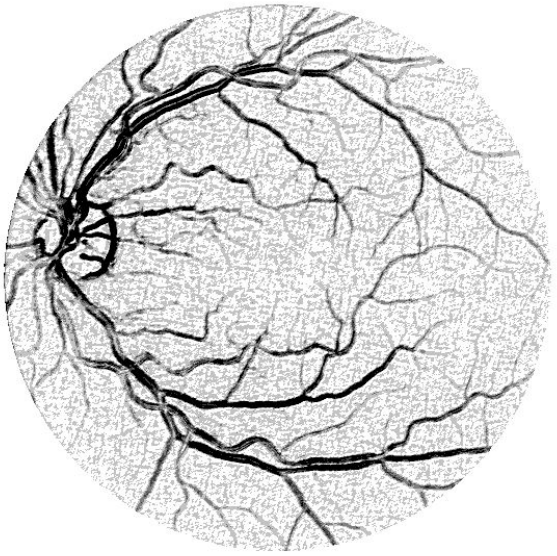
Wykrywanie naczyń dna siatkówki oka

Słowa kluczowe: przetwarzanie obrazów, uczenie maszynowe

Opis

Należy napisać aplikację (okienkowa, notebook), która dla zadanego obrazu wejściowego przedstawiającego dno siatkówki oka (przykład poniżej), wykrywa (automatycznie) 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 (poznane między innymi na przedmiocie KCK – zadanie z samolotami/projekt z obrazów) 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 (porównania 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*) oraz ich warianty przeznaczone dla danych niezrównoważonych (np. średnia arytmetyczna lub geometryczna czułości i swoistości).

Wymagania na 4.0

- Po wstępnym przetworzeniu obrazu należy podzielić go na niewielkie części (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 (decyzja dla środkowego piksela) stanowić będą zbiór danych wykorzystany do budowy prostego klasyfikatora odległościowego (Rocchio, kNN).
- Z uwagi na ograniczenia pamięciowe konieczne może być ograniczenie rozmiaru zbioru testowego poprzez losowy wybór punktów (możliwość zastosowania technik *resamplingu*).
- Trafność klasyfikacji tak opracowanego klasyfikatora należy zweryfikować na niezależnym zbiorze testowym (np. pochodzącym z innej części obrazu lub z innego obrazu).

Wymagania na 5.0

- Przygotowanie danych takie samo, jak w przypadku wymagań na 4.0. Należy jednak wykorzystać bardziej zaawansowany klasyfikator, np.: sieć neuronowa, drzewo decyzyjne lub las, reguły decyzyjne. Można wykorzystać gotowe implementacje klasyfikatorów (scikit-learn, WEKA, ...).
- Należy wykorzystać k-krotną walidację skrośną (k-fold cross validation) w celu oceny zbudowanego klasyfikatora i uniknięcia przeuczenia.

Uwaga

Realizując wymagania na 4.0 i 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.

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

Linki

- Baza obrazów HRF: https://www5.cs.fau.de/research/data/fundus-images/
- Baza obrazów STARE: http://cecas.clemson.edu/~ahoover/stare/probing/
- Baza obrazów CHASE: https://staffnet.kingston.ac.uk/~ku15565/CHASE_DB1/