Specyfikacja funkcjonalna

Program do zliczania komórek na zdjęciach mikroskopowych

Paweł Skiba, 290997

29 marca 2021

Spis treści

1 Opis ogólny

1.1 Nazwa programu

Wytworzony program zostanie nazwany w języku angielskim, w następujący sposób - *biological_cells_counter*. Ze względu na fakt, iż program zostanie zaimplementowany w języku *Pyhton*, pełna nazwa programu z roszerzeniem to *biological_cells_counter.py*. Nazwę programu możemy przetłumaczyć jako *licznik komórek biologicznych*, co oddaje całkowicie cel realizacji tego projektu.

1.2 Opis problemu

Poruszanym problemem jest zagadnienie dotyczące uzyskiwania informacji z obrazu, a dokładnie zliczania komórek na zdjęciach mikroskopowych. Dyscyplina ta jest bardzo dobrze rozwinięta, dzięki czemu możliwe jest podejście do problemu na różne sposoby. Jednym z takich podejść jest cyfrowe przetwarzanie obrazów, które obejmuje różne zagadnienia, takie jak filtracja, binarazycja, segmentacja, zmiana przestrzeni barw, czy morfologia matematyczna. Dzięki takiemu podejściu obraz może zostać przetworzony, a następnie obrobiony w celu uzyskania intersujących nas wyników, tzn. liczby komórek na obrazie. Innym podejściem do rozwiązania tego typu problemu jest uczenie głębokie, a dokładnie sztuczne sieci neruronowe, które imitują zachowanie ich biologicznego odpowiednika. To podejście wymaga procesu uczenia, które polega na przygotowaniu danych w taki sposób, aby sieć neuronowa otrzymała jednocześnie wejście - oryginalny obraz oraz wyjście - informację o obiektach interesujących sieć. W trakcie uczenia, sieć modyfikuje w taki sposób swoje współczynniki, aby po jej wytrenowaniu, była ona w stanie wykrywać interesujące nas obiekty na obrazie.

1.3 Zastosowanie rozwiązania

Kolejny aspekt poruszany w tym projekcie to praktyczne zastosowanie takiego rozwiązania. Dobrze zbudowany algorytm, o dużej dokładności może być wykorzystywany przez specjalistów, w celu wspomagania ich codziennej pracy. Takie rozwiązanie może posłużyć do diagnostyki różnych materii, np. w celu wyznaczenia gęstości.

1.4 Cel projektu

Celem projektu jest zbudowanie uniwersalnego programu, który będzie w stanie zliczyć komórki na obrazie z dużą dokładnością. Uniwersalność programu ma polegać na tym, że obrazy mogą dotyczyć różnorodnych komórek. Jest to związane również z tym, że obrazy mogą być uzyskiwane w różny sposób oraz mogą cechować się innymi parametrami, takimi jak wymiary obrazu, czy przestrzeń barw.

2 Dane

2.1 Opis danych

W projekcie zostanie wykorzystany zbiór obrazów *BBBC038v1*, dostępnych z kolekcji *Broad Bioimage Benchmark Collection*. Zbiór ten zawiera bardzo dużo zdjęć mikroskopowych organizmów żywych o szerkim kontekście. Obrazy znajdujące się w zbiorze są w odcieniach szarości oraz w kolorze. W pracy zostaną wykorzystane dwa zestawy danych z wymienionego wyżej zbioru danych. Będą to następujące zestawy:

- stage1_train - zestaw ten zawiera 671 różnych obrazów, zostanie on wykorzystany w przypadku klasycznego przetwarzania obrazów oraz w przypadku sieci neuronowych, gdzie posłuży jako zbiór treningowy,
- stage1_test - zestaw ten zawiera 66 różnych obrazów, co stanowi ok 9% wszystkich obrazów, posłuży on jako zbiór testowy w przypadku sieci neuronowych.

Dopuszczana jest możliwość ograniczenia wspomnianego wyżej zbioru danych do określonych cech, takich jak przestrzeń barw.

2.2 Struktura katalogów

Każdy ze wspomnianych powyżej zestawów danych, cechuje się inna strukturą katalogów. W przypadku zestawu *stage1_train* wygląda to w następujący sposób dla jednego przypadku:

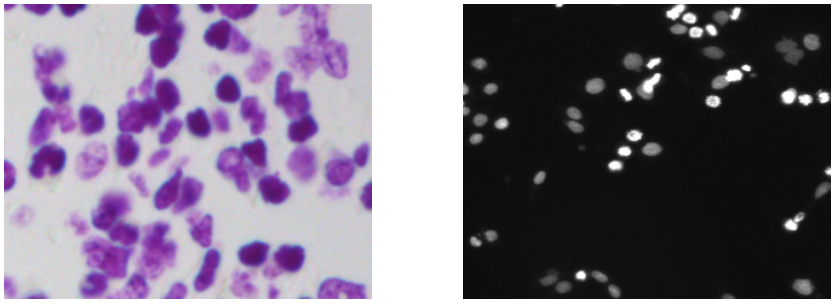
- identyfikator folderu
  - images
    - obraz
  - masks
    - obrazy masek

Natomiast w przypadku zestawu danych *stage1_test* nie ma katalogu zawierającego maski, a struktura katalogów zaprezentowana została poniżej:

- identyfikator folderu
  - images
    - obraz

2.3 Przykładowe obrazy

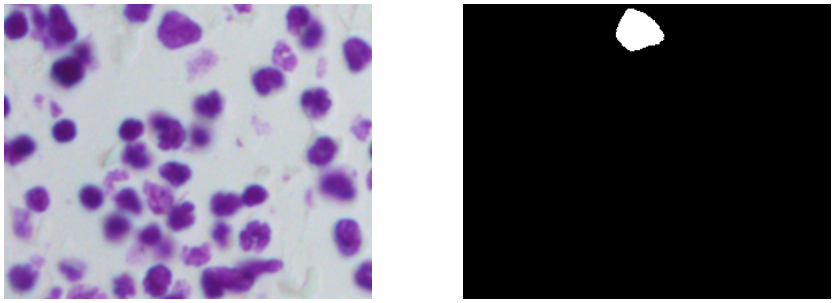
Na Rysunku 1. zostały zaprezentowane przykładowe obrazy, które możemy spotkać w opisywanym zbiorze. Obrazy te pochodzą z zestawu *stage1_train*. W zestawie tym możemy znaleźć bardzo wiele różnych obrazów o odmiennych cechach, a poniższe zdjęcia są jedynie przykładami.



(a) Obraz kolorowy (b) Obraz w odcieniach szarości

Rysunek 1: Przykładowe obrazy ze zbioru treningowego.

Na Rysunku 2. zostały zaprezentowane dwa obrazy. Pierwszy z nich przedstawia oryginalny obraz ze zbioru, natomiast drugi z nich przedstawia jedną, wybraną losowo maskę. Maska reprezentuje jedną wysegmentowaną komórkę z obrazu oryginalnego. W przypadku przedstawionego obrazu, w zestawie znajdują się 62 maski dla tego obrazu.



(a) Obraz oryginalny (b) Wybrana maska

Rysunek 2: Przykładowy obraz oryginalnego obrazu oraz wybranej maski ze zbioru treningowego.

3 Możliwości programu

3.1 Cyfrowe przetwarzanie obrazów

Korzystanie z programu będzie możliwe na dwa sposoby, które zostały wstępnie omówione w sekcji *Opis problemu*. Pierwszą z tych metod będzie cyfrowe przetwarzanie obrazów. Jest to bardzo dobrze rozwinięta dyscyplina, która pozwala modyfikować obrazy na poziomie pojedynczych pikseli. W szczególności zostaną wykorzystane elementy morfologii matematycznej. Program działający w tym trybie będzie potrzebował jedynie ścieżki do pliku, która będzie podawana jako argument uruchomienia programu.

3.2 Sieci neuronowe

Kolejnym sposobem korzystania z programu będzie wyorzystanie sztucznej sieci neuronowej. W tym przypadku program będzie mógł działać w dwóch trybach. Pierwszy z nich będzie służył do zliczania komórek, podobnie jak w przypadku cyfrowego przetwarzania obrazów. Drugi tryb będzie niezbędny do wytrenowania modelu, który jest nieodzownym elementem tej dziedziny. Dzięki temu program będzie miał dodatkową funkcjonalność, która pozwoli w pełni dostosować program do indywidualnych potrzeb, poprzez wytrenowanie sieci neuronowej na własnym zbiorze danych zawierającym obrazy oryginalne oraz maski do każdego z nich.

4 Użytkowanie programu

4.1 Typ programu

Program zostanie zaimplementowany w języku *Python*, natomiast będzie on uruchamiany z poziomu wiersza poleceń. Dodatkowo program będzie przenośny między popularnymi systemami operacyjnymi, pochodzącymi z rodziny *Linux/Unix* oraz systemami z rodziny *Microsoft Windows*.

4.2 Oczekiwany rezultat

W przypadku, zliczania komórek na obrazie, program wypisze na ekran następującą treść:

```
The image contains 32 cells.
```

gdzie 32 jest przykładową liczbą.

Program dodatkowo może wyświetlać poszczególne etapy realizacji procesu zliczania komórek, bądź procesu uczenia sieci neuronowej, w celu poinformowania użytkownika o dotychczsowym stanie programu.

4.3 Uruchomienie programu

Poniżej zostanie przedstawione przykładowe uruchomienie programu z uwzględnieniem systemu operacyjnego. W przypadku systemu operacyjnego z rodziny *Linux/Unix* program należy uruchomić w następujący sposób:

```
python3 biological_cells_counter.py -file path_to_file
```

Natomiast w przypadku systemu operacyjnego z rodziny *Microsoft Windows* będzie to wyglądało następująco:

```
python biological_cells_counter.py -file path_to_file
```

Zarówno w pierwszym, jak i drugim przypadku, podczas zliczania komórek, niezbędna jest ścieżka do pliku, bez której program zakończy swoje działanie bez wykonania głównego zadania. Domyślnie program będzie zliczał komórki z wykorzystaniem algorytmu zaimplementowanego w oparciu o cyforwe przetwarzanie obrazów.

4.4 Parametry programu

W celu wykorzystania wszystkich możliwości dostarczonych do programu należy używać flag zgodnie z ich przeznaczeniem, w sposób zademonstrowany w sekcji *Uruchomienie programu*. W programie będziemy mogli użyć następujących flag:

- -file – za pomocą tej flagi wskazywany jest plik z obrazem zawierającym komórki do zliczenia,
- -output – za pomocą tej opcjonalnej flagi wskazywana jest ścieżka i plik do którego ma zostać zapisany obraz z wysegmentowanymi komórkami,
- -mode – za pomocą tej flagi ustawiany jest tryb, w którym będą zliczane komórki, użytkownik jako argument może podać następujące wartości:
  - dip – [domyślne ustawienie] skrót od nazwy *cyfrowe przetwarzanie obrazów* (*ang. digital image processing*),
  - nn – skrót od nazwy *sieć neuronowa* (*ang. neural network*),
  - tnn – skrót od nazwy *trenowanie sieci neuronowej* (*ang. training neural network*),
- -params – za pomocą tej flagi przekazywane są niezbędne parametry do uczenia sieci, takie jak liczba epok, katalog z obrazami oryginalnymi czy katalog z maskami. Należy pamiętać, aby argumenty zostały przekazane w cudzysłowie, argument ten jest wymagany w przypadku, gdy flaga *mode* przyjmuje wartość *tnn*.
- -h lub –help – za pomocą tych flag, użytkownik będzie miał możliwość wyświetlenia opcji oraz ich krótkich opisów, które są dostępne w programie.

4.5 Obsługa błędów

Program będzie w stanie obsłużyć podstawowe błędy, które mogą powstać w przypadku błędnego korzystania z programu przez użytkownika. Poniżej zostaną przedstawione obsługiwane przez program błędy:

1. Brak niezbędnego argumentu.
   W przypadku braku niezbędnego argumentu, program poinformuje użytkownika za pomocą wypisania na ekran następującej informacji:

   ```
   Missing argument: argument
   ```

   Jako brakujący argument można przyjąć niezdefiniowanie argumentu *parms*, gdy program ma pracować w trybie uczącym [-*mode tnn*] lub brak argumentu [-*file path_to_file*], gdy program ma zliczać komórki na obrazie.

2. Niepoprawna flaga.
   W przypadku, gdy użytkownik poda niepoprawną (niezdefiniowaną) flagę, program poinformuje użytkownika za pomocą wypisania na ekran następującej informacji:

   ```
   Unrecognized arguments: argument
   ```

   Taka sytuacja może mieć miejsce, gdy użytkownik użyje flagi [-*f file_to_path*], zamiast flagi [-*file file_to_path*].

3. Niepoprawny tryb.
   W przypadku, gdy użytkownik poda niepoprawny tryb, program poinformuje użytkownika za pomocą wypisania na ekran następującej informacji:

   ```
   Unrecognized mode: argument
   ```

   Możliwe jest tylko wykorzystanie zdefiniowanych trybów, które zostały opisane powyżej [*dip, nn, tnn*]. Użycie każdego innego trybu doprawdzi do wystąpienia błędu.

4. Niepoprawna ścieżka do pliku
   W przypadku, gdy użytkownik poda niepoprawną ścieżkę do pliku, program poinformuje użytkownika za pomocą wypisania na ekran następującej informacji:

   ```
   File not found: path_to_file
   ```

5. Niepoprawne argumenty do uczenia sieci neuronowej.
   W przypadku, gdy użytkownik poda niepoprawne argumenty służące do procesu trenowania sieci neuronowej, program poinformuje użytkownika za pomocą wypisania na ekran następującej informacji:

   ```
   Incorrect arguments to learning: arguments
   ```

6. Zbyt dużo argumentów.
   W przypadku, gdy użytkownik poda zbyt dużą liczbę argumentów, program będzie dopasowywał argumenty do trybu w którym pracuje, a nadmiarowe argumenty nie zostaną wykorzystane.