Sprawozdanie z projektu
“Wykrywanie naczyń dna siatkówki oka”

Grupa laboratoryjna: L9	Jakub Kamieniarz (155 845) Tomasz Pawłowski (155 965)
Język programowania i dodatkowe biblioteki	Python (jupyter notebook), cv2, numpy, pandas, matplotlib.pyplot, os, glob, tensorflow, sklearn, skimage
Repozytorium	https://github.com/KamieniarzJakub/Medycyna
Wykorzystana baza obrazów:	https://www5.cs.fau.de/research/data/fundus-images/

1. Opis zastosowanych metod:
    a. przetwarzanie obrazów
        i. poszczególne kroki przetwarzania obrazu (w tym zastosowane filtrów),
        ii. krótkie uzasadnienie zastosowanego rozwiązania,
    b. uczenie maszynowe (4.0/5.0)
        i. przygotowanie danych - wyznaczanie wycinków obrazu, ekstrakcja cech z wycinków (jeśli zastosowano)
        ii. wstępne przetwarzanie zbioru uczącego (jeśli zastosowano)
        iii. zastosowane metody uczenia maszynowego wraz z informacją o przyjętych parametrach
        iv. wyniki wstępnej oceny zbudowanego klasyfikatora (testy *hold-out*)
        v. krótkie uzasadnienie zastosowanego rozwiązania

2. Wizualizacja wyników działania programu dla wybranych obrazów (warto pokazać zarówno sukcesy jak i porażki). Dla porównania należy zamieścić maskę ekspercką (*ground truth*). Wyniki wizualizacji należy zaprezentować dla co najmniej 5 obrazów (w przypadku metod uczenia maszynowego nie mogą to być obrazy, które zostały wykorzystane do nauczenia klasyfikatora).

1. Opis zastosowanych metod

Przetwarzanie obrazów

W celu analizy obrazów medycznych (siatkówki oka), zastosowano klasyczne techniki przetwarzania obrazów cyfrowych z wykorzystaniem biblioteki OpenCV oraz metod detekcji struktur naczyniowych. Głównym celem było wydobycie naczyń krwionośnych z obrazu funduszu oka i porównanie uzyskanych wyników z maską ekspercką.

Poszczególne kroki przetwarzania obrazu

1. **Wczytanie i przygotowanie danych:**
   Załadowano trzy obrazy: kolorowy obraz źródłowy, maskę pola widzenia oraz

maskę segmentacji przygotowaną przez eksperta. Obrazy mask są binarne, a obraz wejściowy został przekonwertowany z przestrzeni BGR do RGB.

2. **Wstępne przetwarzanie (funkcja preprocess_image):**
    - Ekstrakcja **zielonego kanału**, ponieważ zawiera on największy kontrast między naczyniami a tłem w obrazach siatkówki.
    - **Dylacja** została użyta do wzmocnienia struktur naczyniowych.
    - **Odszumianie** metodą Non-Local Means w celu usunięcia szumu bez degradacji detali.
    - **Normalizacja histogramu** dla poprawy kontrastu naczyń.

3. **Filtracja naczyniowa (funkcja apply_frangi_filter):**
    - Zastosowano **filtr Frangiego**, który jest specjalnie zaprojektowany do wzmacniania struktur rurowych, takich jak naczynia krwionośne.
    - Obraz wynikowy został przycięty do zakresu percentylowego [5%, 95%] i znormalizowany.

4. **Progowanie (funkcja threshold_vessels):**
    - Automatyczne wyznaczenie progu metodą **Otsu**, w celu uzyskania binarnej maski naczyń.

5. **Postprocessing (funkcja postprocess_image):**
    - Konwersja do wartości binarnych z dodatkowym ręcznym progiem (> 0.35).
    - **Usunięcie małych obiektów** (< 20 pikseli).
    - **Zamykanie morfologiczne** z wykorzystaniem dysku o promieniu 6, w celu wypełnienia luk.
    - **Szkieletowanie**, by uzyskać reprezentację topologiczną naczyń.
    - Na końcu, nałożenie maski pola widzenia (FOV), aby odfiltrować obszary spoza analizowanego regionu.

Uzasadnienie zastosowanego rozwiązania

Wybrana sekwencja przetwarzania obrazu została zaprojektowana z myślą o skutecznej segmentacji naczyń krwionośnych w obrazach siatkówki. Użycie zielonego kanału, filtracji Frangiego oraz szkieletowania to standardowe, dobrze

udokumentowane podejścia w literaturze z zakresu analizy obrazów medycznych. Połączenie tych metod pozwala na zachowanie istotnych struktur przy jednoczesnym ograniczeniu wpływu szumu i artefaktów.

2. Uczenie maszynowe 4.0 - Drzewo decyzyjne

Przygotowanie danych

Aby umożliwić uczenie klasyfikatora, z każdego obrazu siatkówki wydzielono **niezachodzące wycinki (segmenty)** o rozmiarze 15×15 pikseli. Dla każdego segmentu wyznaczono etykietę binarną na podstawie wartości piksela centralnego w masce eksperckiej – etykieta `1` oznacza obecność naczynia, `0` – tło.

Z każdego segmentu wyekstrahowano zestaw **cech statystycznych i morfologicznych**:

- **Moment Hu (7 cech)** – opisuje kształt i symetrię segmentu;
- **Wariancje kanałów RGB (3 cechy)** – pozwalają scharakteryzować zmienność kolorystyczną, przydatną przy odróżnianiu tła od struktur naczyniowych.

Wstępne przetwarzanie zbioru uczącego

Z uwagi na **silną niezbalansowaną naturę danych** (zdecydowana przewaga pikseli tła nad naczyniami), zastosowano metodę **undersamplingu klasy dominującej (tło)** z użyciem `RandomUnderSampler` z biblioteki `imblearn`. Pozwoliło to na zrównoważenie klas i uniknięcie uprzedzeń modelu względem najliczniejszej klasy.

Zastosowane metody uczenia maszynowego

Do klasyfikacji segmentów zastosowano model **drzewa decyzyjnego** (`DecisionTreeClassifier` z `scikit-learn`). Parametry klasyfikatora:

- `max_depth=30` – ograniczenie głębokości drzewa dla uniknięcia nadmiernego dopasowania,
- `min_samples_leaf=5` – minimalna liczba próbek w liściu, co stabilizuje reguły decyzyjne,
- `random_state=42` – zapewnienie powtarzalności wyników.

Dla klasyfikacji nowych obrazów, klasyfikator oceniał każdy segment i przypisywał mu klasę na podstawie prawdopodobieństwa przewyższającego określony **próg decyzyjny (`threshold=0.35`)**.

Wyniki wstępnej oceny klasyfikatora (testy hold-out)

Ocena klasyfikatora została przeprowadzona na pięciu obrazach testowych (podejście *hold-out*). Uzyskano następujące wskaźniki jakości:

Obraz	Accuracy	Sensitivity	Specificity	Harmonic Mean	Arithmetic Mean
01_h.jpg	0.749	0.742	0.750	0.746	0.746
02_h.jpg	0.722	0.769	0.716	0.742	0.743
03_h.jpg	0.750	0.775	0.747	0.761	0.761
04_h.jpg	0.756	0.791	0.753	0.771	0.772
05_h.jpg	0.861	0.617	0.885	0.727	0.751

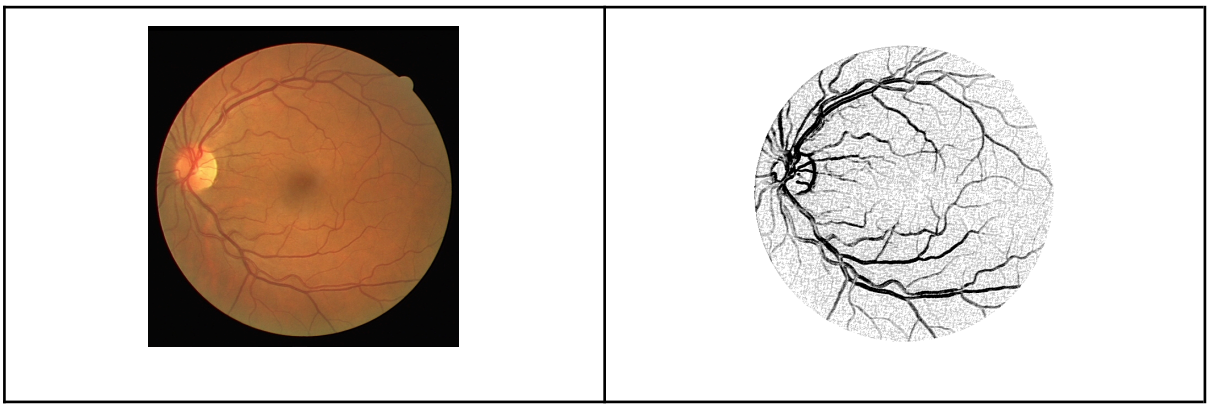
Średnie wartości wskaźników oscylują wokół 75%, co wskazuje na solidną jakość klasyfikatora, szczególnie biorąc pod uwagę prostotę wybranych cech i modelu.

Uzasadnienie zastosowanego rozwiązania

Zastosowanie klasyfikatora drzewa decyzyjnego uzasadnione było potrzebą **transparentności decyzji modelu** oraz łatwej interpretacji wyników. Wybór prostych cech (momenty Hu i wariancje RGB) pozwolił na szybką ekstrakcję informacji lokalnej, wystarczającej do rozróżnienia segmentów zawierających naczynia. Wstępna analiza potwierdza, że nawet przy użyciu podstawowych metod można uzyskać konkurencyjne wyniki w zadaniu binarnej segmentacji naczyń siatkówki.

Opis

Przygotowanie aplikacji (okienkowa, Jupyter notebook, ...), która dla zadanego obrazu wejściowego przedstawiającego dno siatkówki oka (przykład poniżej) automatycznie wykrywa naczynia krwionośnie. Z formalnego punktu widzenia dla każdego piksela wykorzystany algorytm musi stwierdzić, czy ten piksel stanowi naczynie krwionośne, czy nie (klasyfikacja binarna).



Wymagania obowiązkowe

- Algorytm w podstawowej wersji powinien wykorzystywać techniki przetwarzania obrazu do detekcji naczyń krwionośnych. W ramach takiego procesu przetwarzania można wyróżnić 3 główne elementy:

    a) *Wstępne przetworzenie obrazu*: wejściowy obraz może być zaszumiony/zbyt ciemny/jasny. Można tutaj wykorzystać takie techniki jak: rozmycie, wyostrzenie, normalizacja histogramu kolorów itp.
    b) *Właściwe przetworzenie obrazu*: w celu wyodrębnienia naczyń krwionośnych: można zastosować różne techniki wykrywania krawędzi (np. filtr Frangi'ego).
    c) Końcowe przetwarzanie obrazu: przetwarzanie uzyskanego obrazu w celu poprawy skuteczności wykrywania naczyń poprzez "naprawę" błędów z poprzednich kroków.

- Wynik obowiązkowo należy wizualizować np. zamalowując wyróżniającym się kolorem piksele zaklasyfikowane jako naczynie krwionośne. W tym celu najlepiej wygenerować binarną maskę odpowiedzi algorytmu, która zostanie potem wykorzystana do analizy statystycznej (porównanie z maską ekspercką z ręcznie zaznaczonymi naczyniami).

- Ważnym elementem oceny jest skuteczność algorytmu. W tym celu należy dokonać podstawowej analizy statystycznej jakości działania algorytmu. Działanie programu należy przetestować na minimum 5 obrazach. Podczas testów należy wyznaczyć macierze pomyłek oraz takie miary jak *trafność* (*accuracy*), *czułość* (*sensitivity*), *swoistość* (*specificity*). Przy wyznaczaniu czułości i swoistości należy założyć, że naczynie to klasa pozytywna, a tło - negatywna. Ponieważ mamy do czynienia z niezrównoważonym rozkładem klas (punktów tła jest znacznie więcej niż naczyń) należy dodatkowo wykorzystać miary dla danych niezrównoważonych (np. średnią arytmetyczną lub geometryczną czułości i swoistości).

Wymagania na 4.0

- Po wstępnym przetworzeniu obrazu należy podzielić go na wycinki (np. 5x5 px) i dla każdej z nich dokonać ekstrakcji cech z obrazu: np. wariancja kolorów, momenty centralne, momenty Hu itp. Wartości te wraz z informacją pochodzącą z maski eksperckiej (decyzja dla środkowego piksela wycinka) stanowić będą zbiór danych wykorzystany do budowy wybranego klasyfikatora, prostszego niż głęboka sieć neuronowa (np. kNN, drzewo lub las decyzyjny, SVM). Należy skorzystać z gotowej implementacji klasyfikatora (np. w bibliotece scikit-learn).

- Z uwagi na ograniczenia pamięciowe konieczne może być ograniczenie rozmiaru zbioru uczącego poprzez losowy wybór punktów (możliwość zastosowania *undersampling-u* do zrównoważenia rozkładu klas w zbiorze uczącym).

- Zdolności predykcyjne tak opracowanego klasyfikatora należy wstępnie zweryfikować na niezależnym zbiorze testowym *hold-out* (np. pochodzącym z innej części obrazu lub z innego obrazu).
- Gotowy klasyfikator powinien zostać osadzony w aplikacji, a jego działanie powinno zostać zwizualizowane i przetestowane w taki sam sposób, jak działanie technik przetwarzania obrazu z wymagań podstawowych.

Wymagania na 5.0

- Jako model decyzyjny należy wykorzystać głęboką sieć neuronową. W zależności od wybranego rodzaju sieci, może zostać ona nauczona na wycinkach obrazu (podobnie jak w przypadku wymagań na 4.0), jak i na całych obrazach (np. w przypadku sieci UNet). Należy skorzystać z gotowej implementacji sieci (np. w bibliotece Keras, PyTorch lub TensorFlow).
- Zdolności predykcyjne nauczonej sieci neuronowej powinny być wstępnie zweryfikowane na zbiorze testowym *hold-out*.
- Nauczona sieć powinna zostać osadzona w aplikacji i tam dodatkowo przetestowana zgodnie z wymaganiami obowiązkowymi.

Uwaga

Realizując wymagania na 4.0 lub 5.0 należy także zrealizować wymagania obowiązkowe -- wyniki uzyskane za pomocą prostych metod filtrowania obrazu będą stanowić punkt odniesienia (*baseline*) dla bardziej zaawansowanych modeli decyzyjnych. Realizując wymagania na 5.0 należy również zrealizować wymagania na 4.0, aby porównać działanie obu typów klasyfikatorów.

W projekcie należy skorzystać z jednej z dostępnych baz danych z obrazami (patrz linki poniżej) -- ta sama baza powinna być stosowana we wszystkich krokach projektu.

Raport

Raport powinien zawierać następujące elementy (część z nich dotyczy wymagań na 4.0 i 5.0 - zostało to zaznaczone):

1. Skład grupy
2. Zastosowany język programowania oraz dodatkowe biblioteki
3. Opis zastosowanych metod:
    a. przetwarzanie obrazów
        i. poszczególne kroki przetwarzania obrazu (w tym zastosowane filtrów),
        ii. krótkie uzasadnienie zastosowanego rozwiązania,
    b. uczenie maszynowe (4.0/5.0)

i. przygotowanie danych - wyznaczanie wycinków obrazu, ekstrakcja cech z wycinków (jeśli zastosowano)
ii. wstępne przetwarzanie zbioru uczącego (jeśli zastosowano)
iii. zastosowane metody uczenia maszynowego wraz z informacją o przyjętych parametrach
iv. wyniki wstępnej oceny zbudowanego klasyfikatora (testy *hold-out*)
v. krótkie uzasadnienie zastosowanego rozwiązania

4. Wizualizacja wyników działania programu dla wybranych obrazów (warto pokazać zarówno sukcesy jak i porażki). Dla porównania należy zamieścić maskę ekspercką (*ground truth*). Wyniki wizualizacji należy zaprezentować dla co najmniej 5 obrazów (w przypadku metod uczenia maszynowego nie mogą to być obrazy, które zostały wykorzystane do nauczenia klasyfikatora).

5. Analiza wyników działania programu dla wybranych obrazów (tych samych, które wykorzystano w punkcie 4) z wykorzystaniem odpowiednich miar oceny (omawianych wcześniej). Analizę należy przeprowadzić indywidualnie dla każdego z obrazów. W przypadku realizacji zadań na 4.0/5.0 należy dokonać porównania miar oceny osiąganych przez metody przetwarzania obrazów oraz uczenia maszynowego.

TODO:

- ~~automatyczne wyliczanie macierzy pomyłek dla każdego piksela~~
- ~~wyliczanie miary jak *trafność* (*accuracy*), *czułość* (*sensitivity*), *swoistość* (*specificity*), miary dla danych niezrównoważonych (np. średnią arytmetyczną lub geometryczną czułości i swoistości).~~
- ~~na 3: Końcowe przetwarzanie obrazu: przetwarzanie uzyskanego obrazu w celu poprawy skuteczności wykrywania naczyń poprzez "naprawę" błędów z poprzednich kroków.~~

- ~~dzielenie obrazu na wycinki~~

- ekstrakcji cech z obrazu: np. wariancja kolorów, ~~momenty centralne, momenty Hu~~ itp

- ~~budowy wybranego klasyfikatora, prostszego niż głęboka sieć neuronowa (np. kNN, drzewo lub las decyzyjny, SVM). Należy skorzystać z gotowej implementacji klasyfikatora (np. w bibliotece scikit-learn).~~

- ~~ograniczenie rozmiaru zbioru uczącego poprzez losowy wybór punktów (możliwość zastosowania *undersampling-u* do zrównoważenia rozkładu klas w zbiorze uczącym)~~

- zweryfikować klasyfikator na niezależnym zbiorze testowym *hold-out* (np. pochodzącym z innej części obrazu lub z innego obrazu).
- Należy skorzystać z gotowej implementacji sieci (np. w bibliotece Keras, PyTorch lub TensorFlow) do wyuczenia sieci neuronowej
- Zdolności predykcyjne nauczonej sieci neuronowej powinny być wstępnie zweryfikowane na zbiorze testowym *hold-out*.

- sprawozdanie:
    - Opis zastosowanych metod:
        - przetwarzanie obrazów
            - poszczególne kroki przetwarzania obrazu (w tym zastosowane filtrów),
            - krótkie uzasadnienie zastosowanego rozwiązania,
        - uczenie maszynowe (4.0/5.0)

    - przygotowanie danych - wyznaczanie wycinków obrazu, ekstrakcja cech z wycinków (jeśli zastosowano)
    - wstępne przetwarzanie zbioru uczącego (jeśli zastosowano)
    - zastosowane metody uczenia maszynowego wraz z informacją o przyjętych parametrach
    - wyniki wstępnej oceny zbudowanego klasyfikatora (testy *hold-out*)
    - krótkie uzasadnienie zastosowanego rozwiązania

- Wizualizacja wyników działania programu dla wybranych obrazów (warto pokazać zarówno sukcesy jak i porażki). Dla porównania należy zamieścić maskę ekspercką (*ground truth*). Wyniki wizualizacji należy zaprezentować dla co najmniej 5 obrazów (w przypadku metod uczenia maszynowego nie mogą to być obrazy, które zostały wykorzystane do nauczenia klasyfikatora).

- Analiza wyników działania programu dla wybranych obrazów (tych samych, które wykorzystano w punkcie 4) z wykorzystaniem odpowiednich miar oceny (omawianych wcześniej). Analizę należy przeprowadzić indywidualnie dla każdego z obrazów. W przypadku realizacji zadań na 4.0/5.0 należy dokonać porównania miar oceny osiąganych przez metody przetwarzania obrazów oraz uczenia maszynowego.