Politechnika Wrocławska

Wydział Elektroniki, Fotoniki i Mikrosystemów

KIERUNEK: Automatyka i Robotyka (AIR)

PRACA DYPLOMOWA INŻYNIERSKA

TYTUŁ PRACY:

Aplikacja webowa zwiększająca rozdzielczość obrazów

AUTOR:

Eryk Wójcik

PROMOTOR:

dr hab. inż. Andrzej Rusiecki,
Katedra Informatyki Technicznej

WROCŁAW 2023

Spis treści

Rozdział 1

Wstęp

Cel pracy

Opis celu badań, czyli stworzenia aplikacji webowej służącej do zwiększania rozdzielczości obrazów z użyciem algorytmów ESRGAN i DWSR oraz analiza i porównanie tych algorytmów.

Zakres pracy

Przedstawienie koncepcji i zagadnień, które zostaną omówione w pracy, w tym wybrane metody i technologie.

Rys 1. Diagram przepływu użytkownika

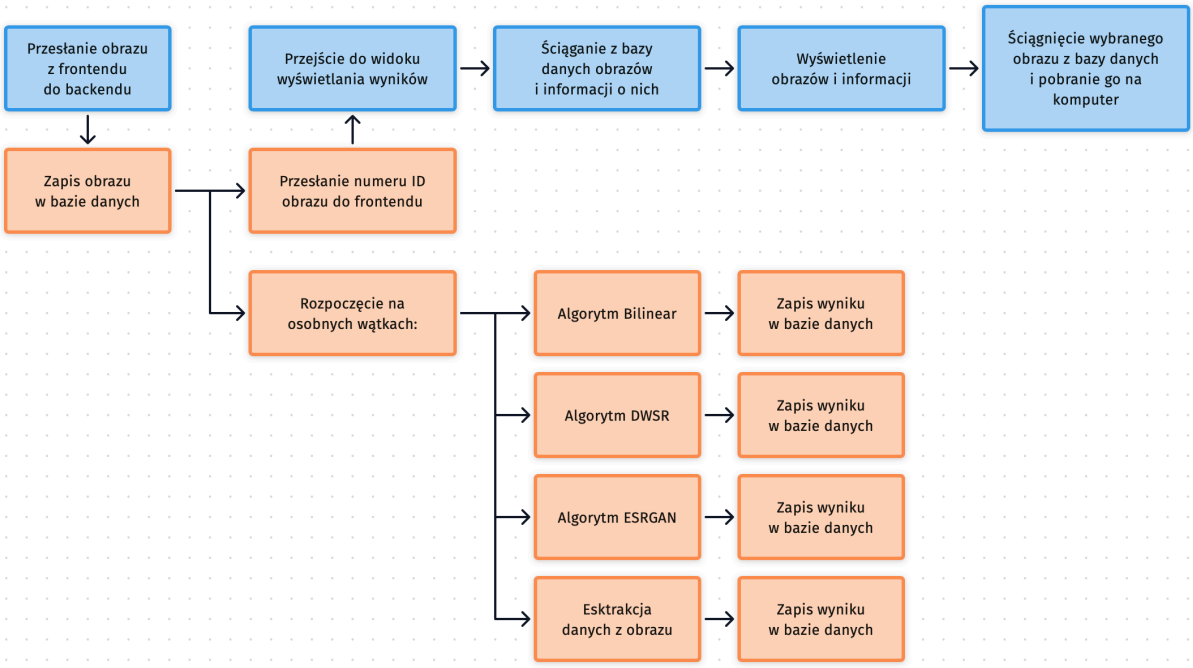
TODO: Usuń diagram przepływu

1.1 Implementacja aplikacji

Po wyborze stosu technologicznego kolejnym krokiem jest skupienie się na implementacji rozwiązań. W tym rozdziale opiszę jakie decyzje podjąłem przy pisaniu kodu aplikacji, jak wygląda jej struktura i jakie problemy napotkałem podczas implementacji.

Struktura aplikacji

Aplikacja składa się z dwóch części - Frontendu i Backendu. Przy tworzeniu takiego projektu warto zadbać o to, żeby każda część była od siebie niezależna i żeby komunikacja między nimi była jak najmniej skomplikowana.
W tym miejscu wracamy do diagramu przepływu użytkownika [Rys 1], jak na nim widać użytkownik nie może wykonać zbyt wiele akcji, struktura aplikacji jest liniowa. Na podstawie diagramu przepływu użytkownika można stworzyć schemat blokowy aplikacji [Rys 2], który pozwoli zrozumieć zachowanie programu.



Rys 2. Schemat blokowy aplikacji (kolor niebieski - Frontend, pomarańczowy - Backend)

W pierwszej kolejności użytkownik wysyła obraz do serwera Backend, który zapisuje go w bazie danych. Następnie serwer zleca wykonanie algorytmów na osobnych wątkach, o czym opowiem w dalszej części rozdziału [1.2]. Gdy algorytmy rozpoczną pracę, serwer zwraca do Frontendu informację o tym że operacja zapisu się powiodła i podaje numer ID obrazu.

Frontend zmienia widok na ten z wynikami i wysyła zapytanie do Backendu o obraz oryginalny i przetworzone. Następnie jeśli serwer zwróci obrazy, Frontend je wyświetla. W przeciwnym wypadku próbuje je pozyskać ponownie aż do skutku. Dzieje się tak, dlatego że zadanie super-rozdzielczości jest czasochłonne i czasem może zająć kilka sekund a w innych wypadkach nawet kilka minut, wszystko w zależności od rozdzielczości obrazów. W tym czasie użytkownik może porównywać uzyskane wyniki i wybrać najlepszy.
Gdy użytkownik wybierze obraz, może go pobrać na swój komputer. Wtedy Frontend wysyła zapytanie do serwera o obraz w pełnej rozdzielczości, a serwer zwraca obraz wraz z nagłówkiem *Content-Disposition: attachment* co powoduje, że przeglądarka automatycznie pobiera obraz na dysk użytkownika.

Architektura bazy danych

Baza danych w aplikacji jest bardzo prosta, przy przesłaniu każdego zdjęcia w bazie tworzone jest pole Image, które przechowuje informacje o obrazie oraz jego przetworzonych wersjach. W tabeli 1.1 przedstawiam strukturę bazy danych.

Image		
Pole	**Typ**	**Opis**
image	ImageField	Przesłany obraz.
bilinear_image	ImageField	Obraz powiększony algorytmem Bilinear.
dwsr_image	ImageField	Obraz powiększony algorytmem DWSR.
esrgan_image	ImageField	Obraz powiększony algorytmem ESRGAN.
original_height	PositiveIntegerField	Wysokość oryginalnego obrazu.
original_width	PositiveIntegerField	Szerokość oryginalnego obrazu.
dominant_colors	TextField	Pole tekstowe z listą dominujących kolorów.

Tabela 1.1: Struktura bazy danych - Image.

Jak widać w bazie danych przechowywane są obrazy w formacie *ImageField*, który jest dostarczany przez bibliotekę Django. Jest to pole, które przechowuje ścieżkę do pliku na dysku serwera. Zapisujemy również informacje o oryginalnych wymiarach obrazu, które są wyświetlane użytkownikowi przez Frontend.
Dodatkowo w bazie danych przechowujemy listę dominujących kolorów, które są wykrywane przez algorytm K-średnie *K-means*, o którym opowiem w kolejnym rozdziale [1.2]. Jest to lista kolorów w formacie HEX, które wykorzystuje Frontend do wyświetlenia kolorowych gradientów tła.

1.2 Integracja algorytmów super-rozdzielczości

```
def upload_image(request):
try:
    form = Image(image=request.FILES['image'])
    form.save()

    input_image_path = form.image.path

    thread1 = threading.Thread(target = run_bilinear,
                               args = (input_image_path, 4, form))
    thread2 = threading.Thread(target = run_dwsr,
                               args = (input_image_path, 4, form))
    thread3 = threading.Thread(target = run_esrgan,
                               args = (input_image_path, form))
    thread4 = threading.Thread(target = extract_image_info,
                               args = (input_image_path, form))

    thread1.start()
    thread2.start()
    thread3.start()
    thread4.start()

    image = Image.objects.latest('id')  # Gets the latest entry

    return JsonResponse({'message': 'Image uploaded, processing started',
                         'image_id': image.id})

except Exception as e:
    return JsonResponse({'error': str(e)}, status=400)
```

Listing 1.1: Obsługa zapisu i przetwarzania obrazów (Django).

1.3 Testowanie aplikacji

1.4 Wdrożenie i utrzymanie aplikacji

Omówienie procesu wdrożenia gotowej aplikacji oraz planów dotyczących jej przyszłego utrzymania i aktualizacji.

1.5 Plany na przyszłość

Opis błędów, rzeczy do poprawy w aplikacji. Omówienie jakie są plany rozbudowy aplikacji.
Przyciski nawigacji z widoku do widoku

Rozdział 2

Porównanie algorytmów ESRGAN i DWSR

[1]

2.1 Kryteria porównawcze

Ustalenie kryteriów, które będą stosowane do oceny i porównania skuteczności i efektywności algorytmów super rozdzielczości.

2.2 Analiza wydajności

Bezpośrednie porównanie wydajności obu metod w różnych warunkach, bazujące na ustalonych kryteriach.

2.3 Jakość odtwarzania obrazów

Ocena jakości obrazów generowanych przez oba algorytmy, uwzględniając różne aspekty jakości wizualnej.

2.4 Ograniczenia i wyzwania

Dyskusja na temat ograniczeń obu metod i potencjalnych wyzwań w ich stosowaniu.

Rozdział 3

Podsumowanie i wnioski

3.1 Dyskusja wyników

Krytyczna analiza uzyskanych wyników w kontekście celów pracy oraz istniejących badań i literatury w dziedzinie.

3.2 Rekomendacje i kierunki dalszych badań

Sugestie dotyczące potencjalnych ulepszeń i obszarów, które wymagają dalszych badań, w oparciu o obserwacje i wyniki badań.

TODO: Odkomentuj TODO: dodaj ref do wzorów

Literatura

[1] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, C. C. Loy. Esrgan: Enhanced super-resolution generative adversarial networks. *The European Conference on Computer Vision Workshops (ECCVW)*, September 2018.