawędzie z małą ilością fałszywie pozytywnych i fałszywie negatywnych elementów krawędzi. Wyniki uzyskane w ten sposób charakteryzują się małymi nieciągłościami krawędzi co zostało przezwyciężone poprzez zastosowanie etapu wykorzystującego deformowalny model aktywnych konturów wraz z mapą gradientów uzyskaną metodą GVF osobno dla każdego obiektu. Pełna segmentacja uzyskiwana jest przy pomocy statystycznego łączenia regionów oraz wieloprogowego progowania Otsu (Rysunek 5.11). W ostatnim etapie wyniki przetwarzania osobnych obiektów łączone są w jeden wynikowy obraz (Rysunek 5.12). Prezentowany algorytm [14] jest wynikiem rozwoju algorytmów prezentowanych wcześniej [38, 39].



(a)

Rysunek 5.11 Dwa przykłady binarnych obrazów uzyskanych z wykorzystaniem opisanego algorytmu.



(a)

Rysunek 5.12 Dwa przykłady binarnych obrazów uzyskanych z wykorzystaniem opisanego algorytmu pokazujących większe fragmenty finalnych segmentacji.

5.3.1 Ewaluacja oraz porównanie wyników

W celu dokładnej ewaluacji wyników potrzebna jest znajomość dokładnego położenia wszystkich pikseli należących zarówno do obiektów jak i tła. Standardowym podejściem jest manualne etykietowanie pikseli obiektów co jest kosztowne czasowo oraz nie jest idealnie precyzyjne. Trudności te nasilają się wraz ze wzrostem złożoności obiektów. W celu ewaluacji wyników niniejszego algorytmu zastosowano sztuczne obrazy przygotowane tak aby najlepiej imitowały prawdziwe obrazy mikro-tomografii. Jako podstawę do stworzenia sztucznego obrazu wykorzystano binarny obraz wynikowy niniejszego algorytmu. Następnie obiekty oraz tło otrzymały kolory w skali szarości na podstawie średnich wartości tych elementów na oryginalnych obrazach. Kolejnym etapem jest dodanie szumu wygenerowanego z wykorzystaniem różnych wartości odchylenia standardowego w celu jak najlepszego naśladowania szumu powstałego na obrazach pochodzących z mikro-tomografii (Rysunek 5.13).



(a)

Rysunek 5.13 Fragmenty sztucznych obrazów z dodanym szumem o odchyleniu standardowym równym 12 (a) oraz 20 (b).

Dla celów porównawczych opisywanego algorytmu z innymi technikami segmentacji przeprowadzono segmentację sztucznego obrazu z dodanym szumem o odchyleniu standardowym równym 12 (wartość zbliżona do poziomu zaszumienia oryginalnych obrazów) przy pomocy porównywanych algorytmów (Rysunek 5.14). Przeprowadzono ewaluacje wyników z wykorzystaniem kilku miar takich jak czułość, swoistość, precyzja, wartość predykcyjna ujemna oraz dokładność (patrz Tabela 5.1). Test wykonany na sztucznych obrazach pokazuje, że podejście wykorzystujące proste progowanie daje słabe rezultaty. Znacznie lepsze rezultaty daje podejście wykorzystujące wieloprogowe progowanie Otsu, które zwraca wyniki porównywalne do opisywanego algorytmu lecz pozostawia małe artefakty wynikające z zaszumienia testowanego obrazu (Rysunek 5.14). Analogicznie przetworzono oryginalne obrazy co pokazało, że opisywane podejście daje lepsze rezultaty niż porównywane algorytmy (Rysunek 5.15). Oryginalne obrazy z mikrotomografii sa trudniejsze do segmentacji niż sztuczne obrazy, co spowodowane jest większym skomplikowaniem i różnorodnością obiektów znajdujących się na obrazach oryginalnych. Występują np. różnice w odcieniach szarości wewnątrz obiektów które czasami są podobne do wartości tła. Losowy szum z zadanym odchyleniem standardowym prawdopodobnie również nie oddaje złożoności oryginalnego szumu.

| Odchylenie standardowe szumu | 6 | 8 | 10 | 12 | 20 |
|------------------------------|-------|-------|-------|-------|-------|
| czułość | 0,964 | 0,955 | 0,944 | 0,932 | 0,711 |
| swoistość | 0,990 | 0,989 | 0,980 | 0,983 | 0,978 |
| precyzja | 0,887 | 0,874 | 0,803 | 0,816 | 0,729 |
| wartość predykcyjna ujemna | 0,997 | 0,996 | 0,995 | 0,994 | 0,976 |
| dokładność | 0,988 | 0,986 | 0,978 | 0,979 | 0,958 |

Tabela 5.1 Ewaluacja wyników segmentacji sztucznych obrazów testowych dla różnych wartości odchylenia standardowego szumu. Na podstawie [14].





(a)









Rysunek 5.14 Fragmenty sztucznych obrazów z odchyleniem standardowym szumu równym 12 przetworzone przez (a) opisywany algorytm z progowaniem pojedynczym progiem, (b) pierwszy etap opisywanego algorytmu bez drugiego etapu filtra bilateralnego oraz z wieloprogowym progowaniem Otsu, (c) pierwszy etap opisywanego algorytmu z wieloprogowym progowaniem Otsu, (d) opisywany algorytm.



(a)



(b)



(c)



(d)

Rysunek 5.15 Fragmenty oryginalnych obrazów przetworzone przez opisywany algorytm: (a) opisywany algorytm z progowaniem pojedynczym progiem, (b) pierwszy etap opisywanego algorytmu bez drugiego etapu filtra bilateralnego oraz z wieloprogowym progowaniem Otsu, (c) pierwszy etap opisywanego algorytmu z wieloprogowym progowaniem Otsu, (d) opisywany algorytm.

5.4 Podsumowanie

Opisywany algorytm pokazuje dobrą skuteczność w segmentacji skomplikowanych struktur dzięki wprowadzeniu dodatkowego etapu segmentacji co znacznie poprawiło jakość segmentacji. Algorytm radzi sobie również dobrze z zaszumionymi obrazami. W rozdziale 5.3.1 pokazano przewagę opisywanego algorytmu nad porównywanymi podstawowymi podejściami. Poprawienie jakości segmentacji wiąże się ze zwiększeniem złożoności obliczeniowej z powodu dodania etapu polegającego na przetwarzaniu każdego obiektu osobno. Ze względu na dość dużą złożoność obliczeniową algorytm słabo nadaje się do przetwarzania obrazów w czasie rzeczywistym lecz jego podstawowe zastosowanie to przetwarzanie obrazów uzyskanych wcześniej przy użyciu mikro-tomografii więc czas przetwarzania nie ma większego znaczenia dopóki mieści się w rozsądnych ramach. W takim zastosowaniu zwiększenie jakości uzyskiwanych wyników jest głównym celem więc poprawienie wyników nawet kosztem dłuższego czasu przetwarzania jest jak najbardziej pożądane.

6 Segmentacja erytrocytów z obrazów z mikroskopu optycznego wraz z obliczaniem stosunku ich wysokości do szerokości

6.1 Wprowadzenie

Niniejszy rozdział powstał na podstawie publikacji autora rozprawy (jako pierwszy autor) [50]. Dotyczy to zarówno samego algorytmu, jego opisu jak i rysunków. Głównym celem algorytmu jest obliczenie wartości statystycznych obiektów znajdujących się na obrazach erytrocytów takich jak ich ilość, rozmiar, stosunek szerokości do wysokości. Obrazy udostępniła prof. dr hab. Kvetoslava Burda. Uzyskano je przy użyciu mikroskopu optycznego. Mikroskopia optyczna jest to technika szeroko wykorzystywana w naukach przyrodniczych z powodu jej prostoty, niskich kosztów, dostępności oraz dużych możliwości w tym obszarze nauki. W nowoczesnych mikroskopach często stosuje się cyfrowe kamery w celu digitalizacji obrazów badanych obiektów. Tak uzyskane cyfrowe obrazy erytrocytów mogą posłużyć do automatyzacji procesu ich zliczania oraz oceny kształtu. Taka automatyzacja pozwala oszczędzić znaczną ilość czasu potrzebną na manualne wykonanie tego zadania. Innym przykładowym zagadnieniem gdzie podejmuje się próby automatyzacji jest segmentacja zmian skórnych [51]. Prezentowany algorytm pozwala na otrzymanie dobrych wyników gdy nie ma dostępu do dużych zbiorów danych z obiektami oznaczonymi przez eksperta jako źródło wiedzy do treningu algorytmów uczących się. Może być również użyty do przetwarzania map prawdopodobieństwa otrzymanych z modeli głębokiego uczenia (np. U-net) w celu uzyskania ostatecznej segmentacji. Zaprezentowano również użycie, gdzie model głębokiego uczenia pozwala klasyfikować wyniki otrzymane za pomocą opisanego algorytmu. Algorytm został opublikowany w artykule pod tytułem: Segmentation of Microscope Erythrocyte Images by CNN-Enhanced Algorithms [50].

6.2 Opis Algorytmu

Obrazy erytrocytów analizowane przy pomocy tego algorytmu są przetwarzane jako obrazy w skali szarości. Główne problemy napotykane podczas segmentacji tych obrazów to fakt, że niektóre erytrocyty stykają się ze sobą, jak również szum na obrazie oraz nierównomierne tło. Nierównomierny profil tła nie jest dobrze widoczny na oryginalnym obrazie więc zastosowano transformację WW/WL w celu lepszego ukazania tej cechy obrazu (Rysunek 6.1).



Rysunek 6.1 Oryginalny obraz w skali szarości (a), obraz po transformacji WW/WL (b).

6.2.1 Wstępne przetwarzanie i segmentacja

Schemat kroków algorytmu daje pogląd na całość procedury (Algorytm 1,

Algorytm 2). Obrazy konwertowane są do skali szarości. Przeprowadzane jest wstępne wygładzanie obrazu z użyciem filtrowania uśredniającego. Obrazy posiadają nierównomierne tło (efekt powodowany podświetleniem preparatu zastosowanym w mikroskopie), są zaszumione oraz w wielu przypadkach stykają się ze sobą. Filtry medianowy, uśredniający i bilateralny zostały użyte w celu redukcji szumów. Do wyrównania tła użyto algorytmu toczącej się kuli [15] (ang. *rolling ball*, wykorzystanie

algorytmu z oprogramowania Fiji [52]) (opis w rozdziale 3.2.4) (Rysunek 6.2). Kolejne kroki algorytmu wykonano przy użyciu biblioteki ITK(Insight Segmentation and Registration Toolkit) [53]. Po odjęciu tła obraz poddano procesowi usuwania szumu w postaci zastosowania filtrów: medianowego (opis w rozdziale 3.2.1), uśredniającego (opis w rozdziale 3.2.2) i na końcu bilateralnego (opis w rozdziale 3.2.3) (Rysunek 6.3, Rysunek 6.4).



Rysunek 6.2 Odejmowanie tła. Fragmenty obrazów z odpowiadającymi im profilami intensywności. Górna część – oryginalny obraz, dolna część – obraz po odjęciu tła.



Rysunek 6.3 Usuwanie szumów: oryginalny obraz (a), po filtracji filtrem medianowym (b), po filtracji filtrem medianowym i uśredniającym (c).



Rysunek 6.4 Obraz po użyciu filtra bilateralnego.

W celu uzyskania wstępnej segmentacji obrazu wykorzystano algorytm Otsu (opis w rozdziale 4.3.1) (Implementacja ITK [53], Rysunek 6.5). Otwory powstałe w tak uzyskanych obiektach zostały wypełnione przy pomocy implementacji algorytmu pochodzącej z biblioteki ITK [53, 54, 55] (Rysunek 6.5). Następnie obraz został poddany erozji (opis w rozdziale 4.2.2) (implementacja ITK [53], Rysunek 6.6) i posłużył

do stworzenia mapy odległości (opis w rozdziale 4.2.1) (implementacja ITK [53], Rysunek 6.6). Wynikową mapę zmodyfikowano tak aby obraz posiadał białe tło oraz zmniejszające się wartości intensywności pikseli wraz ze wzrostem odległości pikseli obiektów do tła. Następnie mapę odległości poddano segmentacji z wykorzystaniem algorytmu segmentacji wododziałowej (opis w rozdziale 4.3.3) (implementacja ITK [53], Rysunek 6.7). Kombinacja algorytmu Otsu, transformacji odległości (ang. *distance transform*) oraz algorytmu wododziałowego (ang. *watershed*) zostały użyte do wstępnej segmentacji obrazu. Taka kombinacja pozwala oddzielić stykające się ze sobą erytrocyty. Następnie obiekty zbyt małe i te położone przy krawędziach obrazu zostały usunięte (Rysunek 6.7). Po tak przeprowadzonej wstępnej segmentacji każdy obiekt obecny na obrazie poddano drugiemu etapowi segmentacji.



Rysunek 6.5 Obraz po segmentacji algorytmem Otsu (a) oraz po wypełnieniu otworów w obiektach uzyskanych po segmentacji (b).



Rysunek 6.6 Obraz po erozji (a) i utworzona na jego podstawie mapa odległości (b).



Rysunek 6.7 Obraz po segmentacji algorytmem wododziałowym (a) oraz po usunięciu obiektów małych i położonych przy krawędziach obrazu zaprezentowany jako obraz binarny (b).

Ogólny zarys algorytmu segmentacji erytrocytów:

Wejście: Mikroskopowy kolorowy obraz erytrocytów (głębia 16-bit)

Wyjście: Obraz zawierający kontury i osie erytrocytów oraz statystyki powiązane z ilością, kształtem i rozmiarami erytrocytów

Krok 1: Przygotowanie wstępne: konwersja do skali szarości, odejmowanie tła i usuwanie szumu z obrazu

Krok 2: Wstępna segmentacja oparta na algorytmie Otsu, mapie odległości, algorytmie wododziałowym, operacjach morfologicznych, usuwaniu obiektów zbyt małych oraz położonych przy krawędziach oraz wypełnianiu otworów w obiektach

Krok 3: Przeprowadzenie dokładnej segmentacji osobno dla każdego obiektu uzyskanego po etapie wstępnej segmentacji:

- Wybranie ROI (obiekt z odpowiednim marginesem) i maskowanie obrazu (maskowanie na podstawie poszerzonego obiektu i wycięcia sąsiednich obiektów)
- Segmentacja algorytmem Otsu i wypełnienie otworów w obiektach
- Wybranie największego obiektu na obrazie do dalszego przetwarzania
- Wyznaczenie konturu obiektu oraz jego osi dłuższej i krótszej

Krok 4: Nałożenie konturów wykrytych erytrocytów wraz z ich osią krótką i dłuższą na oryginalny obraz. Dla każdego erytrocytu zapisanie wyliczonej powierzchni wraz z długością osi krótszej i dłuższej oraz ich stosunku. Dla całości obrazu zapisanie ilości erytrocytów na obrazie i wartości średnich parametrów wyznaczonych dla pojedynczych erytrocytów.

Algorytm 1 Ogólny schemat całego algorytmu segmentacji erytrocytów.

Szczegółowy opis algorytmu segmentacji erytrocytów:

Wejście: Mikroskopowy kolorowy obraz erytrocytów (głębia 16-bit)

Wyjście: Obraz zawierający kontury i osie erytrocytów oraz statystyki powiązane z ilością, kształtem i rozmiarami erytrocytów

Krok 1: Konwersja do skali szarości – konwertuje obraz RGB do skali szarości

Krok 2: Filtr uśredniający – wstępnie wygładza obraz otrzymany w kroku 1. Parametry: Promień maski (r = 1)

Krok 3: Odejmowanie tła- odjęcie tła od obrazu z kroku 2 z użyciem algorytmu toczącej się kuli.

Parametry: promień elementu strukturalnego (r = 100)

Usuwanie szumu

Krok 4: Filtr medianowy – filtrowanie obrazu z kroku 3. Parametry: promień maski (r = 4)

Krok 5: Filtr uśredniający – filtrowanie obrazu z kroku 4. Parametry: promień maski (r = 3)

Krok 6: Przeskalowanie głębi bitowej obrazu – dla obrazu z kroku 5 zmiana głębi 16-bit na 8-bit.

Krok 7: Filtr bilateralny – filtrowanie obrazu z kroku 6. Parametry: sigma przestrzenna ($\sigma_d = 3$), sigma intensywności ($\sigma_r = 3$)

Wstępna segmentacja

Krok 8: Segmentacja algorytmem Otsu – obraz binarny z obrazu z kroku 7

Krok 9: Wypełnienie dziur – wypełnienie dziur wewnątrz obiektów po segmentacji z kroku 8

Krok 10: Morfologiczna erozja – erozja obrazu z kroku 9. Parametry: promień (r = 2)

Krok 11: Mapa odległości – mapa odległości dla obrazu z kroku 10

Krok 12: Segmentacja wododziałowa – segmentacja obrazu z kroku 11. Parametry segmentacji: próg (t = 0,004), poziom (L = 0,4)

Krok 13: Usuwanie małych obiektów – usunięcie małych obiektów z obrazu z kroku 12. Parametry: próg rozmiaru (t = 25)

Krok 14: Usuwanie obiektów bliskich do krawędzi obrazu – usuniecie obiektów bliskich do krawędzi obrazu dla obrazu z kroku 13. Parametry: próg ilości pikseli dotykających granicy obrazu (t = 0 (obiekt usuwany jeżeli przynajmniej jeden piksel obiektu dotyka krawędzi obrazu))

Indywidualne przetwarzanie poszczególnych erytrocytów

Krok 15: Dla każdego wstępnie segmentowanego erytrocytu:

Krok 15.1: Wybranie ROI – prostokątne ROI z przefiltrowanego obrazu (wyjście z kroku 7) dookoła przetwarzanego obecnie obrazu z odpowiednim marginesem. Parametry: margines (m = 10)

Krok 15.2: Tworzenie maski – tworzenie maski dla rozważanego obiektu, przeprowadzenie dylatacji maski i wycięcie z niej obszarów obiektów sąsiadujących. Parametry: promień dylatacji (r = 6)

Krok 15.3: Obraz maskowany – obraz maskowany przy pomocy obrazów z kroków 15.1 i 15.2

Krok 15.4: Segmentacja Otsu – przeprowadź segmentację obrazu z kroku 15.3

Krok 15.5: Wypełnienie otworów – wypełnienie otworów wewnątrz obiektów po segmentacji z kroku 15.4

Krok 15.6: Wybór największego obiektu – Jeżeli na obrazie znajduje się więcej niż jeden obiekt po kroku 15.5 to wybierz największy do dalszego przetwarzania

Krok 15.7: Kontur obiektu – wyznaczenie konturu obiektu z kroku 15.6

Krok 15.8: Znalezienie dłuższej osi erytrocytu – wyznaczenie odległości euklidesowej pomiędzy wszystkimi punktami konturu z kroku 15.7. Dwa punkty z największą odległością pomiędzy nimi definiują dłuższą oś erytrocytu.

Krok 15.9: Znalezienie krótkiej osi erytrocytu

Dla każdego punku osi długiej znalezionej w kroku 15.8

- Znalezienie prostopadłej prostej linii przechodzącej przez rozważany punkt
- Znalezienie wszystkich punktów konturu z kroku 15.7 z dystansem do tej linii mniejszym lub równym $\sqrt{2}$

• Jeżeli istnieją przynajmniej dwa punkty z dystansem pomiędzy nimi większym niż próg to algorytm przechodzi dalej, jeżeli nie krótka oś nie może zostać wyznaczona i algorytm kontynuuje proces z następnym punktem osi długiej (krok 15.9)

• Przyjęcie dwóch punktów z największą odległością pomiędzy sobą jako pozycje dwóch klastrów punktów

• Przypisanie wszystkich pozostałych punktów do klastrów bazując na odległości euklidesowej

• Dla każdego klastra wybór jednego punktu najbliższego do rozważanej prostej linii. Oznaczenie wybranych punktów jako kandydatów na punkty przecięcia.

• Wyznaczenie odległości pomiędzy tymi punktami.

• Jeżeli odległość pomiędzy tymi punktami jest większa niż dla pary wybranej w poprzednich iteracjach oznaczenie ich jako punkty przecięcia.

• Jeżeli odległość pomiędzy tymi punktami jest taka sama jak dla pary wybranej w poprzednich iteracjach wybranie pary dla której przechodząca przez nie prosta przebiega najbliżej środka długiej osi erytrocytu.

Jeżeli nie znaleziono żadnych kandydatów na punkty przecięcia przejście do kolejnego erytrocytu (krok 15). Dla obecnie rozważanego nie można wyznaczyć osi krótkiej.

Krok 16: Nałożenie konturów wykrytych erytrocytów wraz z ich osią krótką i dłuższą na oryginalny obraz. Dla każdego erytrocytu zapisanie wyliczonej powierzchni wraz z długością osi krótszej i dłuższej oraz ich stosunku. Dla całości obrazu zapisanie ilości erytrocytów na obrazie i wartości średnich parametrów wyznaczonych dla pojedynczych erytrocytów.

Algorytm 2 Szczegółowy schemat całego algorytmu segmentacji erytrocytów.

6.2.2 Indywidualne przetwarzanie i segmentacja obiektów

Każdy obiekt otrzymany po wstępnej segmentacji jest przetwarzany oddzielnie w celu uzyskania dokładniejszej segmentacji. Dla każdego obiektu w celu wyodrębnienia go z całego obrazu wyznaczono prostokątny region zainteresowania (ROI) z małym marginesem z każdej strony, a jeżeli w pobliżu przetwarzanego w danej chwili obiektu znajdował się inny obiekt to wykorzystano go do maskowania przetwarzanego obrazu (Rysunek 6.8b). Dodatkowa maska tworzona jest z samego przetwarzanego obiektu poddanego dylatacji (Rysunek 6.8c, d), a suma obu masek otrzymanych w ten sposób tworzy ostateczną maskę zastosowaną dla rozważanego ROI w celu rozdzielenia obiektów znajdujących się blisko siebie (Rysunek 6.8e).





(b)

(c)





Rysunek 6.8 Wybrane ROI (a), ROI maskowane przy użyciu obiektów w pobliżu tego obecnie przetwarzanego (b), maska przetwarzanego obiektu (c), maska przetwarzanego obiektu po dylatacji (d) oraz ostateczna kombinacja masek (e).

Na przefiltrowanym obrazie ograniczonym do aktualnego ROI i zamaskowanym maską otrzymaną w poprzednim kroku wykonywana jest segmentacja algorytmem Otsu. Segmentacji dokonano przy użyciu dwóch progów (Rysunek 6.9), a obraz wynikowy zawiera dwie klasy obiektów: ciemniejszy (zielony) oraz jaśniejszy (niebieski) (Rysunek 6.9 a). Korzystając z podobieństwa w kształcie poszczególnych erytrocytów, które przyjmują kształt dwuwklęsłych dysków można określić, która klasa zawiera krawędzie erytrocytu. Jest to klasa zawierająca ciemniejsze piksele (Rysunek 6.9 b). Tak otrzymany obiekt może zawierać otwory wewnątrz wyznaczonego obiektu. W większości przypadków takim ubytkiem będzie wewnętrzna wklęsła część erytrocytu. W celu uzyskania jednego pełnego obiektu jest on poddawany wypełnianiu otworów przy użyciu algorytmu z biblioteki ITK [53] (Rysunek 6.9 c). W niektórych przypadkach po tym etapie na obrazie może znajdować się więcej niż jeden obiekt, więc aby pozostał tylko jeden to do dalszego przetwarzania wybierany jest ten największy, a z tak zdefiniowanego obiektu wyznaczany jest jego kontur (Rysunek 6.9 d). Tak otrzymany kontur wykorzystany jest do wyznaczenia dłuższej i krótszej osi erytrocytu (Rysunek 6.10). W celu znalezienia dłuższej osi erytrocytu wyznacza się odległość euklidesową pomiędzy wszystkimi punktami konturu, a dwa punkty z największą odległością pomiędzy nimi definiują dłuższą oś erytrocytu (

Algorytm 2 – Krok 15.8). Krótsza oś erytrocytu definiowana jest jako najdłuższy możliwy odcinek z oboma końcami położonymi na konturze obiektu oraz jednocześnie prostopadły do dłuższej osi. Wyznaczana jest poprzez powtórzenie procedury opisanej w opisie algorytmu dla każdego punktu położonego na dłuższej osi (

Algorytm 2 - Krok 15.9).



Rysunek 6.9 Segmentacja Otsu z dwoma poziomami progowania (a), obiekt po segmentacji (b), obiekt po segmentacji z wypełnionymi otworami (c), kontur otrzymany z obrazu po segmentacji i wypełnieniu otworów (d).



Rysunek 6.10 Przykładowe kontury erytrocytów (czerwony) oraz krótkie (zielony) i długie (niebieski) osie erytrocytów.

6.2.3 Prezentowana technika segmentacji jako procesor map prawdopodobieństwa uzyskanych przy użyciu głębokiego uczenia

Wyniki otrzymywane na wyjściu sieci neuronowych często potrzebują dodatkowego kroku przetwarzania aby uzyskać ostateczny rezultat. Taki etap potrzebny jest gdy wynikiem działania sieci neuronowej jest mapa prawdopodobieństwa. Podejście do segmentacji prezentowane w niniejszym rozdziale z powodzeniem może być użyte jako procesor takich map prawdopodobieństwa. Do wytrenowania sieci neuronowej wykorzystano zbiór obrazów BBBC038v1 [56] z projektu "Broad Bioimage Benchmark Collection". Na rozpatrywanym zbiorze danych zastosowano klasteryzację w celu otrzymania zbiorów podobnych obrazów, ponieważ w przypadku modeli opartych na głębokim uczeniu łatwiej osiągnąć lepsze wyniki dla zbiorów danych zawierających podobne obrazy (przy założeniu podobnego rozmiaru tych zbiorów). Inaczej mówiąc, w celu osiągnięcia dobrych wyników na niejednorodnym zbiorze (uzyskanie bardziej ogólnego modelu) należy zapewnić odpowiednio większy zbiór danych uczących i odpowiednio pojemny model. Zbiór zawierający podobne obrazy pozwala zmniejszyć złożoność problemu, więc model głębokiego uczenia może osiągnąć odpowiednią generalizację dla zadanego problemu nawet dla stosunkowo małych zbiorów danych (patrz rozdział 8.3). Do klasteryzacji wykorzystano algorytm centroidów (k-średnich, ang. k-means) w przestrzeni barw HSV, a wszystkie obrazy zostały przeskalowane do tego samego rozmiaru 256×256 w celu zasilenia modelu. Model głębokiego uczenia (U-net) został wytrenowany na zbiorze 480 obrazów należących do jednego wybranego klastra. Podobne modele można również wytrenować na pozostałych klastrach otrzymując modele dopasowane do danego zbioru danych. Użyto dwóch metod w celu przetworzenia tak uzyskanych mapy prawdopodobieństwa. Pierwsza metoda to przyjęte jako punkt odniesienia często używane zastosowanie sekwencyjnej kombinacji metod: segmentacja algorytmem Otsu, erozja, mapa odległości oraz algorytm wododziałowy. Druga metoda to opisany w rozdziale 6.2 algorytm. W przypadku obu metod wykorzystano uproszczony etap filtracji wstępnej w postaci filtra bilateralnego. Analiza wyników działania obu metod pozwala stwierdzić, że algorytm opisany w rozdziale 6.2 lepiej radzi sobie z zadaniem rozdzielenia złączonych obiektów podczas segmentacji (Rysunek 6.11) co pozwala uzyskać lepszy wynik segmentacji dla tego samego modelu.

97



Rysunek 6.11 Mapa prawdopodobieństwa (wynik działania modelu) (a), mapa po segmentacji z użyciem referencyjnego podejścia (zbliżenie połączonych obiektów) (b), wynik proponowanego algorytmu (zbliżenie połączonych obiektów) (c). Proponowany algorytm daje trochę lepsze wyniki w zadaniu segmentacji połączonych obiektów znajdujących się na mapie prawdopodobieństwa.

6.2.4 Opisywana technika segmentacji połączona z uczeniem maszynowym w celu kategoryzacji wyników

Algorytm prezentowany w rozdziale 6.2 może zostać rozbudowany o analizę jego wyników przy użyciu metod głębokiego uczenia. Taką analizę przeprowadzono na podstawie 1339 obrazów pojedynczych erytrocytów otrzymanych w wyniku segmentacji 13 obrazów zawierających erytrocyty. Tak uzyskane obrazy pojedynczych erytrocytów zostały przypisane do trzech kategorii: normalne (1021 obiektów), anormalne (88 obiektów) oraz błędna segmentacja (230 obiektów) (Rysunek 6.12). Dla każdej kategorii wybrano losowo 88 próbek w celu wyrównania rozkładu próbek w poszczególnych kategoriach. Przyporządkowanie do klas nie zostało dokonane przez specjaliste w dziedzinie biologii komórkowej, więc podział na erytrocyty normalne i anormalne został wykonany bazując na podstawowej znajomości tematu. W celu otrzymania dokładniejszych wyników należałoby wykorzystać eksperta w tej dziedzinie do kategoryzacji danych. Użyty zbiór danych ma dużą dysproporcję rozmiaru pomiędzy kategoriami, jest niewielki a dane uczące nie były kategoryzowane przez eksperta, wiec opisany model głebokiego uczenia jest traktowany raczej jako sposób pokazania możliwych zastosowań opisywanego algorytmu segmentacji niż w pełni funkcjonalne rozwiązanie (choć uzyskane wyniki są obiecujące). Przykładowym zastosowaniem takiego podejścia może być np. wykluczenie erytrocytów z błędną segmentacją ze statystyki rozmiarów erytrocytów w celu uzyskania dokładniejszych wyników lub oznaczenie anormalnych obiektów w celu dokładniejszej ich analizy przez człowieka (wytrenowanego specjalistę).

Model zbudowano przy pomocy Keras API oraz TensorFlow. Wykorzystano uczenie transferowe (ang. *transfer learning*) z użyciem modelu MobileNetV2 [57] z wagami wytrenowanymi na zbiorze ImageNet. W pełni połączona warstwa została zastąpiona kombinacją: GlobalAveragePooling2D, Dropout, BatchNormalization, Dense(1280 units, ReLU activation), Dropout, BatchNormalization, Dense (predykcja trzech klas, aktywacja softmax). MobileNetV2 jest rozwojową wersją modelu MobileNet odpowiadającą na potrzebę mniej rozbudowanego modelu w celu stosowania go w sytuacjach ograniczonych zasobów sprzętowych. Jego wydajność została również zwiększona [57, 58]. Warstwy pochodzące z modelu MobileNetV2 zostały wyłączone z treningu, użyto

zestawu przekształceń w celu zwiększenia zbioru danych (ang. *data augmentation*). Zbiór walidacyjny zawierał 30% danych. Przykładowe wyniki zostały zaprezentowane w tabeli (Tabela 6.1). Zwiększenie zbioru danych pozwoliłoby uzyskać dokładniejsze wyniki.



Rysunek 6.12 Przykład erytrocytów po segmentacji skategoryzowanych jako normalne, anormalne oraz błędna segmentacja.

| Tabela 6.1 Wyr | niki predykcji mo | lelu na walidacyjnym | a zbiorze danych. | Na podstawie [50]. |
|----------------|-------------------|----------------------|-------------------|--------------------|
|----------------|-------------------|----------------------|-------------------|--------------------|

| | Precyzja (ang. precision) | Czułość (ang. <i>recall</i>) | Ilość obiektów |
|-----------------|------------------------------|-------------------------------|----------------|
| Anormalne | 0,79 | 0,58 | 26 |
| Normalne | 0,70 | 0,81 | 26 |
| Zła segmentacja | 0,72 | 0,81 | 26 |

6.3 Wyniki

Dla większość komórek znajdujących się na badanych obrazach segmentacja przebiegła poprawnie, również w przypadku erytrocytów, które znajdowały się obok siebie, ponieważ zostały poprawnie sklasyfikowane jako osobne obiekty. Dla wszystkich testowanych obiektów udało się wyznaczyć kontur oraz oś długa i krótką (Rysunek 6.10, Rysunek 6.13), co wymagane jest do otrzymania statystyki rozmiarów i kształtu. W celu poprawnej ewaluacji wyników dokładne pozycje wszystkich pikseli obiektów muszą być znane więc zastosowano podejście oparte na sztucznie generowanych obrazach (patrz rozdział 6.3.1). Kontury obiektu mogą być oznaczone ręcznie lecz jest to czasochłonne zadanie obarczone również błędem ludzkim. Ten sam obiekt może mieć inaczej wyznaczony kontur w zależności od tego kto dokonuje oznaczenia. Użycie syntetycznych danych pozwala ominąć te trudności.

6.3.1 Porównanie wyników ewaluacji do podejścia zgodnego ze stanem wiedzy

Obrazy wyjściowe otrzymane przy pomocy opisywanego algorytmu posłużyły za bazę do konstrukcji sztucznych obrazów testowych, na których pozycje wszystkich pikseli obiektów i tła są znane, więc można przeprowadzić statystyczną ewaluację wyników opisywanego algorytmu. Poziom podobieństwa pomiędzy oryginalnymi i sztucznie otrzymanymi obrazami powinien być jak największy ponieważ skorelowany jest on z miarodajnością przeprowadzonej w ten sposób ewaluacji wyników. Im sztuczne obrazy bliższe są oryginałom tym wyniki walidacji przeprowadzonej na sztucznych obrazach powinny być bardziej zbliżone do wyników jakie byłyby otrzymane przy użyciu obrazów oryginalnych. W tym celu m. in. wartości intensywności pikseli obiektów i tła zostały tak dobrane aby były jak najbardziej zbliżone do tych znajdujących się na obrazach oryginalnych.

Tło zostało przekształcone przy pomocy dwuwymiarowej funkcji Gaussa tak aby profil intensywności odpowiadał tłu z obrazu oryginalnego, które posiada wartości intensywności pikseli wyższe w okolicy centrum i coraz niższe w miarę oddalania się od niego, co jest spowodowane sposobem podświetlenia preparatu w mikroskopie optycznym. Erytrocyty utrwalone na obrazach uzyskanych przy pomocy mikroskopu optycznego charakteryzują się większymi wartościami intensywności pikseli wewnątrz, a mniejszymi w okolicach ich krawędzi, więc w celu odtworzenia takiej cechy wejściowy obraz binarny został poddany erozji w celu wyznaczenia wewnętrznego obszaru erytrocytu, który w połączeniu z obiektem przed erozją i poddany wygładzaniu Gaussa w celu odzwierciedlenia rozmytych krawędzi tworzy obraz erytrocytu. Jako finalny krok dodano szum charakteryzujący się kilkoma zdefiniowanymi wartościami odchylenia standardowego (Rysunek 6.14, Rysunek 6.15). Sześć kombinacji rozmiaru maski wygładzania Gaussa oraz odchylenia standardowego szumu zostało przetestowanych. Została dla nich przeprowadzona ewaluacja statystyczna z wykorzystaniem następujących miar klasyfikacji binarnej: czułość (ang. *sensitivity*), swoistość (ang. *specificity*), precyzja (ang. *precision*), dokładność (ang. *accuracy*) oraz wartość predykcyjna ujemna (ang. *negative predictive value*). Została obliczona również ilość obiektów wykrytych na obrazie (Tabela 6.2).

| Promień wygładzania Gaussowskiego | | 3 | | | 5 | |
|-----------------------------------|-------|-------|-------|-------|-------|-------|
| Odchylenie standardowe szumu | 6 | 15 | 30 | 6 | 15 | 30 |
| czułość | 0,988 | 0,988 | 0,994 | 0,996 | 0,996 | 0,993 |
| swoistość | 0,997 | 0,997 | 0,997 | 0,997 | 0,997 | 0,996 |
| precyzja | 0,978 | 0,980 | 0,978 | 0,981 | 0,976 | 0,968 |
| wartość predykcyjna ujemna | 0,998 | 0,998 | 0,999 | 0,999 | 0,999 | 0,999 |
| dokładność | 0,996 | 0,996 | 0,997 | 0,997 | 0,997 | 0,995 |
| Ilość wykrytych obiektów | 124 | 124 | 124 | 124 | 124 | 124 |

Tabela 6.2 Ewaluacja wyników segmentacji dla opisywanego algorytmu gdzie rzeczywista ilość obiektów znajdujących się na obrazie wynosiła 124. Na podstawie [50].

Wyniki algorytmu zostały porównane z wynikami uzyskanymi za pomocą tylko pierwszego etapu segmentacji tzn. bez wykonania drugiego etapu segmentacji (Tabela 6.3).

Podczas testów rozpatrywano dwa rozmiary elementu strukturalnego służącego do wygładzania Gaussowskiego (3 i 5), oraz trzy wartości odchylenia standardowego szumu (6, 15 i 30). Dla niektórych miar wartości nie zmieniają się znacząco wraz ze wzrostem odchylenia standardowego szumu użytego w danym pomiarze. Tego typu sytuacje prezentują tabele: Tabela 6.2, Tabela 6.3. Dla niektórych miar wyniki są niekiedy nawet lepsze dla bardziej zaszumionych obrazów. Rozważając te wyniki trzeba wziąć pod uwagę małe różnice pomiędzy nimi oraz to, że szum generowany jest z użyciem liczb pseudolosowych. Wartość odchylenia standardowego szumu w rozważanym przedziale nie ma znaczącego wpływu na tak zachowujące się miary, a tylko dla miary precyzji widoczny jest znaczący trend, która dla zbioru wartości uzyskanych przy użyciu wygładzania Gaussowskiego z promieniem równym 5 zmniejsza się wraz ze wzrostem odchylenia standardowego szumu. W przypadku wstępnej segmentacji (Tabela 6.3) powyższy trend występuje również dla promienia wygładzania równego 3. Pozwala to stwierdzić, że prezentowany algorytm jest bardziej odporny na zmianę poziomu szumu dla mniejszego rozmycia (promienia wygładzania Gaussowskiego równy 3).

Tabela 6.3 Ewaluacja wyników segmentacji dla wstępnego etapu segmentacji (bez przetwarzania indywidualnych obiektów) gdzie rzeczywista ilość obiektów znajdujących się na obrazie wynosiła 124. Na podstawie [50].

| Promień wygładzania Gaussowskiego | 3 | | 5 | | | |
|-----------------------------------|-------|-------|-------|-------|-------|-------|
| Odchylenie standardowe szumu | 6 | 15 | 30 | 6 | 15 | 30 |
| czułość | 0,994 | 0,995 | 0,997 | 0,999 | 0,999 | 0,999 |
| swoistość | 0,999 | 0,998 | 0,996 | 0,985 | 0,982 | 0,978 |
| precyzja | 0,990 | 0,984 | 0,971 | 0,902 | 0,882 | 0,857 |
| wartość predykcyjna ujemna | 0,999 | 0,999 | 0,999 | 0,999 | 0,999 | 0,999 |
| dokładność | 0,998 | 0,997 | 0,996 | 0,987 | 0,984 | 0,980 |
| Ilość wykrytych obiektów | 124 | 124 | 124 | 124 | 124 | 124 |

W celu otrzymania bardziej miarodajnych wyników algorytm przetestowano również z użyciem losowo przyciętych i przeskalowanych obrazów. Zbiór testowy uzyskano wykorzystując sztucznie stworzony obraz z promieniem wygładzania Gaussowskiego równym 3 i odchyleniem standardowym szumu równym 6, z którego zostały losowo wycięte mniejsze fragmenty (5 fragmentów), a następnie każdy z nich został dodatkowo przeskalowany z użyciem współczynnika 0,7 oraz 1,3 co pozwoliło uzyskać zbiór testowy zawierający 15 obrazów w różnym formacie oraz rozmiarze. Ostatecznie wysokość oraz szerokość obrazów w zbiorze testowym wahała się w granicach od 322 do 1333 pikseli. Wyniki uzyskane na tak przygotowanym zbiorze pozwalają stwierdzić, że proponowany algorytm daje porównywalne wyniki dla obrazów zawierających różnej wielkości erytrocyty (Tabela 6.4).

Tabela 6.4 Ewaluacja wyników segmentacji losowo przyciętych i przeskalowanych obrazów testowych (wartości uśrednione dla wszystkich testowanych próbek) wraz z wynikami dla obrazu testowego w pełnym rozmiarze. Na podstawie [50].

| Typ obrazu | Testowy obraz w pełnym rozmiarze | Przycięte i przeskalowane obrazy |
|----------------------------|-------------------------------------|-------------------------------------|
| czułość | 0,994 | 0,969 |
| swoistość | 0,999 | 0,990 |
| precyzja | 0,990 | 0,930 |
| wartość predykcyjna ujemna | 0,999 | 0,996 |
| dokładność | 0,998 | 0,988 |


Rysunek 6.13 Przykładowy obraz wynikowy. Na wstępnie przefiltrowany obraz oryginalny nałożono kontur (czerwony), oś dłuższą (niebieski) oraz oś krótszą (zielony).



Rysunek 6.14 Przykładowy sztucznie wytworzony obraz (wygładzanie Gaussa - promień równy 3 oraz szum posiadający odchylenie standardowe równe 30).



Rysunek 6.15 Fragmenty sztucznie wytworzonego testowego obrazu z wygładzaniem Gaussa równym 3 - (a), 5 - (b) oraz odchyleniem standardowym szumu równym: 15 - (a), 30 - (b).

6.4 Dyskusja wyników

Statystyczna ewaluacja wyników pokazuje poprawne działanie algorytmu nawet dla obrazów testowych mocno zniekształconych przez szum. Na podstawie wyników (Tabela 6.2, Tabela 6.4), można stwierdzić, że opisywany algorytm pozwala na osiągnięcie dobrych wyników segmentacji erytrocytów dla testowanych obrazów (z mikroskopu optycznego) oraz akceptowalną wrażliwość na zmianę rozmiaru erytrocytów. Tabela 6.3 prezentuje wyniki, które uzyskano na podstawie etapu wstępnej segmentacji bez etapu przetwarzania indywidualnych obiektów. Jako podejście referencyjne przyjęto etap wstępnej segmentacji, ponieważ zastosowana tam kombinacja algorytmów jest popularnym sposobem segmentacji obrazów gdzie występują obiekty łączące się ze sobą. Algorytmy użyte do odjęcia tła i filtracji wstępnej obrazów zostały opisane w rozdziale 3.

Analiza obrazów zawierających erytrocyty jest zadaniem czasochłonnym, a opisany algorytm może zostać użyty w celu jej zautomatyzowania, tam gdzie ich przetwarzanie w czasie rzeczywistym nie jest wymagane, ponieważ duża złożoność obliczeniowa algorytmu wyklucza go z takich zastosowań. Czas przetwarzania dla etapu wstępnej segmentacji zależny jest od rozmiaru obrazu, a czas przetwarzania indywidualnych erytrocytów w drugim etapie zależy dodatkowo od rozmiaru poszczególnych obiektów oraz

ich ilości (Tabela 6.5). Prezentowany algorytm w podstawowej wersji nie jest algorytmem uczącym się więc nie potrzebuje zbioru danych uczących. Rozbudowany zbiór danych uczących jest potrzebny w przypadku użycia algorytmu jako procesor map prawdopodobieństwa pochodzących z konwolucyjnych sieci neuronowych (rozdział 6.2.3) lub w przypadku rozszerzenia algorytmu przez konwolucyjne sieci neuronowe w celu kategoryzacji wyników (rozdział 6.2.4). Modele głębokiego uczenia mają możliwość uzyskania silniejszej generalizacji wraz ze wzrostem rozmiaru uczącego zbioru danych, co pozwala uzyskać lepsze wyniki w przypadku danych nie użytych w procesie uczenia.

Ewaluacja wyników dla podejścia referencyjnego oraz prezentowanego algorytmu pokazuje, że oba podejścia pozwoliły uzyskać segmentację poprawnej ilości obiektów. Rozważając precyzję definiowaną jako stosunek prawdziwie pozytywnych do sumy prawdziwie pozytywnych i fałszywie pozytywnych pikseli jako metrykę ewaluacji, można stwierdzić, że referencyjne podejście okazało się minimalnie lepsze tylko dla dwóch najmniej zaszumionych obrazów (promień wygładzania Gaussa równy 3 oraz odchylenie standardowe szumu równe 6 oraz 15). Dla mocniej zaszumionych obrazów opisywany algorytm daje lepsze wyniki segmentacji.

| Etap algorytmu | Obraz 1 (19 obiektów) | Obraz 2 (10 obiektów) |
|--|-----------------------|-----------------------|
| Czas wstępnej segmentacji [s] | 0,40 | 0,35 |
| Czas segmentacji indywidualnych obiektów [s] | 0,85 | 0,57 |
| Pełny czas przetwarzania (Wstępna segmentacja wraz z przetwarzaniem indywidualnych obiektów) [s] | 1,25 | 0,92 |

Tabela 6.5 Koszt obliczeniowy dla dwóch przykładowych testowych obrazów (256×256 pikseli, głębia bitowa 8 bit) przetwarzanych przy użyciu procesora i5 4670k. Na podstawie [50].

6.5 Podsumowanie

Algorytm opisany w niniejszym rozdziale powstał w celu rozwiązania zadanego problemu segmentacji erytrocytów na obrazach uzyskanych z mikroskopu optycznego oraz wyznaczenia ich rozmiaru i stosunku osi. Na to rozbudowane podejście składa się kombinacja algorytmów przetwarzania obrazów prezentowana w rozdziale 6, która jest mieszanka znanych i szeroko stosowanych metod. Pomimo tego, że metody zastosowane w prezentowanym algorytmie rozważane indywidualnie nie są nowe to ich zastosowanie do dwuetapowego przetwarzania jest nowatorską kombinacją tych metod. Algorytm wyznaczania osi dłuższej i krótszej erytrocytu został opracowany przez autora niniejszej pracy. Algorytm nie jest uniwersalny ze względu na dużą złożoność obliczeniową spowodowaną znaczną ilością użytych technik przetwarzania obrazów składających się na całościowy algorytm. Powoduje to, że algorytm nie nadaje się do stosowania tam, gdzie wymagane jest przetwarzanie w czasie rzeczywistym, nie jest to jednak problemem w zagadnieniu segmentacji erytrocytów gdzie duża szybkość segmentacji nie jest wymagana. Użycie drugiego etapu segmentacji w postaci przetwarzania indywidualnych erytrocytów pozwoliło zwiększyć precyzję segmentacji z 0,857 do 0,968 dla najbardziej zaszumionego sztucznego obrazu testowego (Tabela 6.6).

| Tabela | 6.6 | Podsumowanie | wstępnej | segmentacji | dla | najbardziej | zaszumionego | testowanego | obrazu |
|----------|--------|-------------------|-------------|----------------|------|---------------|------------------|----------------|----------|
| (bez inc | lywid | lualnego przetwa | rzania obie | ektów) – pierv | vszy | etap jako ref | erencja oraz dru | gi etap (indyw | vidualne |
| przetwa | ırzani | e obiektów). Na j | podstawie | [50]. | | | | | |

| Etap | Pierwszy etap segmentacji (referencja) | Drugi etap segmentacji | | |
|----------------|---|------------------------|--|--|
| Precyzja | 0,857 | 0,968 | | |
| Ilość obiektów | 124 | 124 | | |

108

Dwa podejścia przedstawione w rozdziałach 6.2.3 i 6.2.4 są dobrymi przykładami synergii metod klasycznych oraz tych opartych na konwolucyjnych sieciach neuronowych, co pozwala uzyskać dobre rezultaty. Przykładem metody bazującej na sztucznych sieciach neuronowych, która potrzebuje etapu przetworzenia wyników za pomocą klasycznych metod jest model U-net. Wynikiem jego działania jest mapa prawdopodobieństwa, którą trzeba przetworzyć na konkretne obiekty. Ten etap może mieć znaczący wpływ na ostateczny wynik segmentacji. W celu przetworzenia map prawdopodobieństwa uzyskanych z modelu U-net często wykorzystywane są metody takie jak progowanie, czy algorytm wododziałowy połączony z mapą odległości. Ponadto w celu wytrenowania takich modeli wymagane jest zapewnienie wystarczająco dużego zbioru danych z segmentacją przeprowadzoną przez eksperta. Klasyczne metody przetwarzania obrazów mogą wspomóc metody bazujące na sztucznych sieciach neuronowych by uzyskać lepsze wyniki (rozdział 6.2.3) oraz mogą być stosowane tam gdzie nie ma dostępnych odpowiednio dużych zbiorów danych opisanych przez eksperta.

Opisywany algorytm ma pewien potencjał do rozszerzenia w przyszłości poprzez automatyzację procesów wybierania wartości parametrów poszczególnych etapów algorytmu w celu przetwarzania różnorodnych typów obrazów. Takie podejście nie stworzy uniwersalnego algorytmu lecz ma potencjał do rozszerzenia automatycznego działania na pewien ograniczony zbiór typów obrazów wejściowych. Uniwersalne rozwiązanie problemu segmentacji erytrocytów możliwe jest przy zastosowaniu modeli opartych na sztucznych sieciach neuronowych. Przykładem takiej próby jest model SAM opisany w rozdziale 8.3.

7 Opracowanie obrazów pochodzących z mikroskopu sił atomowych w celu uwidocznienia powierzchni erytrocytów

7.1 Wprowadzenie

Metodę obrazowania za pomocą mikroskopu sił atomowych (AFM – ang. *atomic force microscope*) wykorzystuje się do badania powierzchni różnych materiałów. W wyniku wykorzystania tej metody uzyskuje się bardzo dokładny trójwymiarowy skan badanej powierzchni. Znajduje ona zastosowanie również w badaniu próbek biologicznych np. w postaci żywych komórek. Przetwarzane obrazy zawierają erytrocyty z widocznymi agregatami dwutlenku tytanu (TiO_2). Celem algorytmu jest lepsze uwidocznienie tekstury powierzchni erytrocytu. Obrazy źródłowe udostępniła prof. dr hab. Kvetoslava Burda.

Niewątpliwą zaletą mikroskopii sił atomowych jest możliwość obrazowania próbek biologicznych w ich oryginalnym stanie. Obraz uzyskiwany jest poprzez skanowanie powierzchni próbki przy pomocy sondy w postaci dźwigni (ang. cantilever) zakończonej ostrzem pomiarowym (ang. tip) przy jednocześnie prowadzonym pomiarze ugięcia tego ramienia [59]. Ruch ramienia wynika z sił oddziałujących pomiędzy ostrzem sondy, a elementami powierzchni skanowanej próbki [59]. Powstało wiele wariacji trybów skanowania próbki. Przykładem takiego statycznego trybu jest tryb kontaktowy, gdzie ostrze sondy pozostaje w ciągłym kontakcie z próbką i jest od niej odpychane poprzez odziaływania van der Waalsa (odległość w zakresie oddziaływań odpychających) [59]. Innym trybem jest tryb bezkontaktowy gdzie ostrze pozostaje w dalszej odległości od próbki [59]. Sonda wprowadzana jest w oscylacje, a odziaływania van der Waalsa pomiędzy atomami powierzchni badanej próbki oraz atomami powierzchni ostrza sondy (odległość w zakresie oddziaływań przyciągających) powodują zmiany częstotliwości i okresu drgań poprzez przyciąganie ostrza sondy do powierzchni próbki [59]. Pozycjonowanie sondy podczas skanowania próbki możliwe jest dzięki systemowi opartemu na kryształach piezoelektrycznych [59].

Odchylenie sondy mierzone jest za pomocą fotodetektora oraz wiązki lasera skierowanej na górną część sondy, która po odbiciu jest kierowana do fotodetektora [59]. Fotodetektor podzielony jest na cztery części gdzie proporcja pomiędzy sygnałem odbieranym przez części A+B oraz C+D pozwala ocenić wychylenie pionowe ramienia sondy (powodują to zmiany wysokości powierzchni badanej próbki) oraz odchylenie poziome dla części A+C oraz B+D (powoduje je tarcie powstałe pomiędzy ostrzem sondy, a powierzchnią próbki podczas skanowania w trybie kontaktowym) [59].



Rysunek 7.1 Schemat działania mikroskopu sił atomowych.



Rysunek 7.2 Obraz erytrocytu uzyskany przy pomocy techniki mikroskopii sił atomowych z użyciem trybu kontaktowego (a), oraz jego trójwymiarowa reprezentacja z naniesionymi wymiarami (b).

7.2 Opis Algorytmu

Obraz wejściowy jest obrazem w skali szarości o głębi 16-bit oraz rozdzielczości 256 na 256 pikseli. Pierwszym etapem przetwarzania badanych obrazów jest dwukrotne sekwencyjne zastosowanie maski wyostrzającej (Rysunek 2.22). Następnym etapem jest przeprowadzenie odejmowania tła przy pomocy algorytmu toczącej się kuli (ang. *Rolling ball algorithm*) (3.2.4). Kolejnym etapem jest limitowane kontrastem adaptacyjne wyrównanie histogramu (ang. *contrast limited adaptive histogram equalization* - CLAHE) (rozmiar regionu równy 4, ilość możliwych poziomów intensywności równa 256 oraz maksymalne nachylenie równe 7) (3.2.5). Tak otrzymany obraz jest przekształcany na końcu do głębi 8-bit. W ten sposób otrzymujemy obraz z uwydatnioną teksturą powierzchni (Rysunek 7.3 c, d).

W celu wyodrębnienia tekstury powierzchni tworzony jest drugi obraz. Na oryginalnym obrazie wyjściowym przeprowadzone jest filtrowanie w dziedzinie częstotliwości przy pomocy szybkiej transformaty Fouriera (szczegółowy opis metody można znaleźć w [1, 2, 5]) z wykorzystaniem filtra pasmowo przepuszczającego (ang. *band-pass filter*) z ustawieniami pozostawiającymi obiekty większe bądź równe 26 i mniejsze bądź równe 256. Następnie obraz konwertowany jest do głębi 8-bit, a ostatnim krokiem jest przeprowadzenie filtracji filtrem bilateralnym o rozmiarze maski 7 × 7 oraz zakresie w dziedzinie intensywności równym 50 (Rysunek 7.4 a, b).

Następnie tak otrzymany obraz odejmowany jest od obrazu uwydatniającego teksturę powierzchni. Pozwala to otrzymać obraz kładący nacisk na ukazanie tekstury powierzchni (Rysunek 7.4 c, d). Wszystkie etapy przetwarzania zostały przeprowadzone przy pomocy oprogramowania Fiji [52].

7.3 Wyniki

Algorytm pozwala na znaczne polepszenie widoczności tekstury powierzchni erytrocytów. W przypadku badanych obrazów po procesie uwidaczniania szczegółów (Rysunek 7.3 c, d oraz Rysunek 7.4 c, d) agregaty dwutlenku tytanu są znacznie lepiej widoczne niż na obrazach oryginalnych (Rysunek 7.3 a, b). Po zastosowaniu operacji odjęcia dużych obiektów (Rysunek 7.4 a, b) od obrazu z uwydatnioną teksturą (Rysunek 7.3 c, d) otrzymywane są obrazy, które zawierają głównie informacje o szczegółach badanej powierzchni (Rysunek 7.4 c, d). Na badanych obrazach erytrocytów poddanych działaniu dwutlenku tytanu dzięki procesowi uwydatniania tekstury można ponadto zaobserwować, że agregaty dwutlenku tytanu znajdują się wewnątrz komórki. Obserwujemy wypukłości powierzchni erytrocytu spowodowane przez agregaty dwutlenku tytanu, a jednocześnie na tych wypukłościach widoczna jest struktura szkieletu membranowego. Taka obserwacja pozwala postawić tezę, że tlenek tytanu wniknął do wnętrza komórki, gdzie utworzył agregaty. Pokazuje to użyteczność klasycznych algorytmów analizy i przetwarzania obrazów w opracowywaniu obrazów powstałych w trakcie prac badawczych. Obrazy szkieletu membranowego erytrocytu można zaobserwować w publikacji [60].





(a)







(d)

Rysunek 7.3 Obrazy erytrocytów uzyskane przy pomocy techniki mikroskopii sił atomowych (a, b), obrazy z uwydatnioną teksturą (c, d).





(a)









Rysunek 7.4 Obrazy erytrocytów uzyskane przy pomocy techniki mikroskopii sił atomowych zawierające tylko duże obiekty z obrazów oryginalnych (a, b), (c, d) obrazy będące wynikiem odjęcia obrazów (a, b) od obrazów z uwydatnioną teksturą (Rysunek 7.3 c, d).

7.4 Podsumowanie

Zastosowany algorytm przetwarzania obrazów erytrocytów uzyskanych przy pomocy mikroskopu sił atomowych pozwala na uwydatnienie cech powierzchni badanych komórek. Obrazy uzyskane w ten sposób tracą jednak informację o rzeczywistej wysokości badanych obiektów. Uwydatnienie tekstury powierzchni erytrocytów pozwala lepiej obserwować jej zmiany, tak jak w przypadku badanych obrazów, gdzie na powierzchni erytrocytów znajdują się agregaty dwutlenku tytanu. Dzięki zastosowanemu podejściu można lepiej ocenić zmiany powierzchni. Prezentowane podejście powinno również sprawdzić się w innych podobnych zastosowaniach gdzie obszarem zainteresowania jest tekstura powierzchni. Obrazy wynikowe mogą również posłużyć jako obrazy wejściowe do algorytmów opartych na głębokim uczeniu zastosowanych w celu segmentacji zmian widocznych na powierzchni erytrocytów. Wymaga to jednak przygotowania zbioru danych z ręcznym etykietowaniem zmian. Dodatkowo algorytm pozwolił zobrazować jaki wpływ tlenek tytanu ma na erytrocyty. Mianowicie algorytm pozwala zobrazować fakt przedostania się dwutlenku tytanu do wnętrza komórki i utworzenia agregatów poniżej szkieletu membranowego. Widoczna jest struktura szkieletu membranowego na agregatach dwutlenku tytanu co pozwala postawić taką tezę. Dwutlenek tytanu jest szeroko stosowany m.in. w żywności (oznaczany jako E171) oraz w przemyśle farmaceutycznym. Pojawiają się watpliwości co do bezpieczeństwa dwutlenku tytanu w tym zastosowaniu, a szczególnie w formie nanocząsteczek. Europejski Urząd ds. Bezpieczeństwa Żywności stwierdził, że dwutlenku tytanu (E 171) nie można już uznawać za bezpieczny w przypadku stosowania go jako dodatek do żywności m.in. ze względu na obawy dotyczące potencjalnej genotoksyczności. W rozporządzeniu Komisji (UE) 2022/63 z dnia 14 stycznia 2022 r. nakazano stopniowe wycofanie dwutlenku tytanu z zastosowania jako dodatek do żywności oraz poddanie przeglądowi utrzymanie dwutlenku tytanu w unijnym wykazie dodatków do żywności do wyłącznego stosowania jako barwnik w produktach leczniczych bądź usunięcia go z tego wykazu w ciągu trzech lat [61]. W świetle tych watpliwości nie można wykluczyć, że dwutlenek tytanu przenikający do wnętrza erytrocytów może mieć potencjalnie negatywny wpływ na ich funkcjonowanie.

8 Wnioski Końcowe

8.1 Podsumowanie wyników pracy

Niniejsza rozprawa w części teoretycznej zawiera informacje o podstawach związanych z cyfrowym przetwarzaniem i analizą obrazów w zakresie wystarczającym do zrozumienia metod użytych w części doświadczalnej. W części doświadczalnej pracy zaproponowano algorytmy: segmentacji porowatych struktur na obrazach pochodzących z mikrotomografii, erytrocytów na obrazach pochodzących z mikroskopu optycznego oraz algorytm uwidoczniający teksturę powierzchni erytrocytów na obrazach pochodzących z mikroskopu sił atomowych. Każdy z algorytmów operuje na obrazach uzyskanych przy pomocy innej techniki obrazowania. Dają one satysfakcjonujące wyniki opisane szczegółowo w odpowiednich rozdziałach przedstawiających poszczególne algorytmy. Pierwszy z nich dobrze radzi sobie z segmentacja skomplikowanych porowatych struktur na mocno zaszumionych obrazach pochodzących z mikro tomografii, drugi radzi sobie z segmentacją erytrocytów na obrazach mikroskopowych pozwalając na wyliczenie statystyk takich jak np. ilość erytrocytów na obrazie, czy stosunek osi długiej do krótkiej dla każdego erytrocytu. Ostatni prezentowany algorytm pozwala uwidocznić mało widoczne struktury na obrazach erytrocytów pochodzących z mikroskopu sił atomowych co pomaga analizować badane obrazy i jak opisano może prowadzić do ważnych spostrzeżeń jak przykładowo wnikanie dwutlenku tytanu do wnętrza erytrocytu. Wyniki i sposób rozwiązania przedstawionych problemów stanowią również bazę do dalszego rozwijania zaproponowanych podejść.

Zgodnie z celami niniejszej pracy pokazano, że klasyczne algorytmy przetwarzania i analizy obrazów (Rozdziały 5, 6, 7) jak również te wspierane przez sztuczne sieci neuronowe (Rozdział 6) znajdują zastosowanie w rozpatrywanych problemach. W rozdziale 6 opisano przykładowe sposoby wykorzystania algorytmów klasycznych w połączeniu ze sztucznymi sieciami neuronowymi. Potwierdzono również ich skuteczność przedstawiając sposób walidacji wyników segmentacji oparty na sztucznie wygenerowanych obrazach (Rozdziały 5, 6) oraz pokazano polepszanie się jakości segmentacji w przypadku indywidualnego przetwarzania obiektów znajdujących się na obrazie (Rozdziały 5, 6). Pozwala to potwierdzić tezę, że klasyczne metody

segmentacji pozwalają uzyskać satysfakcjonujące wyniki segmentacji w zastosowaniach rozpatrywanych w niniejszej pracy oraz, że klasyczne metody segmentacji mogą stanowić uzupełnienie algorytmów bazujących na głębokim uczeniu i odwrotnie, jak również, że w rozpatrywanych zastosowaniach indywidualne przetwarzanie obiektów znajdujących się na obrazie pozwala polepszyć jakość segmentacji kosztem złożoności obliczeniowej.

8.2 Oryginalne elementy pracy

Oryginalnymi elementami pracy są przedstawione w części doświadczalnej algorytmy. Zawierają one wiele znanych technik analizy i przetwarzania obrazów które zostały w sposób oryginalny połączone ze sobą w celu rozwiązania zadanych problemów. Dla metody segmentacji obrazów erytrocytów pochodzących z mikroskopu optycznego zastosowane algorytmy rozważane indywidualnie nie są nowe lecz zastosowanie dwuetapowego przetwarzania jest nowatorską kombinacją tych algorytmów. Użycie drugiego etapu segmentacji w postaci przetwarzania indywidualnych erytrocytów pozwoliło zwiększyć precyzję segmentacji. Podobnie Algorytm segmentacji porowatych struktur na obrazach uzyskanych przy pomocy mikro-tomografii łączy znane metody w oryginalny dwuetapowy proces segmentacji przetwarzający wyodrębnione w pierwszym etapie obiekty indywidualnie w celu zwiększenia precyzji segmentacji. W przypadku algorytmu uwidaczniającego teksturę powierzchni dla obrazów erytrocytów z agregatami dwutlenku tytanu unikalna jest kombinacja znanych metod z odpowiednio dobranymi parametrami co jako całość daje pożądane rezultaty. W tym przypadku dzięki zastosowanemu algorytmowi ponadto zaobserwowano fakt, że dwutlenek tytanu wnika do wnętrza erytrocytów. Proponowany algorytm uwidocznił szkielet membranowy znajdujący się na agregatach dwutlenku tytanu, więc oznacza to, że dwutlenek tytanu przeniknał do wnętrza erytrocytów. Kolejnym oryginalnym elementem pracy są autorskie rozważania na temat przyszłości i kierunków rozwoju algorytmów analizy i przetwarzania obrazów w tym obrazów medycznych przedstawione w ostatnim rozdziale uzupełniające rozprawę o spojrzenie biorące pod uwagę bardzo szybki rozwój algorytmów opartych na sztucznych sieciach neuronowych (patrz rozdział 8.3).

119

8.3 Rozważania na temat przyszłości oraz kierunków rozwoju algorytmów analizy i przetwarzania obrazów w tym obrazów medycznych

Klasyczne algorytmy cyfrowego przetwarzania i analizy obrazów przez wiele lat stanowiły podstawowe narzędzie rozwiązywania problemów w tej dziedzinie. Pozwalały one efektywnie rozwiązywać problemy przy istniejących ograniczeniach mocy obliczeniowej ówczesnych komputerów. Takie metody mają swoje ograniczenia. Algorytm dopasowany do konkretnego problemu jest wrażliwy na zmiany danych wejściowych, ponieważ wynik zależy od wielu parametrów dobranych do rozwiązania konkretnego zadania. Przykładowo chcąc stworzyć algorytm segmentujący kilka rodzajów komórek na zdjęciach pochodzących z mikroskopu optycznego ciężko uzyskać dobre wyniki z wykorzystaniem algorytmów klasycznych. Algorytmy uczące pozwalają lepiej generalizować wyniki i automatycznie dopasowują się do różnorodnych danych wejściowych podobnych do tych występujących w zbiorze uczącym. W przypadku algorytmów klasycznych trzeba je rozbudowywać o kolejne reguły dopasowujące parametry do innych danych wejściowych co staje się tym trudniejsze im problem jest bardziej skomplikowany, aż staje się to praktycznie niemożliwe. Szerszy opis różnic w rozdziale 4.6.1.

Jedną z głównych przyczyn nagłego rozwoju modeli opartych na głębokim uczeniu w ostatnich latach jest rozwój mocy obliczeniowej komputerów, również tych dostępnych dla przeciętnego człowieka co pociągnęło za sobą rozwój całej dziedziny. Modele wykorzystujące koncepcję sztucznych sieci neuronowych mogły w ostatnich latach przejść z poziomu teorii oraz prostych modeli stosowanych do rozwiązania mało skomplikowanych problemów do coraz większych sieci neuronowych co pozwoliło skutecznie zastosować je do rozwiązywania bardziej złożonych problemów. Wynika to z samej natury działania modeli opartych na sztucznych sieciach neuronowych, które potrzebują bardziej pojemnych modeli (większa ilość parametrów) do zaabsorbowania wiedzy potrzebnej do rozwiązania bardziej skomplikowanych problemów. W celu wizualizacji pojemności modelu można przeprowadzić prosty eksperyment polegający na przetestowaniu kilku modeli o różnej ilości perceptronów w warstwie ukrytej oraz o różnej ilości warstw ukrytych na kilku sztucznie stworzonych zbiorach danych z różną ilością reguł odpowiadających za ich stworzenie. Jednocześnie w celu pokazania obecnie

istniejących możliwości algorytmów opartych na głębokim uczeniu do wygenerowania kodu w języku Python został użyty duży model językowy, który jest ogólnodostępny dla każdego poprzez interfejs czatu umieszczony w wyszukiwarce internetowej Bing (pierwotnie znany pod nazwa Bing Chat, a obecnie przemianowany na Microsoft Copilot) [62]. Należy zwrócić uwage że obecna wersja tego modelu świetnie radzi sobie z obsługa poleceń w wielu językach, również w języku polskim. Pierwszym etapem jest stworzenie skryptu w języku Python, który wygeneruje 10000 wierszy z danymi wejściowymi w postaci zbioru parametrów wejściowych oraz jednej wartości wyjściowej 0 bądź 1, gdzie 1 oznacza erytrocyt zdrowy, a 0 erytrocyt nieprawidłowy/uszkodzony. Po wprowadzeniu w interfejsie tekstowym następującej podpowiedzi: "Napisz program w języku python, który wygeneruje 10000 wierszy gdzie każdy wiersz zawiera 10 wartości bedacych losowymi liczbami zmiennoprzecinkowymi w zakresie od 0 do 1 z wyjątkiem ostatniej w kolumnie o nazwie healthy gdzie będzie to losowa liczba 0 lub 1. Nazwy kolumn dla tych wartości to: color, roundness, edge smoothness, feature1, feature2, feature3, feature4, feature5, feature6, healthy.", model LLM (ang. Large Language Model) zwraca odpowiedź w postaci gotowego skryptu języka Python wraz z krótkim objaśnieniem jego działania (Rysunek 8.1). Jak można zauważyć wygenerowany kod robi to co powinien ale nie zapisuje wyniku do pliku csv. Po dodaniu kolejnej podpowiedzi precyzującej oczekiwania o treści: "wynik działania programu powinien zostać zapisany do pliku csv o nazwie data", model LLM jest w stanie poprawić wynik (Rysunek 8.2). Należy zwrócić uwagę, że jest to bardzo naturalny sposób interakcji z modelem LLM przypominający rozmowę z innym człowiekiem. Następnie w celu stworzenia kilku zbiorów danych z różna ilością reguł oznaczania erytrocytu jako nieprawidłowy/uszkodzony podano kolejną podpowiedź: "Napisz program w języku python, który wczyta stworzony wcześniej plik csv o nazwie data i jeżeli dla danego wiersza wartość kolumny edge smoothness jest mniejsza niż 0.5 to ustawi wartość dla kolumny kolumny healthy jako 0" oraz "plik wynikowy powinien mieć nazwę data 1 condition" otrzymując nowy skrypt (Rysunek 8.3). Modele LLM radzą sobie również z nie do końca poprawnie gramatycznie sformułowanymi podpowiedziami zawierającymi literówki czy jak w komendzie powyżej powtórzenie wyrazu kolumny. Jak można zauważyć model LLM udostępniony przez przeglądarkę Bing ma dostęp do internetu i przeszukuje go po hasłach stworzonych na podstawie zapytania użytkownika (Rysunek 8.3). Na końcu podaje również odnośniki do materiałów na podstawie których udzielił odpowiedzi (Rysunek 8.3). Pozwala to zweryfikować otrzymaną odpowiedź. Ponieważ modele LLM udzielają odpowiedzi statystycznie prawdopodobnych, a nie koniecznie prawdziwych zawsze należy weryfikować odpowiedź. Przykładowo jeżeli ktoś generuje algorytm komputerowy z wykorzystaniem modelu LLM to dobrze aby sam posiadał umiejętność programowania i był w stanie zweryfikować odpowiedź czy przykładowo wygenerowany kod uruchomiony na komputerze użytkownika nie usunie ważnych plików. Korzystając z gotowego szablonu kodu wygenerowanego przez model LLM uzupełniono go o kod pisany ręcznie i stworzono algorytm zwracający sześć zbiorów danych gdzie każdy posiada kolejno coraz większą liczbę warunków. Algorytm wykorzystuje plik z losowymi danymi otrzymany w pierwszym kroku aby każdy zbiór danych miał taki sam zestaw losowych parametrów wejściowych. Na początku wszystkie wartości wynikowe ustawiane są na wartość 1, a każdy kolejny warunek w przypadku jego spełnienia ustawia tę wartość jako 0 (Rysunek 8.4).

Oto program w języku Python, który generuje 10000 wierszy, gdzie każdy wiersz zawiera 10 wartości będących losowymi liczbami zmiennoprzecinkowymi w zakresie od 0 do 1 z wyjątkiem ostatniej kolumny o nazwie healthy, gdzie będzie to losowa liczba 0 lub 1. Nazwy kolumn dla tych wartości to: color, roundness, edge_smoothness, feature1, feature2, feature3, feature4, feature5, feature6, healthy.

```
0
import random
for i in range(10000):
    color = round(random.uniform(0, 1), 2)
    roundness = round(random.uniform(0, 1), 2)
    edge_smoothness = round(random.uniform(0, 1), 2)
    feature1 = round(random.uniform(0, 1), 2)
   feature2 = round(random.uniform(0, 1), 2)
    feature3 = round(random.uniform(0, 1), 2)
   feature4 = round(random.uniform(0, 1), 2)
    feature5 = round(random.uniform(0, 1), 2)
   feature6 = round(random.uniform(0, 1), 2)
    healthy = random.randint(0, 1)
    print(color,
          roundness,
          edge_smoothness,
          feature1,
          feature2.
          feature3,
          feature4,
          feature5,
          feature6,
          healthy)
```

Rysunek 8.1 Skrypt języka Python wygenerowany przez model LLM, generujący losowe dane w zadanym formacie.

Oto program w języku Python, który generuje 10000 wierszy, gdzie każdy wiersz zawiera 10 wartości będących losowymi liczbami zmiennoprzecinkowymi w zakresie od 0 do 1 z wyjątkiem ostatniej kolumny o nazwie healthy, gdzie będzie to losowa liczba 0 lub 1. Nazwy kolumn dla tych wartości to: color, roundness, edge_smoothness, feature1, feature2, feature3, feature4, feature5, feature6, healthy.

```
0
import random
import csv
with open('data.csv', mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['color', 'roundness', 'edge smoothness', 'feature1', 'feature2',
'feature3', 'feature4', 'feature5', 'feature6', 'healthy'])
    for i in range(10000):
        color = round(random.uniform(0, 1), 2)
        roundness = round(random.uniform(0, 1), 2)
        edge smoothness = round(random.uniform(0, 1), 2)
        feature1 = round(random.uniform(0, 1), 2)
        feature2 = round(random.uniform(0, 1), 2)
        feature3 = round(random.uniform(0, 1), 2)
        feature4 = round(random.uniform(0, 1), 2)
        feature5 = round(random.uniform(0, 1), 2)
        feature6 = round(random.uniform(0, 1), 2)
        healthy = random.randint(0, 1)
        writer.writerow([color,
                          roundness,
                          edge_smoothness,
                          feature1,
                          feature2,
                          feature3,
                          feature4,
                          feature5,
                          feature6,
                          healthy])
```

Program zapisze wynik działania do pliku csv o nazwie data.csv . Mam nadzieję, że to Ci pomogło!

Rysunek 8.2 Skrypt języka Python wygenerowany przez model LLM, generujący losowe dane w zadanym formacie oraz zapisujący wynik do pliku csv.

plik wynikowy powinien mieć nazwę data_1_condition

- Wyszukiwanie dla: how to rename csv file in python
- Wyszukiwanie dla: python program rename csv file
- Vyszukiwanie dla: python program rename csv file data.csv data_1_condition
- Wyszukiwanie dla: python program read csv file and modify healthy column
- Wyszukiwanie dla: python program read csv file and modify healthy column data_1_condition
- Trwa generowanie odpowiedzi dla Ciebie...

Oto program w języku Python, który wczyta plik csv o nazwie data.csv i jeśli wartość kolumny edge_smoothness jest mniejsza niż 0.5, to ustawi wartość dla kolumny healthy jako 0. Następnie program zapisze zmodyfikowany plik do data_1_condition.csv.

```
import csv
 with open('data.csv', mode='r') as file:
     reader = csv.DictReader(file)
     rows = []
     for row in reader:
         if float(row['edge_smoothness']) < 0.5:</pre>
             row['healthy'] = 0
         rows.append(row)
 with open('data_1_condition.csv', mode='w', newline='') as file:
     writer = csv.DictWriter(file, fieldnames=reader.fieldnames)
     writer.writeheader()
     writer.writerows(rows)
Program wczyta plik data.csv , a następnie przeiteruje po każdym wierszu. Jeśli wartość
kolumny edge_smoothness jest mniejsza niż 0.5, to wartość dla kolumny healthy
zostanie ustawiona na 0. Na końcu program zapisze zmodyfikowany plik do
data_1_condition.csv . Mam nadzieję, że to Ci pomogło!
```

 Dowiedz się więcej:
 1. stackoverflow.com
 2. stackoverflow.com
 +12 więcej
 4 z 30 •

Rysunek 8.3 Skrypt języka Python wygenerowany przez model LLM zmieniający dane wynikowe odczytane z pliku csv. Użytkownik otrzymuje również odnośniki do danych źródłowych gdzie może zweryfikować odpowiedź.

```
for row in rows:
    row['healthy'] = 1
    # Warunek 6
    if row['feature5'] * row['feature5'] + 1.9*row['feature2'] < row['feature6']:</pre>
        row['healthy'] = 0
   # Warunek 5
   if row['edge_smoothness'] < 0.5 and row['roundness'] < 0.4 and row['color'] < 0.5:
        row['healthy'] = 0
   # Warunek 4
   if row['feature1'] * row['feature2'] + 0.2 < row['feature3'] * row['feature4']:</pre>
        row['healthy'] = 1
    # Warunek 3
    if row['edge smoothness'] < 0.2:</pre>
        row['healthy'] = 0
   # Warunek 2
   if row['roundness'] < 0.1:</pre>
        row['healthy'] = 0
    # Warunek 1
    if row['color'] < 0.1:</pre>
        row['healthy'] = 0
```

Rysunek 8.4 Uproszczony fragment kodu w języku Python pokazujący warunki do oznaczenia erytrocytu jako nieprawidłowy/uszkodzony.

W następnym kroku wykorzystano model LLM do wygenerowania szkieletu skryptu do przeprowadzenia procesu trenowania sztucznej sieci neuronowej na zbiorze uczącym i przetestowania na zbiorze testowym. Następnie ręcznie zmodyfikowano wygenerowany kod w celu uruchomienia tej procedury dla wszystkich kombinacji zbiorów danych oraz modeli sztucznych sieci neuronowych. Dla każdej z tych kombinacji proces trenowania i testowania powtórzono trzykrotnie, a następnie wyznaczono średnią dokładność dla zadanej kombinacji oraz ilości iteracji w fazie trenowania. Średnią z trzech prób wyznaczono w celu zniwelowania fluktuacji wyników związanej z elementem losowości wbudowanym w proces trenowania sztucznej sieci neuronowej. Wyniki przedstawiono na wykresie typu mapa ciepła (ang. *Heatmap*) (Rysunek 8.5).



Rysunek 8.5 Wykresy typu mapa ciepła przedstawiające dokładność otrzymanego wyniku (ang. *Accuracy*) dla kombinacji poszczególnych zbiorów danych i modeli.

Wykonany eksperyment pozwala stwierdzić, że już obecnie modele typu LLM stanowią narzędzie zwiększające produktywność zdolne do generowania kodu źródłowego na podstawie naturalnej interakcji z człowiekiem za pomocą interfejsu przypominającego chat. Wyniki potwierdzają również powszechnie znany fakt, że ilość wiedzy (pojemność modelu) którą można umieścić w modelu podczas procesu jego trenowania wzrasta

z ilością sztucznych neuronów w warstwie ukrytej. Zbyt prosty model nie jest w stanie pomieścić wiedzy o bardziej złożonych problemach. Można zauważyć również ciekawe zjawisko, że dla zbioru danych zawierającego losowe dane im bardziej złożony model zastosujemy tym bardziej dokładność jego przewidywania wzrasta od okolic 0.50 co jest wartością oczekiwaną dla losowego rozkładu danych do wartości 0.56. Może to sugerować, że bardziej skomplikowany model był w stanie znaleźć jakieś prawidłowości w danych uzyskanych z generatora liczb pseudolosowych w języku python. Wynik powtarza się dla kolejnych wygenerowanych zbiorów danych co sugeruje, że nie są to jakieś spontanicznie powstałe przypadkowe reguły w konkretnym zbiorze danych ale raczej uchwycenie właśnie tej pseudolosowości samego generatora liczb losowych.

Wykorzystywanie dużych modeli językowych do programowania może zwiększyć wydajność ale nie uwalnia to programisty od potrzeby opanowania programowania do poziomu pozwalającego zrozumieć wygenerowany kod. Brak możliwości zrozumienia wygenerowanego kodu prowadzi do utraty kontroli nad działaniem programu i może prowadzić do negatywnych skutków. Przykładowo nikt odpowiedzialny nie użyje wygenerowanego programu, który nie został sprawdzony przez człowieka potrafiącego zrozumieć ten kod do zarządzania ważnym procesem bądź do obliczeń na podstawie, których podejmowane są ważne decyzje.

Podczas fali nagłego rozwoju metod opartych na sztucznych sieciach neuronowych trwającej w 2023 roku zaprezentowano również model o nazwie SAM (ang. *Segment Anything Model*) służący do segmentacji obrazów, który pretenduje do stania się modelem fundamentalnym (ang. *Foundation Model*) [63]. Jest to model stworzony przez przedsiębiorstwo Meta. Model pozwala przeprowadzać segmentację zarówno w sposób interaktywny (użytkownik wyznacza obszary do segmentacji w postaci punktów, prostokątnego zaznaczenia obszaru bądź podpowiedzi tekstowej) jak i w pełni automatyczny. Model został wytrenowany na zbiorze danych SA-1B zawierającym ponad miliard masek uzyskanych z około 11 milionów obrazów. Tak ogromny zbiór danych byłby bardzo trudny do uzyskania przy użyciu ręcznego etykietowania wykorzystano więc sam trenowany model w procesie etykietowania zbioru danych (ang. *model-in-the-loop*) w celu

128

zautomatyzowania tego procesu. W pierwszym etapie model wspomagał ludzi etykietujących dane, następnie etykietował dane półautomatycznie (człowiek dodawał etykiety których model nie wykrył automatycznie), a ostatnim etapem było etykietowanie całkowicie automatyczne. Model pozwala przeprowadzać segmentację na kilka sposobów. Jedna z możliwości jest segmentacja wspomagana przez człowieka gdzie użytkownik wyznacza punkty lub prostokątne ROI na obrazie, a model przeprowadza segmentację interesujących nas obiektów. Planowane jest również wsparcie dla podpowiedzi tekstowych. Obecnie istnieje kilka rozwiązań pozwalających osiągnąć segmentację na podstawie podpowiedzi tekstowych ale są to projekty w początkowej fazie rozwoju. Przykładami sa: Alpha-CLIP [64] i wykorzystanie go razem z modelem SAM oraz projekt Grounded-SAM [65] wykorzystujący model Grounding DINO [66] w połączeniu z modelem SAM. Drugim trybem jest segmentacja automatyczna która polega na wygenerowaniu równomiernej siatki punktów i przeprowadzeniu automatycznej segmentacji całego obrazu. Ten drugi tryb jest najbardziej interesujący w kontekście porównania wyników z tymi uzyskanymi przez proponowane w niniejszej rozprawie algorytmy. Model SAM jest dostępny publicznie jednak obserwowane są różnice w wynikach segmentacji automatycznej dla modelu dostępnego na stronie projektu dostępnej na platformie GitHub [67] oraz demonstracyjnej aplikacji internetowej dostępnej online, co w połączeniu z problemami zgłaszanymi przez użytkowników na stronie projektu [67] pozwala przypuszczać, że model użyty w aplikacji demonstracyjnej wykorzystuje inne wartości parametrów modelu oraz sam model, a przynajmniej jego fragmenty zostały ulepszone w porównaniu do modelu dostępnego publicznie. Wyniki uzyskane po zastosowaniu modelu SAM dla obrazów porowatych struktur (Rysunek 8.6) pokazują, że publicznie dostępny model SAM vit_h działający w trybie automatycznej segmentacji z domyślnymi parametrami nie daje satysfakcjonujących wyników. Model SAM dostępny przez demonstracyjna aplikację internetową daje lepsze wyniki ale również nie idealne. Wyniki uzyskane przez oba wymienione modele są słabsze niż wyniki algorytmu opisanego w niniejszej rozprawie (rozdział 5) dla tego konkretnego problemu. Dla kolejnego rozpatrywanego przykładu w postaci obrazu zawierającego erytrocyty pochodzącego z mikroskopu optycznego wyniki dla publicznie dostępnego modelu SAM vit h z wykorzystaniem automatycznej segmentacji z domyślnymi parametrami są znacznie

lepsze niż w poprzednim przykładzie lecz nadal są gorsze od wyników uzyskanych w wyniku działania algorytmu opisanego w niniejszej rozprawie w rozdziale 6 (Rysunek 8.7). Model SAM dostępny przez demonstracyjną aplikację internetową daje jednak świetne wyniki. Biorąc pod uwagę, że prezentowany w niniejszej pracy algorytm jest zależny od zestawu parametrów, a model SAM do uzyskania poprawnej segmentacji nie potrzebował ustawiania żadnych parametrów to jest on lepszym, bardziej uniwersalnym rozwiązaniem. Wraz z rozwojem tego typu ogromnych modeli coraz więcej problemów związanych z segmentacja będzie przez nie rozwiązywane, a coraz mniej problemów będzie wymagało modeli wyspecjalizowanych pod konkretne zadanie. Rozwiązanie takich problemów będzie również ułatwione dzięki wykorzystaniu takich modeli jak SAM jako wsparcie w procesie etykietowania obrazów. Pozwoli to sprawniej tworzyć zbiory danych pozwalające trenować wyspecjalizowane modele bądź przeprowadzić dostrojenie (ang. fine-tunning) dużego modelu w celu osiagniecia możliwości segmentacji automatycznej. Dzięki modelom fundamentalnym będzie możliwe również automatyczne generowanie zbiorów uczących, które ułatwią proces uczenia mniejszych modeli podobnie jak to ma miejsce w przypadku modeli LLM gdzie modele fundamentalne są często używane do generowania danych uczących. Wprowadza to pewną demokratyzację dziedziny, ponieważ zebranie dużych zbiorów danych jest czasochłonne i kosztowne, a automatyczne etykietowanie czy generowanie syntetycznych danych za pomocą modeli fundamentalnych znacznie upraszcza ten proces oraz zmniejsza jego koszty.



Rysunek 8.6 Wyniki segmentacji fragmentu obrazu mikro-tomograficznego porowatej struktury w postaci piany polimerowej uzyskane po automatycznej segmentacji modelem SAM vit_h (a) oraz modelem SAM dostępnym poprzez internetową aplikację demonstracyjną (b).



Rysunek 8.7 Wyniki segmentacji obrazu zawierającego erytrocyty pochodzącego z mikroskopu optycznego uzyskane po automatycznej segmentacji modelem SAM vit_h (a) oraz modelem SAM dostępnym poprzez internetową aplikację demonstracyjną (b).

Modele będące ogólną sztuczną inteligencją zwaną skrótowo AGI (ang. artifical general *intelligence*) prawdopodobnie będą mogły rozwiązywać bardzo skomplikowane problemy również z dziedziny analizy i przetwarzania obrazu. Wydaje się, że zarówno modele uznawane za AGI czy jakieś ich stadia pośrednie mogłyby korzystać z wielu mniejszych modeli specjalizowanych w konkretnych zadaniach np. model konwolucyjny znajdujacy wzorce na obrazie czy interfejs użytkownika oparty na dużych modelach językowych (LLM). Koncept zmierzający w podobnym kierunku zrealizowano tworząc bardzo dobry model Mixtral 8x7b wykorzystujący architekturę mieszaniny ekspertów (ang. Mixture of Experts), która polega na tym, że każda warstwa jest skomponowana z ośmiu bloków, gdzie każdy blok jest oddzielnym ekspertem o odmiennej specjalizacji [68]. Dla każdego przetwarzanego tokena na poziomie każdej warstwy wybieranych jest dwóch z ośmiu ekspertów do przetworzenia obecnego stanu [68]. W rezultacie każdy token ma dostęp do całego modelu liczącego 47 miliardów parametrów ale sama inferencja (ang. Inference) wykorzystuje jedynie 13 miliardów aktywnych parametrów [68]. Wykorzystanie tylko części parametrów dostępnego modelu per token pozwala uzyskać znacznie szybsze działanie modelu [68]. Mixtral 8x7b pokonał w wielu zadaniach znacznie większe modele takie jak LLAMA 2 70B, GPT-3.5 Turbo czy Gemini Pro [68]. Podobna idea w innym wydaniu jest obecnie zaimplementowana na poziomie kooperacji różnych wyspecjalizowanych modeli LLM, również z możliwością używania przez te modele funkcji zdefiniowanych przez użytkownika w postaci rozwiązań określanych jako LLM następnej generacji (ang. Next-Gen LLM) jak np. AutoGen [69]. Taka struktura wykazuje w jakiś sposób podobieństwo do ludzkiego mózgu gdzie różne jego obszary specjalizują się w różnych zadaniach. Ta specjalizacja obszarów mózgu w połączeniu z obserwowaną neuroplastycznością, która pozwala przykładowo na przejmowanie funkcji jednych obszarów przez drugie np. w przypadku uszkodzenia fragmentu mózgu jest bardzo ciekawym fenomenem [70]. Korzystanie z takiego systemu polegałoby na zasileniu go dużymi zbiorami danych. Przykładowo medycznymi danymi pacjentów zawierającymi dane otrzymane z różnych modalności: tekstowe (diagnozy, zalecenia, przepisane leki, wyniki badań krwi, moczu itd.), obrazowe (skany CT, skany MRI, zdjęcia RTG, zdjęcia/filmy z USG), liczbowe (wyniki różnych badań, dane z noszonych urządzeń – holtery, zegarki sportowe z pomiarem tętna, utlenowania krwi czy jakości snu,

dane lokalizacyjne (gdzie pacjent żyje, przebywa, jaki kraj odwiedził ostatnio)). Taki system zasilony danymi wielu pacjentów znalazłby powiązania pomiędzy czynnikami ryzyka, które są tak odległe od siebie i na pierwszy rzut oka nieskorelowane, że znalezienie tak złożonych wzorców jest niedostępne dla człowieka. Pozwoliłoby to również stawiać diagnozy nawet bardzo rzadkich i trudnych do zdiagnozowania chorób czy przewidywać je z dużym wyprzedzeniem zanim choroby się rozwiną do stanu dającego bardzo widoczne objawy. Taki wirtualny asystent pozwoliłby zniwelować różnice w poziomie wykształcenia czy indywidualnych predyspozycjach lekarzy i zniwelowałby problem, polegający na tym, że lekarz specjalizujący się w swojej dziedzinie nie jest w stanie zauważyć problemów pacjenta za które odpowiedzialna jest inna dziedzina. Czas wizyty u lekarza jest również ograniczony, a człowiek nawet jeżeliby miał dostęp do całej historii leczenia i wszystkich badań pacjenta ma ograniczone możliwości percepcji więc nie jest w stanie przyswoić, oraz przeanalizować ogromnych ilości danych. Taki wirtualny asystent byłby świetnym podpowiadającym lekarzowi diagnozę w trakcie wizyty, sugerującym narzędziem na co zwrócić uwagę czy jakie dodatkowe badania wykonać. Pozwoliłoby to na szeroki dostęp do diagnostyki na wysokim, a przede wszystkim równym poziomie niezależnie od miejsca zamieszkania pacjenta. Oczywiście zgromadzenie tak wielkiego zbioru danych o pacjentach rodzi problemy w postaci ochrony prywatności pacjenta. Takie dane musiałyby być szczególnie chronione przed niepowołanym czy zbyt szerokim dostępem. Przykładowo tego typu dane mogłyby być łakomym kąskiem dla firm ubezpieczeniowych, które mogłyby na ich podstawie odmawiać ubezpieczenia zdrowotnego czy różnicować jego wysokość.

Do tej pory rozwiązania oparte na sztucznych sieciach neuronowych pozostawały w cieniu. Często ludzie korzystali z tych modeli nie będąc tego nawet świadomym. Przykładowo wykonywanie zdjęć z wykorzystaniem smartfona jest powszechną czynnością, a większość użytkowników nie zdaje sobie sprawy, że zdjęcia te są bardzo często przetwarzane przez modele oparte na sztucznych sieciach neuronowych [71, 72, 73]. Niniejszy akapit powstaje w październiku 2023-go roku. Jest to około 10 miesięcy po rewolucji w dziedzinie sztucznej inteligencji, która dokonała się w niespotykanym dotąd wysokim tempie. Za początek rewolucji można przyjąć opublikowanie 30-go listopada 2022-go roku przez przedsiębiorstwo OpenAI modelu LLM (dużego modelu językowego) połączonego

z interfejsem użytkownika w postaci czatu pod nazwą ChatGPT [74]. Właśnie w tym przyjaznym interfejsie użytkownika dostępnym również dla ludzi nie będących specjalistami w dziedzinie sztucznej inteligencji czy programowania można upatrywać przyczyny ogromnego sukcesu i zaistnienia w świadomości przeciętnego człowieka. Skokowy wzrost użytkowników nie miał wcześniej precedensu. ChatGPT zyskał milion użytkowników w ciągu pięciu dni. Dla porównania dla innych usług online czas ten mierzony był w miesiącach bądź latach. Kilka miesięcy wcześniej zadebiutowała inna usługa online z obszaru generatywnej sztucznej inteligencji (ang. Generative artificial intelligence) oparta na sztucznych sieciach neuronowych o nazwie Midjourney [75] stworzona przez przedsiębiorstwo o tej samej nazwie. Usługa ta pozwala na generowanie obrazów za pomocą podpowiedzi tekstowych (ang. text prompt). Udostępnienie usługi Midjourney również przebiło się do świadomości ludzi niezwiązanych z branżą informatyczną. Obrazy generowane za pomocą tego narzędzia stały się trendem głośnym w internecie. Powstała również nowa umiejętność zwana inżynierią podpowiedzi (ang. prompt engineering) polegająca na konstruowaniu podpowiedzi tekstowych tak aby uzyskać oczekiwany wynik działania zastosowanego modelu. Podpowiedzi tekstowe służą do sterowania zarówno modelami typu tekst do tekstu (ang. text to text) jak ChatGPT oraz tekst do obrazu (ang. text to image) jak Midjourney. Wszystkie te rozwiązania bazują na architekturze transformerów (ang. transformers) opartej na mechanizmie uwagi (ang. Attention) co zostało opisane w przełomowej i wielokrotnie cytowanej pracy pod obrazowym tytułem "Attention Is All You Need" [76]. Pojawienie się tych narzędzi jak również ich bardzo szybkie tempo rozwoju wywołało wręcz szok. Spowodowało to proliferacje podobnych rozwiązań, ich zastosowań jak również startupów upatrujących swojej szansy na fali popularności sztucznej inteligencji. Po kilku miesiącach euforia związana z generatywną sztuczną inteligencją opadła lecz ten obszar rozwija się nadal w ekspresowym tempie. Przykładowo modele typu LLM nie sprawdzają się dobrze w zadaniach matematycznych wymagających precyzji. Dają one odpowiedzi statystycznie prawdopodobne, które nie koniecznie są poprawne. ChatGPT został rozszerzony o mechanizm wtyczek pozwalający przezwyciężyć tego typu problemy. Jednym z przykładów jest wtyczka Wolfram [77], która rozszerza możliwości ChatGPT o dokładne obliczenia oparte o narzędzie Wolfram Alpha przy jednoczesnym zachowaniu bardzo naturalnego w obsłudze interfejsu ChatGPT. Oprogramowanie oferowane przez OpenAI doczekało się również możliwych do konfiguracji przez użytkownika modeli pozwalających wyspecjalizować się im w konkretnych zadaniach. Takie konfiguracje są następnie dostępne dla twórcy badź opcjonalnie udostępniane publicznie co pozwala używać ich innym użytkownikom modelu ChatGPT. We wrześniu 2023-go roku zadebiutował szkielet aplikacyjny (ang. framework) AutoGen tworzony przez firmę Mircrosoft, który ma za zadanie ułatwić tworzenie aplikacji LLM, które używają wielu agentów (ang. Agents) LLM konwersujących ze sobą w celu rozwiązania zadanego problemu. Można w ten sposób tworzyć kombinacje interakcji z ludzkim użytkownikiem, modeli LLM oraz narzędzi. Można również tworzyć skomplikowane topologie z użyciem tych elementów składowych. Twórcy udostępnili również narzędzie AutoGen Studio, które udostępnia interfejs graficzny. Jest to kierunek, który moim zdaniem zrewolucjonizuje sposób rozwiązywania problemów, również takich jak przetwarzanie i analiza obrazów cyfrowych. Będą powstawać bardzo skomplikowane systemy oparte na wykorzystaniu wielu zbiorów danych, narzędzi, modeli LLM, modeli CNN czy ogólniej modeli ANN ukierunkowanych na rozwiązanie konkretnych problemów, które działając razem będą bardzo potężnym narzędziem, a użytkownik będzie je wykorzystywał poprzez interfejs wykorzystujący język naturalny. Obecnie najpopularniejszą formą jest komunikacja za pomocą tekstu wpisywanego w oknie czatu ale prawdopodobnie komunikacja za pomocą głosu będzie stawać się bardziej popularna. Interakcja będzie przebiegała wtedy na zasadzie rozmowy z asystentem. Możliwości techniczne obu sposobów komunikacji istnieją już obecnie. Jedną z przyczyn dominacji interfejsu tekstowego są prawdopodobnie przyzwyczajenia użytkowników. Oba sposoby komunikacji maja jedną zasadniczą wadę polegającą na stosunkowo niskiej przepustowości wymiany danych. Obecnie toczą się również intensywne badania mające na celu stworzyć interfejs mózg-komputer, który pozwoliłby przezwyciężyć to ograniczenie. Przedsiębiorstwo Neuralink prowadzi intensywne badania w tej dziedzinie [78]. Obecnie uzyskana została zgoda na prowadzenie badań na ludziach i pierwszy pacjent został już poddany zabiegowi wszczepienia implantu przedsiębiorstwa Neuralink. Stworzenie takiego działającego sprawnie interfejsu spowoduje rewolucję o trudnych do przewidzenia skutkach ale zapewne zrewolucjonizuje sposób interakcji z modelami sztucznej inteligencji tworzac jednocześnie cała paletę nowych nieznanych wcześniej zagrożeń jak wpływanie na proces myślowy człowieka poprzez ten interfejs czy zbytnie uzależnienie funkcjonowania człowieka od sztucznych rozszerzeń. Rozkwita również obszar modeli udostępnianych na otwartych lub częściowo otwartych licencjach (ang. Open Source) jak przykładowo Llama 2 [79]. Obecnie udostępniane są przeważnie modele o rozmiarze 7, 13 bądź 70 miliardów parametrów. Dzięki możliwości wczytywania modelu częściowo do pamieci karty graficznej, a częściowo do pamięci RAM oraz kwantyzacji modeli można je uruchamiać już na prywatnych komputerach. Przykładowo komputer z procesorem i7-13700K, kartą graficzną GeForce RTX 3090 TI 24 GB oraz 64 GB pamięci RAM typu DDR5 pozwala uruchamiać lokalnie małe i średnie modele mieszczące się w pamięci karty graficznej, które będą odpowiadały szybko. Można również uruchomić te duże modele ale nie będą one odpowiadać już tak szybko ponieważ tylko część modelu będzie wczytana do pamięci karty graficznej. Dzięki udostępnianiu takich modeli, których wytrenowanie to koszt rzędu kilku milionów dolarów społeczność naukowców i pasjonatów może z nimi eksperymentować. Osiągają one świetne wyniki jak przykładowo model Mixtral 8x7B [68], czy model Code Llama 70b, który przez kilka tygodni pierwszego kwartału 2024 r. był najlepszym modelem fundamentalnym w zakresie tworzenia kodu i pokonał w tym zakresie również modele o zamkniętej licencji [80]. Palmę pierwszeństwa niespodziewanie przejął po nim komercyjny model Claude 3 ujawniony 4 marca 2024 r., który znacznie pokonał całą konkurencję we wszystkich najpopularniejszych testach porównawczych w tym w zakresie tworzenia kodu minimalnie przegrywając jedynie w zakresie przetwarzania danych obrazowych [81].

Jednym z głównych problemów obecnych modeli LLM jest rozmiar kontekstu jaki są w stanie przyswoić. Popularne stały się rozwiązania pozwalające na tak zwaną rozmowę z dokumentami (ang. *chat with documents*) czyli odpowiedzi na pytania związane z dokumentami dostarczonymi przez użytkownika modelu. Modele mają choć coraz większy to jednak ograniczony kontekst jaki można wykorzystać. Popularnym rozwiązaniem stało się dzielenie tekstu na fragmenty i przechowywanie go w wektorowej bazie danych. Model tworzy tzw. przedstawienia wektorowe (ang. *Embeddings*) czyli w tym przypadku wektorowe reprezentacje fragmentów tekstu które pozwalają podczas zapytania odnaleźć fragmenty najbardziej związane z pytaniem użytkownika i użyć

ich jako kontekst na podstawie którego można zbudować odpowiedź. Jest to pomocne gdy cały dokument mający posłużyć jako źródło wiedzy do odpowiedzi na zadane pytanie jest zbyt duży i nie może być wykorzystany w całości jako kontekst dla modelu. Same modele również cały czas są rozwijane w kierunku obsługi coraz to większych kontekstów. Wspomniany wcześniej model Claude 3 na starcie oferuje kontekst o rozmiarze dwustu tysięcy tokenów lecz twórcy modelu deklarują, że cała rodzina modeli ma możliwość obsługi kontekstu przekraczającego milion tokenów [81]. Opcja tak dużego kontekstu zostanie udostępniona wybranym klientom [81]. Możliwości modelu w tym zakresie zostały sprawdzone metodą igły w stogu siana (NIAH od ang. *Needle In A Haystack*), która polega na umieszczeniu losowego faktu bądź komunikatu będącego igłą w długim tekście [81]. Model ma za zadanie udzielić odpowiedzi związanej z tym sztucznie wstawionym tekstem. Opisywany model Claude 3 charakteryzuje się ponad 99% skutecznością dla kontekstu o długości dwustu tysięcy tokenów [81]. Model w niektórych przypadkach wskazywał wręcz które zdanie zostało sztucznie wstawione do tekstu [81].

Coraz bardziej rozwijają się modele multimodalne, które pozwalają na stosowanie podpowiedzi zawierających wiele modalności jednocześnie. Przykładowo można przesłać obraz i zadać pytanie dotyczące tego obrazu. Pozwalają na to m.in. modele takie jak Gemini 1.5 Pro, Chat GPT-4 czy model z otwartą licencją LLaVA-1.6-34b. Można wykorzystywać również inne modalności takie jak nagrania dźwiękowe czy filmy. Modele multimodalne stają się coraz lepsze i jest to na pewno właściwy krok w drodze do AGI, lecz w zadaniu segmentacji nie są w stanie w tym momencie zastąpić bardziej wyspecjalizowanych modeli jak przykładowo SAM czy algorytmów klasycznych. Model LLaVA-1.6-34b jest w stanie odpowiedzieć poprawnie na pytanie o ilość białych obiektów na obrazie tylko w przypadku bardzo prostych obrazów (Rysunek 8.8 a). Już bardzo niewielkie zwiększenie złożoności obrazu powoduje błędną odpowiedź (Rysunek 8.8 b). Model radzi sobie znacznie lepiej odpowiadając na pytanie o kształt obiektów na obrazie (Rysunek 8.8). Model LLaVA-1.6-34b pokazuje swoją siłę w zadaniach związanych z opisem obrazu. Odpowiadając na prośbę o opis mikroskopowego obrazu erytrocytów model poprawnie opisał kolorystykę obrazu, kształt obiektów oraz fakt, że jest to obraz z mikroskopu optycznego, nie był w stanie jednak określić, że na obrazie znajdują się erytrocyty (Rysunek 8.9 a). Można to zrozumieć ponieważ jest to obraz specjalistyczny więc prawdopodobnie obrazy tego typu nie były licznie reprezentowane w zbiorze uczącym. Model LLaVA-1.6-34b daje świetne rezultaty opisując zdjęcie ręki człowieka zamierzającego chwycić spadające jabłko (Rysunek 8.9 b). Cała scena została dokładnie opisana również z takimi detalami jak ocena, że tło jest rozmyte oraz, przypuszczenie na podstawie zielonego koloru dominującego w tle, że zdjęcie zostało zrobione na zewnątrz w parku bądź ogrodzie. Imponujące jest też opisanie przez model, że jabłko zostanie złapane przez wyciągniętą dłoń. Rozwój modeli multimodalnych pozwoli stworzyć bardzo naturalny sposób interakcji człowieka z dużymi modelami językowymi.



Rysunek 8.8 Wynik działania modelu LLaVA-1.6-34b na przykładzie prostych obiektów na jednolitym tle.



(a)

(b)

Rysunek 8.9 Wynik działania modelu LLaVA-1.6-34b na przykładzie złożonych obrazów przedstawiających erytrocyty na obrazie pochodzącym z mikroskopu optycznego (a) oraz zdjęcie ręki człowieka zamierzającego chwycić spadające jabłko (b).

Zagrożeniem, które warto wskazać jest cenzurowanie modeli LLM odbywające się według arbitralnych reguł wyznaczanych przez poszczególne przedsiębiorstwa tworzące te modele. Może to prowadzić do manipulacji społeczeństwem na dużą skalę. Przeciwwagą dla tego zagrożenia są modele niecenzurowane takie jak udostępniony na licencji otwartej model Grok-1 od przedsiębiorstwa X [82]. Jest to ogromny model zawierający 314 miliardów parametrów oparty na architekturze mieszaniny ekspertów. Jest to kolejny znaczący krok w rozwoju modeli z otwartą licencją.

Jako katalizatory, które moim zdaniem będą przyspieszać rozwój metod cyfrowego przetwarzania i analizy obrazów należy wskazać rozwój elektronicznych urządzeń ubieralnych (ang. wearable devices) takich jak Rabbit R1 czyli przenośny asystent z wbudowaną sztuczną inteligencją [83], czy Apple Vision Pro będące okularami rzeczywistości mieszanej (ang. Mixed reality). Tego typu urządzenia będą potrzebowały coraz doskonalszych rozwiązań cyfrowego przetwarzania i analizy obrazów pozwalających na coraz to lepszą analizę i interakcję z otoczeniem. Nagły rozwój sztucznej inteligencji, który przebił się do świadomości przeciętnego człowieka stworzył pewien sprzyjający klimat pozwalający wdrażać jeszcze do niedawna bardzo futurystycznie kojarzące się rozwiązania. Być może pociągnie to za sobą szybsze ich upowszechnienie co przełoży się na wzrost nakładów na rozwój technologii w nich użytych. Innym obszarem gdzie szeroko stosuje się cyfrowe przetwarzanie i analizę obrazów jest robotyka. Rozwój sztucznej inteligencji wprowadził wielkie ożywienie również w tej dziedzinie. Możliwość analizy i interakcji z otoczeniem dzięki modelom opartym na sztucznych sieciach neuronowych już na obecnym poziomie pozwala myśleć o realnym zastosowaniu robotów w wielu obszarach. Prawdopodobnie jednym z pierwszych obszarów, gdzie znajdą zastosowanie roboty (również humanoidalne) będzie przemysł wytwórczy oraz magazyny. Fabryki obecnie używaja na dużą skalę robotów przemysłowych, które są odgórnie zaprogramowane, a magazyny automatyczne wykorzystują m.in. roboty, które potrzebują wyspecjalizowanej infrastruktury i działają według wyznaczonych wcześniej reguł. Wielkie przedsiębiorstwa pracują nad rozwiązaniami z obszaru robotyki. Przykładowo Tesla rozwija swojego humanoidalnego robota Optimus, 1x roboty EVE i NEO [84], czy Amazon, który testuje wdrożenie robota Digit od Agility Robotics w swoich magazynach [85]. Przedsiębiorstwo Figure podpisało umowę z producentem samochodów BMW na wdrożenie humanoidalnych robotów w swoich fabrykach [86]. Nowa fala robotów wykorzystujących sztuczną inteligencję będzie operować w otoczeniu nieprzystosowanym specjalnie dla robotów, więc analiza i interakcja z otoczeniem głównie przy użyciu obrazu z wbudowanych kamer będzie miała kluczowe znaczenie. takiemu Wytrenowanie modelu sztucznej inteligencji pozwalającego robotowi na poruszanie się i wykonywanie zadań jest skomplikowanym zadaniem. Przedsiębiorstwo NVIDIA stworzyło środowisko symulacyjne o nazwie Isaac Gym pozwalające na uczenie
przez wzmacnianie modeli sterujących robotami [87]. Wewnątrz przedsiębiorstwa powstała specjalna grupa, której zadaniem jest stworzenie modelu fundamentalnego pozwalającego na przemieszczenie się i interakcje zarówno w świecie realnym jak i w światach wirtualnych [88, 89]. Uczenie tych modeli w świecie wirtualnym pozwoli wygenerować syntetyczne dane i przyspieszyć proces. Rozwój modeli generatywnych jak przykładowo Sora, który to model jest w stanie tworzyć realistyczne i spójne filmy na podstawie podpowiedzi tekstowej znacznie ułatwi tworzenie takich wirtualnych światów [90]. Jednym z zagrożeń, które może spowolnić rozwój sztucznej inteligencji, a tym samym rozwój cyfrowego przetwarzanie i analizy obrazów jest niedobór chipów. Zapotrzebowanie na moc obliczeniową pozwalającą na trenowanie coraz to większych modeli stale rośnie, a podaż chipów, które mogą tę moc obliczeniową dostarczyć nie nadąża za popytem. Prowadzone są badania nad rozwiązaniami opartymi o architekturę neuromorficzną co pozwoli na wydajniejsze obliczenia przy mniejszym zużyciu energii oraz użycie architektur sieci opartych na pulsujących sieciach neuronowych (SNN od ang. Spiking neural network) [91]. Współzałożyciel i prezes przedsiębiorstwa NVIDIA Jensen Huang podczas wystąpienia z 18 marca 2024 roku, ogłosił nową platformę obliczeniową ukierunkowaną na rozwój AI nazwaną Blackwell, która jest znacznie wydajniejsza, efektywna energetycznie oraz bardziej skalowalna od istniejących rozwiązań [92, 93]. Potwierdził również kierunek polegający na trenowaniu modeli wykorzystywanych w robotyce w wirtualnym środowisku Isaac Sim stworzonym w celu projektowania oraz testowania robotów [94]. Korzysta ono z platformy Omniverse do tworzenia wirtualnych światów, które mogą być odzwierciedleniem rzeczywistości. Ponadto jest ona zintegrowana z wspomnianym wcześniej w tym rozdziale Apple Vision Pro pozwalającym na poruszanie się po tym wirtualnym świecie [95]. Przykładowo może powstać cyfrowy bliźniak (ang. digital twin) fabryki pozwalający na stworzenie i przetestowanie systemu automatyzacji opartej na robotach wykorzystujących sztuczną inteligencję, a następnie wdrożenie gotowego rozwiązania w realnym świecie. To podejście wraz z zestawem narzędzi ma stać się podstawą kolejnej generacji robotyki opartej na sztucznej inteligencji. Ogłoszono również projekt GR00T będący modelem fundamentalnym dla robotów humanoidalnych [96]. Została zaprezentowana już istniejąca możliwość generowania ruchów robota w reakcji na polecenia głosowe wydane językiem naturalnym. Umiejętność ta została wytrenowana właśnie z wykorzystaniem podejścia opartego na symulacji z użyciem modelu językowego współpracującego z modelem GR00T. Wizja zaprezentowana przez przedsiębiorstwo NVIDIA potwierdza, że w najbliższych latach będzie nacisk na rozwój robotyki w oparciu o modele sztucznej inteligencji.

Podsumowując, prawdopodobne wydaje się szerokie wykorzystanie w przyszłości narzędzi typu AutoGen czyli pozwalających łączyć wiele narzędzi potrzebnych do rozwiązania problemu z dostępem poprzez przyjazny interfejs modelu LLM. Pozwoli to przykładowo łączyć wiele modeli z odmienną architekturą oraz wytrenowanych na różnych zbiorach danych do rozwiązania zadanego problemu. Zautomatyzuje to również sam proces znajdowania rozwiązywania rozpatrywanego zadania. Przykładowo osoba pragnąca przeprowadzić segmentację zadanego zbioru danych obrazowych w celu wyciągnięcia wniosków będzie w stanie używając komend uzyskać przepis jak użyć kombinacji istniejących algorytmów klasycznych, gotowych modeli czy architektur sieci neuronowych do wytrenowania na zadanym zbiorze danych, a to wszystko przez przyjazny interfejs. Obecnie podobne zadania realizowane są za pomocą narzędzi automatyzujących proces uczenia maszynowego (ang. AutoML), które potrafia optymalizować proces wyboru modeli czy wyboru hiperparametrów jak przykładowo biblioteka AutoKeras [97]. Takie narzędzia również będą mogły być wykorzystywane w ramach narzędzi z interfejsem w postaci modelu LLM lub zostaną przez nie zastąpione. Przyszłość zapowiada się bardzo interesująco. Rozwój algorytmów opartych na sztucznych sieciach neuronowych przebiega w zawrotnym tempie. Największe firmy technologiczne świata inwestują miliardy dolarów żeby zdobyć pozycję lidera w tym wyścigu. Prawdopodobnie już istnieją nieupublicznione jeszcze rozwiązania rozwijane w tajemnicy przez te firmy, które już niedługo ujrza światło dzienne i podniosą poprzeczkę jeszcze wyżej. Demokratyzacja dostępności modeli sztucznej inteligencji w postaci udostępniania modeli na licencjach otwartych oraz związane z tym tworzenie się społeczności badaczy i pasjonatów tematu przyciągniętych na fali popularności modeli generatywnych, również może mieć znaczący wkład w rozwój tej dziedziny. Pojawienie się ogólnej sztucznej inteligencji, która zrewolucjonizuje wszystko to pytanie raczej kiedy, a nie czy. Nieprawdopodobnie szybki rozwój trwa i zmieni wiele dziedzin naszego życia, również obszar cyfrowego przetwarzanie i analizy obrazów w tym obrazów medycznych.

Bibliografia

- R. E. Woods i R. C. Gonzalez, Digital Image Processing Fourth Edition, New York: Pearson, 2018.
- [2] W. K. Pratt, Digital Image Processing, 4 edition red., Wiley-Interscience, 2007.
- [3] T. Pavlidis, Grafika i przetwarzanie obrazów, Warszawa: Wydawnictwa naukowotechniczne, 1987.
- [4] A. K. Jain, Fundamentals of Digital Image Processing, New Jersey: Prentice Hall, 1989.
- [5] R. Tadeusiewicz i P. Korohoda, Komputerowa analiza i przetwarzanie obrazów, Kraków: Wydawnictwo Fundacji Postępu Telekomunikacji, 1997.
- [6] D. Mehta, M. Siddiqui i A. Javaid, "Recognition of Emotion Intensities Using Machine Learning Algorithms: A Comparative Study," Sensors 19(8), p. 1897, 21 Apr 2019.
- [7] W. Malina, S. Ablameyko i W. Pawlak, Podstawy cyfrowego przetwarzania obrazów, Warszawa: Akademicka Oficyna Wydawnicza EXIT, 2002.
- [8] Y. Wan i Q. Xie, "A Novel Framework for Optimal RGB to Grayscale Image Conversion," w 2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2016, pp. pp. 345-348.
- [9] S. Suhas i R. C. Venugopal, "Mri image preprocessing and noise removal technique using linear and nonlinear filters," w *International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, 2017, p. 1–4.

- [10] A. Raut, S. Kulkarni i S. S. Hunnur, "Implementation of image processing for detection of brain tumors," w *International Conference on Computing Methodologies* and Communication (ICCMC), 2017, p. 717 – 722.
- [11] B. K. Shreyamsha Kumar, "Image denoising based on gaussian/bilateral filter and its method noise thresholding," *Springer, editor, Signal image and video processing*, p. 1159 – 1172, 2013.
- [12] S. Allner, T. Koehler, A. Fehringer, L. Birnbacher, M. Willner, F. Pfeiffer i P. B. Noël, "Bilateral filtering using the full noise covariance matrix applied to x-ray phasecontrast computed tomography.," *Physics in Medicine and Biology*, 61(10), pp. 3867-3884, 2016.
- [13] S. Paris, P. Kornprobst, J. Tumblin i F. Durand, "Bilateral Filtering: Theory and Applications," w Foundations and Trends in Computer Graphics and Vision Vol. 4, No. 1, 2008, p. 1–73.
- [14] M. Buczkowski i K. Saeed, "Fusion-Based Noisy Image Segmentation Method," w Chaki R., Cortesi A., Saeed K., Chaki N. (eds) Advanced Computing and Systems for Security. Advances in Intelligent Systems and Computing, vol 396, New Delhi, Springer, 2016, pp. pp 21-35.
- [15] R. S. Sternberg, "Biomedical image processing," IEEE Computer: Computer architectures for image processing, 16, p. 22 – 34, 1983.
- [16] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. H. Romeny, J. B. Zimmerman i K. Zuiderveld, "Adaptive histogram equalization and its variations," w *Computer Vision, Graphics, and Image Processing, Volume 39, Issue 3*, 1987, pp. 355-368.
- [17] A. Reza, "Realization of the Contrast Limited Adaptive Histogram Equalization (CLAHE) for Real-Time Image Enhancement.," *The Journal of VLSI Signal*

Processing-Systems for Signal, Image, and Video Technology 38, p. 35–44, 2004.

- [18] L. Ping-Sung, C. Tse-Sheng i C. Pau-Choo, "A fast algorithm for multilevel thresholding," *Journal of Information Science and Engineering* 17, 17(5), p. 713 – 727, 2008.
- [19] A. Fabijanska, "A survey of thresholding algorithms on yarn images," w 2010⁻ Proceedings of VIth International Conference on Perspective Technologies and Methods in MEMS Design, 2010., p. 23–26.
- [20] J. Canny, "A computational approach to edge detection.," *IEEE Trans. Pattern Anal. Mach. Intel.* 6, p. 679–698, 1986.
- [21] R. Deriche, "Using Canny's criteria to derive a recursively implemented optimal edge detector." Int. J. Comput. Vis. 1(2), p. 167–187, 1987.
- [22] Z. Huang, S. Jiang, Z. Yang, Y. Ding, W. Wang i Y. Yu, "Automatic multi-organ segmentation of prostate magnetic resonance images using watershed and nonsubsampled contourlet transform," *Biomedical Signal Processing and Control, 25,* p. 53 61, 2016.
- [23] R. Nock i F. Nielsen, "Statistical region merging.," IEEE Trans. Pattern Anal. Mach. Intell. 26 (11), p. 1452–1458, 2004.
- [24] K. Saeed, Image Analysis for Object Recognition, Białystok: Bialystok University of Technology Press, 2004.
- [25] X. Chenyang, D. L. Pham i J. L. Prince, "Image segmentation using deformable models.," *Handbook Med. Imaging 2*, p. 129–174, 2000.
- [26] L. He, Z. Peng, B. Everding, X. Wang, C. Y. Han, K. L. Weiss i W. G. Wee, "A comparative study of deformable contour methods on medical image segmentation.," *Image Vis. Comput. 26*(2), p. 141–163, 2008.

- [27] C. Xu i J. L. Prince, "Snakes, shapes, and gradient vector flow.," IEEE Trans. Image Process. 7 (3), p. 359–369, 1998.
- [28] I. Goodfellow, Y. Bengio i A. Courville, Deep Learning, MIT Press, 2016.
- [29] R. Tadeusiewicz, Sieci Neuronowe, Warszawa: Akademicka Oficyna Wydawnicza, 1993.
- [30] R. Tadeusiewicz i M. Szaleniec, Leksykon sieci neuronowych, Wrocław: Wydawnictwo Fundacji "Projekt Nauka", 2015.
- [31] ImageNet, ",http://image-net.org/challenges/LSVRC/," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [32] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu i X. Zheng, *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, 2015.
- [33] F. Chollet i inni, Keras, GitHub, 2015.
- [34] NVIDIA, "https://developer.nvidia.com/cuda-zone," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [35] Google, "https://colab.research.google.com/," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [36] Kaggle, "https://www.kaggle.com/," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [37] K. Simonyan i A. Zisserman, "Very Deep Convolutional Networks for Large-Scale

Image Recognition," *arXiv:1409.1556*, 2014.

- [38] M. Buczkowski i K. Saeed, "A multistep approach for micro tomography obtained medical image segmentation," *Journal of medical informatics & technologies Vol.* 23/2014, 2014.
- [39] M. Buczkowski, K. Saeed, J. Tarasiuk, S. Wroński i J. Kosior, "An approach for micro-tomography obtained medical image segmentation.," w Chaki, R., et al. (eds.) Applied Computation and Security Systems, Advances in Intelligent Systems and Computing, vol 304, 2015.
- [40] F. H. Attix, Introduction to Radiological Physics and Radiation Dosimetry, Wiley, 1986.
- [41] R. M. Eisberg, Podstawy Fizyki Współczesnej, Warszawa: PWN, 1968.
- [42] J. H. Hubbell, "Radiation Physics," w Encyclopedia of Physical Science and Technology (Third Edition), Academic Press, 2003, pp. 561-580.
- [43] L. King, U. Tayal, I. Castellano, J. Stirrup, F. Pontana, J. Earls, E. Nicol i R. Schofield, "Image reconstruction: Part 1 understanding filtered back projection, noise and image acquisition," *Journal of Cardiovascular Computed Tomography, Volume 14, Issue 3*,, pp. 219-225, 2020.
- [44] J. W. Leis, Digital Signal Processing Using MATLAB for Students and Researchers, New Jersey: John Wiley & Sons, Inc., 2011.
- [45] P. Szczepańczyk, M. Szlachta, N. Złocista-Szewczyk, J. Chłopek i K. Pielichowska, "Recent Developments in Polyurethane-Based Materials for Bone Tissue Engineering," *Polymers 13(6)*, 946, 2021.
- [46] M. Tanzi, F. S., P. Petrini, A. Tanini, E. Piscitelli, S. Zecchi-Orlandini i M. L. Brandi, "Cytocompatibility of Polyurethane foams as Biointegrable Matrices for the

Preparation of Scaffolds for Bone Reconstruction," *Journal of Applied Biomaterials* and Biomechanics, Volume 1, Issue 1, pp. 58-66, 2003.

- [47] J. Rogowska, "Overview and fundamentals of medical image segmentation.," w Handbook of Medical Imaging, Academic Press Inc., 2000, p. 69–85.
- [48] B. Jahne, Digital Image Processing: Concept, Algorithms, and Scientific Applications., New York: Springer, 1997.
- [49] M. Sezgin i B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation.," J. Electron. Imaging 13(1), p. 146–165, 2004.
- [50] M. Buczkowski, P. Szymkowski i K. Saeed, "Segmentation of Microscope Erythrocyte Images by CNN-Enhanced Algorithms," Sensors 21(5), p. 1720, 2021.
- [51] S. Tao, Y. Jiang, S. Cao, C. Wu i Z. Ma, "Attention-Guided Network with Densely Connected Convolution for Skin Lesion Segmentation". *Sensors 2021, 21, 3462.*.
- [52] J. Schindelin, I. Arganda-Carreras i E. Frise, "Fiji: an open-source platform for biological-image analysis," *Nature methods 9(7)*, pp. 676-682, 2012.
- [53] Insight toolkit (itk), "https://itk.org/," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [54] G. Lehmann, "Label object representation and manipulation with itk," *The Insight Journal*, 2007..
- [55] P. Soille, "Morphological image analysis principles and applications," *Springer-Verlag Berlin Heidelberg*, 2003.
- [56] J. Caicedo, A. Goodman, K. Karhohs, B. Cimini, J. Ackerman, M. Haghighi, C. Heng, T. Becker, M. Doan, C. McQuin i e. al., "Publisher correction: Nucleus segmentation across imaging experiments: The 2018 data science bowl.," *Nat. Methods*, p. 1247– 1253, 2020.

- [57] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov i L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," w *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 18–22 June 2018.
- [58] V. Lebedev i V. Lempitsky, "Speeding-up convolutional neural networks: A survey," Bull. Pol. Acad. Sci. Tech. Sci. 66, p. 799–810, 2018.
- [59] D. Johnson, N. Hilal i W. R. Bowen, "Basic Principles of Atomic Force Microscopy," w Atomic Force Microscopy in Process Engineering, Butterworth-Heineman, 2009, pp. 1-30.
- [60] M. Kaczmarska, M. Fornal, F. H. Messerli, J. Korecki, T. Grodzicki i K. Burda, "Erythrocyte Membrane Properties in Patients with Essential Hypertension," *Cell Biochemistry and Biophysics* 67, p. 1089–1102, 2013.
- [61] Komisja Europejska, "Rozporządzenie Komisji (UE) 2022/63 z dnia 14 stycznia 2022
 r.," Dziennik Urzędowy Unii Europejskiej, styczeń 2022.
- [62] Microsoft, "https://copilot.microsoft.com/," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [63] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. Berg, W. Lo, P. Dollár i R. Girshick, "Segment Anything," arXiv:2304.02643, 2023.
- [64] Z. Sun, Y. Fang, T. Wu, P. Zhang, Y. Zang, S. Kong, Y. Xiong, D. Lin i J. Wang, "Alpha-CLIP: A CLIP Model Focusing on Wherever You Want," arXiv:2312.03818, 2023.
- [65] IDEA-Research, "https://github.com/IDEA-Research/Grounded-Segment-Anything,"[Online]. [Data uzyskania dostępu: 06 03 2024].

- [66] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu i L. Zhang, "Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection," arXiv:2303.05499, 2023.
- [67] Meta, "https://github.com/facebookresearch/segment-anything," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [68] A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. Singh Chaplot, D. de las Casas, E. B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. Renard Lavaud, L. Saulnier, M.-A. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. Le Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix i W. El Sayed, "Mixtral of experts," *arXiv:2401.04088*, January 2024.
- [69] Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li, E. Zhu, L. Jiang, X. Zhang, S. Zhang, J. Liu, A. H. Awadallah, R. W. White, D. Burger i C. Wang, "AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation," arXiv:2308.08155, 2023.
- [70] S. Su YR, A. Veeravagu i G. Grant, "Neuroplasticity after Traumatic Brain Injury," w *Translational Research in Traumatic Brain Injury*, Boca Raton, CRC Press/Taylor and Francis Group, 2016.
- [71] A. Ignatov, L. Van Gool i R. Timofte, "Replacing Mobile Camera ISP with a Single Deep Learning Model," w IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, 2020.
- [72] Z. Hui, X. Wang, L. Deng i X. Gao, "Perception-Preserving Convolutional Networks for Image Enhancement on Smartphones," w Leal-Taixé, L., Roth, S. (eds) Computer Vision – ECCV 2018 Workshops. ECCV 2018. Lecture Notes in Computer Science(), vol 11133, 2019.
- [73] A. Ignatov, R. Timofte, A. Kulik, S. Yang, K. Wang, F. Baum, M. Wu, L. Xu i L. Van Gool, "AI Benchmark: All About Deep Learning on Smartphones in 2019,"

arXiv:1910.06663, 2019.

- [74] OpenAI, "https://openai.com/chatgpt," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [75] Midjourney, "https://www.midjourney.com/," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [76] A. Vaswani, N. M. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser i I. Polosukhin, "Attention is All you Need," *Neural Information Processing Systems*, 2017.
- [77] W. Research, "https://www.wolfram.com/wolfram-plugin-chatgpt/," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [78] E. Musk i Neuralink, "An Integrated Brain-Machine Interface Platform With Thousands of Channels," *J Med Internet Res. 2019*, 2019.
- [79] T. Hugo, M. Louis, S. Kevin i e. al, "Llama 2: Open Foundation and Fine-Tuned Chat Models," arXiv:2307.09288, 2023.
- [80] B. Rozière, J. Gehring, F. Gloeckle, S. Sootla, I. Gat, X. E. Tan, Y. Adi, J. Liu, R. Sauvestre, T. Remez, J. Rapin, A. Kozhevnikov, I. Evtimov, J. Bitton, M. Bhatt, C. Canton Ferrer, A. Grattafiori, W. Xiong, A. Défossez, J. Copet, F. Azhar, H. Touvron, L. Martin, N. Usunier, T. Scialom i G. Synnaeve, "Code Llama: Open Foundation Models for Code," *arXiv:2308.12950*, 2023.
- [81] Anthropic, "https://www.anthropic.com/news/claude-3-family," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [82] X, "https://x.ai/blog/grok-os," [Online]. [Data uzyskania dostępu: 20 3 2024].
- [83] Rabbit, "https://www.rabbit.tech/," [Online]. [Data uzyskania dostępu: 06 03 2024].

- [84] 1x, "https://www.1x.tech/," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [85] A. Robotics, "https://agilityrobotics.com/products/digit," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [86] Figure, "https://www.figure.ai/," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [87] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa i G. State, "Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning," *arXiv:2108.10470*, 2021.
- [88] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan i A. Anandkumar, "Voyager: An Open-Ended Embodied Agent with Large Language Models," arXiv:2305.16291, 2023.
- [89] Y. Jiang, A. Gupta, Z. Zhang, G. Wang, Y. Dou, Y. Chen, L. Fei-Fei, A. Anandkumar, Y. Zhu i L. Fan, "VIMA: General Robot Manipulation with Multimodal Prompts," arXiv:2210.03094, 2022.
- [90] OpenAI, "https://openai.com/sora," [Online]. [Data uzyskania dostępu: 06 03 2024].
- [91] A. Siddique, M. Vai i S. Pun, "A low cost neuromorphic learning engine based on a high performance supervised SNN learning algorithm," *Scientific Reports 13*, p. 6280, 2023.
- [92] NVIDIA, "https://nvidianews.nvidia.com/news/nvidia-blackwell-platform-arrives-to-power-a-new-era-of-computing," [Online]. [Data uzyskania dostępu: 20 03 2024].
- [93] NVIDIA, "https://www.nvidia.com/en-us/data-center/technologies/blackwellarchitecture/," [Online]. [Data uzyskania dostępu: 20 03 2024].
- [94] NVIDIA, "https://developer.nvidia.com/isaac-sim," [Online]. [Data uzyskania dostępu: 20 03 2024].

- [95] NVIDIA, "https://www.nvidia.com/en-us/omniverse/," [Online]. [Data uzyskania dostępu: 20 03 2024].
- [96] NVIDIA, "https://nvidianews.nvidia.com/news/foundation-model-isaac-roboticsplatform," [Online]. [Data uzyskania dostępu: 20 3 2024].
- [97] D. L. a. T. A. University, "https://autokeras.com/," [Online]. [Data uzyskania dostępu: 06 03 2024].

Spis używanych skrótów

RGB – model przestrzeni barw oparty na kolorach czerwonym, zielonym i niebieskim (ang. R – red, G – green, B - blue)

CNN – konwolucyjna sieć neuronowa (ang. Convolutional neural network)

MLP – perceptron wielowarstwowy (ang. *multilayer perceptron*)

API - interfejs programistyczny aplikacji (ang. Application Programming Interface)

GVF – ang. *Gradient vector flow*

AFM – mikroskop sił atomowych (ang. *atomic force microscope*)

CLAHE - limitowane kontrastem adaptacyjne wyrównanie histogramu (ang. *contrast limited adaptive histogram equalization*)

WW/WL – przekształcenie typu szerokośc okna/poziom okna (ang. *Window width/Window level*)

LLM – duży model językowy (ang. Large Language Model)

AGI – ogólna sztuczna inteligencja (ang. artifical general intelligence)

Spis rysunków

Rysunek 2.1 Układ współrzędnych stosowany do opisu obrazu cyfrowego (układ kartezjański obrócony o 90°) gdzie centra pikseli posiadają współrzędne opisane liczbami całkowitymi.....9 Rysunek 2.2 Przykładowy obraz zawierający trzy obiekty posiadające różne poziomy intensywności (a) oraz odpowiadający mu histogram (b) na którym można zaobserwować, że na obrazie występują cztery poziomy intensywności (trzy dla obiektów i jeden Rysunek 2.3 Przykładowy rzeczywisty obraz pochodzący z mikroskopu optycznego w skali szarości (8-bit) zawierający wiele obiektów posiadających różne poziomy intensywności (a) oraz odpowiadający mu histogram (b) na którym można zaobserwować, że na obrazie występują dwa maksima liczby pikseli (jedno lokalne i jedno globalne) dla konkretnej intensywności: jedno w okolicy poziomu intensywności równego 26, a drugie w okolicy Rysunek 2.4 Histogramy z grupowaniem w przedziały ilości pikseli dla zadanego zakresu wartości intensywności: 64 przedziały (a), 32 przedziały (b), 16 przedziałów (c).....14 Rysunek 2.5 Przykładowy rzeczywisty kolorowy obraz pochodzący z mikroskopu optycznego, przedstawiający erytrocyty w modelu RGB (8-bit na kanał) zawierający wiele obiektów posiadających różne poziomy intensywności (a) oraz dopowiadające mu histogramy dla kanału R (b), G (c) oraz B (d) na których można zaobserwować różnice Rysunek 2.6 Przykładowy obraz o małym kontraście (a) i jego histogram (c) oraz obraz o Rysunek 2.7 Przykładowy obraz o dużej jasności (a) i jego histogram (c) oraz obraz o małej Rysunek 2.8 Histogram oryginalnego obrazu (a) oraz histogram tego samego obrazu po Rysunek 2.9 Przykładowy obraz o rozmiarze 7 na 7 pikseli (7 wierszy i 7 kolumn) w skali

Rysunek 2.10 Piksel p(x, y) oraz jego sasiedztwo N4(p(x, y)) oznaczające sasiadów poziomych i pionowych (a) oraz sasiedzi po przekatnej ND(p(x, y)) (b) (należy pamietać, że kartezjański układ współrzędnych zastosowany dla obrazów cyfrowych jest obrócony o Rysunek 2.11 Przykładowy układ pikseli gdzie założono zbiór $V = \{1\}$ (a), 4-przyleganie oznaczone liniami (b), 8-przyleganie oznaczone liniami (c), m-przyleganie oznaczone liniami (d). Na podstawie [1]......25 Rysunek 2.12 Dwa regiony Ri oraz Rj z zaznaczonymi na niebiesko pikselami, które decydują o przyleganiu regionów. Jeżeli rozpatrujemy 4-przyleganie to regiony są rozłaczne, a w przypadku 8-przylegania sa regionami przylegającymi. Na podstawie [1]. 26 Rysunek 2.13 Piksel oznaczony kolorem niebieskim w przypadku 8-łaczności jest pikselem należącym do granicy regionu, a w przypadku użycia 4-łączności nie jest. Na podstawie Rysunek 2.14 Zobrazowanie metryki D4 gdzie cyfry odpowiadają odległościom. Na Rysunek 2.15 Zobrazowanie metryki D8 gdzie cyfry odpowiadaja odległościom. Na Rysunek 2.16 Przykład obrazu po którego wszystkich pikselach przesuwa się maska $\omega(s,t)$ o rozmiarach 3 × 3 gdzie w powiększeniu widoczne są współrzędne pikseli obrazu znajdujących się pod maską gdy jej środek przypada w punkcie (x, y) oraz współczynniki Rysunek 2.17 Maska o rozmiarze 3 × 3 zastosowana dla pikseli skrajnych w części wykracza poza obraz co powoduje, że wynik działania maski na obrazie jest w tych Rysunek 2.18 Obraz o rozmiarze 3×3 do którego planuje się zastosować maskę 3×3 powiększony poprzez dodanie wierszy i kolumn wypełnionych zerami z każdej strony Rysunek 2.19 Korelacja (c, d, e) oraz konwolucja (f, g, h) obrazu f z wykorzystaniem

| Rysunek 2.21 Jądro oparte na funkcji Gaussa wraz ze współczynnikiem normalizującym. |
|---|
| Na podstawie [1] |
| Rysunek 2.22 Przykładowa maska wyostrzająca |
| Rysunek 2.23 Euklidesowe odległości od centrum dla kwadratowych jąder (masek) |
| różnych rozmiarów <i>m</i> . Na podstawie [1]38 |
| Rysunek 2.24 Dwuwymiarowa funkcja Gaussa użyta w celu stworzenia maski do |
| konwolucji przestrzennej |
| Rysunek 3.1 Obraz erytrocytów uzyskany przy pomocy mikroskopu optycznego w skali |
| szarości oraz po przekształceniu typu WW/WL (ang. Window width/Window level) |
| zastosowanym w celu uwypuklenia niejednorodności tła (a), wizualizacja tego samego |
| obrazu w postaci wykresu 3d (b)44 |
| Rysunek 4.1 Przykładowy obraz binarny erytrocytów po segmentacji (a), oraz otrzymana |
| na jego podstawie mapa odległości (b) |
| Rysunek 4.2 Architektura perceptronu wielowarstwowego |
| Rysunek 4.3 Architektura konwolucyjnej sieci neuronowej na przykładzie sieci VGG16 |
| [37]. Wraz z przesuwaniem się w stronę głębszych warstw sieci wzrasta ilość filtrów w |
| poszczególnych warstwach, a zmniejsza się rozmiar warstwy ze względu na zastosowanie |
| warstw łączenia maksimum60 |
| Rysunek 4.4 Wizualizacja przykładowych filtrów konwolucyjnych o rozmiarze 3x3 na |
| przykładzie architektury CNN nazwanej VGG16 [37]. Każdy filtr pozwala wyodrębnić |
| inną cechę obrazu. Filtry z pierwszej warstwy operują na oryginalnym obrazie, a te z |
| kolejnych, głębszych warstw na coraz bardziej abstrakcyjnych wynikach działania filtrów z |
| poprzednich warstw61 |
| Rysunek 4.5 Wizualizacja map cech (ang. <i>feature maps</i>) na przykładzie architektury CNN |
| nazwanej VGG16 [37] (b) oraz przykładowy oryginalny obraz którym zasilono sieć |
| neuronową (a). Można zauważyć, że każda mapa zawiera odpowiedź na inną cechę obrazu |
| wejściowego62 |
| Rysunek 5.1 Fragment oryginalnego obrazu przekroju piany polimerowej67 |
| Rysunek 5.2 Fragment oryginalnego obrazu po filtracji filtrem medianowym. Pierwsza |
| iteracja (a) oraz druga iteracja (b) |

| Rysunek 5.3 Detektor krawędzi Canny-Deriche: fragment oryginalnego obrazu (a), mapa |
|---|
| gradientu (b), mapa gradientu po ścienieniu krawędzi (c) oraz wyniki detekcji krawędzi po |
| progowaniu z histerezą (d)70 |
| Rysunek 5.4 Przykładowa mapa gradientu uzyskana przy pomocy metody GVF71 |
| Rysunek 5.5 Przykłady mapy gradientów uzyskanych przy pomocy metody GVF oraz |
| deformowalnych modeli które iteracyjnie wyewoluowały w celu odnalezienia krawędzi |
| obiektów. Niebieskie punkty oznaczają dyskretne punkty użyte w celu wyznaczenia energii |
| modelu, a czerwone linie są połączeniami pomiędzy nimi72 |
| Rysunek 5.6 Przykłady pikseli wyciętych z oryginalnego obrazu po filtracji wstępnej przy |
| pomocy maski uzyskanej metodą deformowalnych modeli73 |
| Rysunek 5.7 Przykłady pikseli wyciętych z oryginalnego obrazu po filtracji wstępnej przy |
| pomocy maski uzyskanej metodą deformowalnych modeli oraz po wykorzystaniu metody |
| statystycznego łączenia obszarów73 |
| Rysunek 5.8 Przykłady binarnych obrazów uzyskanych przy pomocy wieloprogowego |
| progowania Otsu |
| Rysunek 5.9 Diagram przepływu opisanego algorytmu74 |
| Rysunek 5.10 Fragmenty przetwarzanych obrazów w oryginalnym rozmiarze: zawierające |
| małe obiekty (a) oraz duże obiekty (b)75 |
| Rysunek 5.11 Dwa przykłady binarnych obrazów uzyskanych z wykorzystaniem opisanego |
| algorytmu77 |
| Rysunek 5.12 Dwa przykłady binarnych obrazów uzyskanych z wykorzystaniem opisanego |
| algorytmu pokazujących większe fragmenty finalnych segmentacji |
| Rysunek 5.13 Fragmenty sztucznych obrazów z dodanym szumem o odchyleniu |
| standardowym równym 12 (a) oraz 20 (b) |
| Rysunek 5.14 Fragmenty sztucznych obrazów z odchyleniem standardowym szumu |
| równym 12 przetworzone przez (a) opisywany algorytm z progowaniem pojedynczym |
| progiem, (b) pierwszy etap opisywanego algorytmu bez drugiego etapu filtra bilateralnego |
| oraz z wieloprogowym progowaniem Otsu, (c) pierwszy etap opisywanego algorytmu z |
| wieloprogowym progowaniem Otsu, (d) opisywany algorytm |
| Rysunek 5.15 Fragmenty oryginalnych obrazów przetworzone przez opisywany algorytm: |
| (a) opisywany algorytm z progowaniem pojedynczym progiem, (b) pierwszy etap |

opisywanego algorytmu bez drugiego etapu filtra bilateralnego oraz z wieloprogowym progowaniem Otsu, (c) pierwszy etap opisywanego algorytmu z wieloprogowym Rysunek 6.1 Oryginalny obraz w skali szarości (a), obraz po transformacji WW/WL (b)..85 Rysunek 6.2 Odejmowanie tła. Fragmenty obrazów z odpowiadającymi im profilami intensywności. Górna część – oryginalny obraz, dolna część – obraz po odjęciu tła.........86 Rysunek 6.3 Usuwanie szumów: oryginalny obraz (a), po filtracji filtrem medianowym (b), Rysunek 6.5 Obraz po segmentacji algorytmem Otsu (a) oraz po wypełnieniu otworów w Rysunek 6.7 Obraz po segmentacji algorytmem wododziałowym (a) oraz po usunięciu obiektów małych i położonych przy krawędziach obrazu zaprezentowany jako obraz Rysunek 6.8 Wybrane ROI (a), ROI maskowane przy użyciu obiektów w pobliżu tego obecnie przetwarzanego (b), maska przetwarzanego obiektu (c), maska przetwarzanego Rysunek 6.9 Segmentacja Otsu z dwoma poziomami progowania (a), obiekt po segmentacji (b), obiekt po segmentacji z wypełnionymi otworami (c), kontur otrzymany z obrazu po Rysunek 6.10 Przykładowe kontury erytrocytów (czerwony) oraz krótkie (zielony) i długie Rysunek 6.11 Mapa prawdopodobieństwa (wynik działania modelu) (a), mapa po segmentacji z użyciem referencyjnego podejścia (zbliżenie połączonych obiektów) (b), wynik proponowanego algorytmu (zbliżenie połączonych obiektów) (c). Proponowany algorytm daje trochę lepsze wyniki w zadaniu segmentacji połączonych obiektów Rysunek 6.12 Przykład erytrocytów po segmentacji skategoryzowanych jako normalne, anormalne oraz błędna segmentacja......100 Rysunek 6.13 Przykładowy obraz wynikowy. Na wstępnie przefiltrowany obraz oryginalny nałożono kontur (czerwony), oś dłuższą (niebieski) oraz oś krótszą (zielony). 105 Rysunek 6.14 Przykładowy sztucznie wytworzony obraz (wygładzanie Gaussa - promień Rysunek 6.15 Fragmenty sztucznie wytworzonego testowego obrazu z wygładzaniem Gaussa równym 3 - (a), 5 - (b) oraz odchyleniem standardowym szumu równym: 15 - (a), Rysunek 7.2 Obraz erytrocytu uzyskany przy pomocy techniki mikroskopii sił atomowych z użyciem trybu kontaktowego (a), oraz jego trójwymiarowa reprezentacja z naniesionymi wymiarami (b).....112 Rysunek 7.3 Obrazy erytrocytów uzyskane przy pomocy techniki mikroskopii sił atomowych (a, b), obrazy z uwydatnioną teksturą (c, d)......115 Rysunek 7.4 Obrazy erytrocytów uzyskane przy pomocy techniki mikroskopii sił atomowych zawierające tylko duże obiekty z obrazów oryginalnych (a, b), (c, d) obrazy będące wynikiem odjęcia obrazów (a, b) od obrazów z uwydatnioną teksturą (Rysunek 7.3 c, d)......116 Rysunek 8.1 Skrypt języka Python wygenerowany przez model LLM, generujący losowe Rysunek 8.2 Skrypt języka Python wygenerowany przez model LLM, generujący losowe Rysunek 8.3 Skrypt języka Python wygenerowany przez model LLM zmieniający dane wynikowe odczytane z pliku csv. Użytkownik otrzymuje również odnośniki do danych źródłowych gdzie może zweryfikować odpowiedź. 125 Rysunek 8.4 Uproszczony fragment kodu w języku Python pokazujący warunki do oznaczenia erytrocytu jako nieprawidłowy/uszkodzony......126 Rysunek 8.5 Wykresy typu mapa ciepła przedstawiające dokładność otrzymanego wyniku Rysunek 8.6 Wyniki segmentacji fragmentu obrazu mikro-tomograficznego porowatej struktury w postaci piany polimerowej uzyskane po automatycznej segmentacji modelem

| SAM vit_h (a) oraz modelem SAM dostępnym poprzez internetową aplikację |
|--|
| demonstracyjną (b)131 |
| Rysunek 8.7 Wyniki segmentacji obrazu zawierającego erytrocyty pochodzącego z |
| mikroskopu optycznego uzyskane po automatycznej segmentacji modelem SAM vit_h (a) |
| oraz modelem SAM dostępnym poprzez internetową aplikację demonstracyjną (b) 131 |
| Rysunek 8.8 Wynik działania modelu LLaVA-1.6-34b na przykładzie prostych obiektów |
| na jednolitym tle |
| Rysunek 8.9 Wynik działania modelu LLaVA-1.6-34b na przykładzie złożonych obrazów |
| przedstawiających erytrocyty na obrazie pochodzącym z mikroskopu optycznego (a) oraz |
| zdjęcie ręki człowieka zamierzającego chwycić spadające jabłko (b)139 |

Spis tabel