Analiza obrazów

Projekt: rozpoznawanie tekstu ze zdjęcia

Piotr Kowalczyk

Maciej Dąbrowski

Zuzanna Sulima

Spis treści:

1. Tytuł i autorzy projektu

Wykonano projekt pod tytułem „Rozpoznawanie tekstu ze zdjęcia”, którego autorami są:

- Piotr Kowalczyk
- Maciej Dąbrowski
- Zuzanna Sulima

2. Opis projektu

Celem projektu było stworzenie programu umożliwiającego rozpoznawanie tekstu zawartego na wgranym obrazie. Użytkownik może wgrać dowolny obraz zawierający tekst, a następnie przygotować go do rozpoznawania. Otrzymany przetworzony obraz, może przekonwertować na tekst.

3. Założenia wstępne przyjęte w realizacji projektu

Kierując się wcześniej zdobytym doświadczeniem do wykonania projektu zastosowano program MATLAB. Powstała aplikacja umożliwia rozpoznawanie tekstu zawartego na obrazie wczytanym przez użytkownika.

- Wczytany obraz powinien mieć dobrą rozdzielczość
- Litery na obrazie nie powinny się zlewać
- Wczytany obraz powinien być cyfrowy lub mieć duży kontrast między kartką i tekstem.

4. Analiza projektu

4.1 Specyfikacja danych wejściowych

Danymi wejściowymi przyjmowanymi przez program są obrazy o rozszerzeniu PNG oraz JPG zawierające tekst drukowany.

4.2 Opis oczekiwanych danych wyjściowych

Dane wyjściowe są reprezentowane przez rozpoznany ciąg liter/znaków z obrazu wejściowego.

4.3 Zdefiniowanie struktur danych

W programie są zdefiniowane następujące struktury danych:

- **originalImage –** obiekt reprezentujący oryginalny obraz, wgrany przez użytkownika
- **binaryImage –** zmienna globalna reprezentująca obraz przygotowany do rozpoznawania tekstu
- **objects –** obraz binarny reprezentujący obiekty (linijki tekstu)
- **merged** – obraz, gdzie każdy znak otrzymał unikalną etykietę
- **objectsPosition –** tablica obiektów reprezentująca pozycje obiektów (znaków)
- **labeledObjects –** tablica obiektów reprezentująca etykietowane obiekty
- **fontsList –** tablica zawierająca listę czcionek użytych do trenowania sieci neuronowej
- **charsList –** tablica zawierająca listę znaków użytych do trenowania sieci neuronowej
- **mainWindow** – obiekt reprezentujący interfejs użytkownika i będący kontenerem dla pozostałych elementów interfejsu użytkownika
- **database/** - folder zawierający foldery z obrazami wszystkich znaków wygenerowanymi w celu wytrenowania sieci neuronowej

4.4 Specyfikacja interfejsu użytkownika

Na interfejs użytkownika składają się trzy główne elementy: przycisk „Wybierz zdjęcie”, po kliknięciu którego użytkownik może wybrać obraz, który będzie analizowany przez aplikację, przycisk „Przygotuj zdjęcie”, po kliknięciu którego aplikacja przetwarza oryginalny obraz i przygotowuje go do procesu rozpoznawania tekstu oraz pole tekstowe, w którym rozpoznany tekst jest wyświetlany. Dostępne są również 3 suwaki: „Wrażliwość”, na którym można ustawić próg binaryzacji. „Rotacja” dzięki któremu można ustawić pochylenie obrazu tak, aby tekst na nim znajdujący się był poziomo. „Otwarcie”, którym ustawia się poziom operacji otwarcia. W przypadku wykonywania operacji na słabo oświetlonym zdjęciu, można

zaznaczyć opcję „Złe oświetlenie”. Wtedy obraz zostanie przetworzony z uwzględnieniem tej informacji.

4.5 Wyodrębnienie i zdefiniowanie zadań

Planując sposób realizacji projektu zdecydowano się na podział na mniejsze, niezależne moduły:

- Stworzenie bazy danych do trenowania sieci
- Opracowanie algorytmu do trenowania sieci
- Stworzenie interfejsu użytkownika
- Przygotowanie obrazu do rozpoznawania tekstu
- Opracowanie algorytmu odczytywania z obrazu
- Opracowanie dokumentacji opisującej projekt

4.6 Decyzja o wyborze narzędzi programistycznych

Zdecydowano, aby przy realizacji projektu korzystać z programu MATLAB. Narzędzie to wybrano ze względu na doświadczenie w jego stosowaniu zdobyte podczas tego semestru.

5. Podział pracy i analiza czasowa

Po wspólnym omówieniu sposobu realizacji projektu, zdecydowano o podziale na mniejsze moduły ułatwiające niezależną pracę.

Piotr Kowalczyk	Stworzenie bazy danych do trenowania sieci, Opracowanie algorytmu do trenowania sieci
Maciej Dąbrowski	Stworzenie interfejsu użytkownika, Przygotowanie obrazu do rozpoznawania tekstu, Opracowanie algorytmu odczytywania z obrazu
Zuzanna Sulima	Opracowanie dokumentacji opisującej projekt

6. Opracowanie i opis niezbędnych algorytmów

Algorytm przygotowani obrazu (funkcja **prepareImage**) – używany w celu przetworzenia oryginalnego obrazu na obraz binarny wykorzystywany do rozpoznawania tekstu. Najpierw sprawdzane jest czy obraz jest w pełni biały, jeśli tak, to uznawany jest za cyfrowy i jest zwracany bez dalszej analizy. W przeciwnym wypadku, obraz jest binaryzowany. Następnie, obiekty znajdujące się na obrazie są analizowane przez funkcję objectprops, która zwraca informacje o właściwościach obiektów. Na koniec, na podstawie tych informacji, obraz jest obracany o odpowiedni kąt i wycięte zostają fragmenty poza kartką

A**lgorytm tworzenia bazy obrazów** – używany do utworzenia bazy danych zawierającej podfoldery, w których są zapisywane wygenerowane obrazy ze zniekształconymi literami/znakami dla każdej czcionki z tablicy fontList

Algorytm trenowania sieci – polega na przygotowaniu danych do trenowania przez utworzenie bazy danych oraz podzielenia ich na dane treningowe i walidacyjne. Następnie definiowana jest struktura sieci neuronowej, ustawia się opcje jej treningu i wykonuje trening za pomocą funkcji trainNetwork. Po zakończonym treningu sieć jest zapisywana, a jej dokładność sprawdzana jest na danych walidacyjnych

Algorytm scalania obiektów (funkcja **mergeObjects**) – polega na nadaniu etykiety każdemu znakowi znajdującemu się na obrazie, a następnie na podstawie pozycji obiektów uzyskanych za pomocą funkcji regionprops, sprawdza czy dwa obiekty są blisko siebie. Jeśli tak, scala oba obiekty przypisując jednemu z nich etykietę drugiego

7. Kodowanie

- **prepareImage**– funkcja przygotowująca obraz do rozpoznawania tekstu. Przyjmuje cztery parametry: obraz wejściowy, wartość logiczną mówiącą czy tekst jest za ciemny, poziom wrażliwości progowania oraz poziom operacji otwarcia. Zwraca ona zbinaryzowany obraz
- **OCR** – główna funkcja uruchamiająca okno aplikacji
- **loadImage** – funkcja wywoływana po naciśnięciu „Wybierz zdjęcie”. Otwiera okno dialogowe do wyboru pliku PNG lub JPG i ładuje wybrany obraz
- **loadPreparedImage** – funkcja wywoływana po naciśnięciu „Przygotuj zdjęcie”. Przetwarza oryginalny obraz i wyświetla wynik
- **translateImage** – funkcja wywoływana po naciśnięciu „Konwertuj na tekst”. Rozpoznaje tekst na przygotowanym obrazie i wyświetla wynik w polu

tekstowym

- **rotateImage** – funkcja obracająca obraz o zadany kąt
- **splitEachLabel** – funkcja losowo dzieląca obrazy z bazy danych na obrazy przeznaczone do trenowania sieci oraz na przeznaczone do jej walidacji
- **trainingOptions** – funkcja ustawiająca opcje treningu sieci neuronowej
- **trainNetwork** – funkcja trenująca sieć neuronową na podstawie danych treningowych oraz zdefiniowanych wcześniej architektury sieci oraz opcji treningu
- **randImage** – funkcja generująca obrazy znaków z różnymi zniekształceniami
- **setSensitivity** – funkcja ustawiająca wartość progu binaryzacji („sensitivity") na wybraną przez użytkownika
- **setOpenSize** – funkcja ustawiająca wartość zmiennej „openSize" na wartość wybraną przez użytkownika
- **setBadLIghting** – funkcja ustawiająca wartość zmiennej „badLighting" na true lub false w zależności od wyboru użytkownika
- **waitbar –** funkcja tworząca pasek postępu podczas trwania przetwarzania obrazu oraz konwersji na tekst, w celu poinformowania użytkownika, że należy poczekać aż proces się skończy
- **openSize** – zmienna określająca rozmiar kropki, która jest używana do usunięcia szumów z obrazu
- **badLighting** – zmienna określająca czy obraz jest zdjęciem o złym oświetleniu
- **imdsTrain** – zmienna przechowująca obrazy do trenowania sieci
- **imdsValidation** – zmienna przechowująca obrazy do walidacji sieci
- **gridLayout** – obiekt reprezentujący siatkę dla elementów interfejsu użytkownika. Jest odpowiedzialny za wymiary oraz rozmieszczenie elementów na ekranie
- **layers** – obiekt definiujący architekturę sieci neuronowej
- **fontsList** – tablica zawierająca listę czcionek użytych do trenowania sieci neuronowej
- **charsList** – tablica zawierająca listę znaków użytych do trenowania sieci neuronowej
- **database** – folder zawierający podfoldery z wygenerowanymi obrazami dla każdego znaku
- **originalImage** – obiekt reprezentujący oryginalny obraz wczytany przez użytkownika
- **preparedImage** – obiekt reprezentujący obraz po przetworzeniu przez funkcję prepareImage
- **textArea** – obiekt reprezentujący pole tekstowe, w którym wyświetlany jest rozpoznany tekst

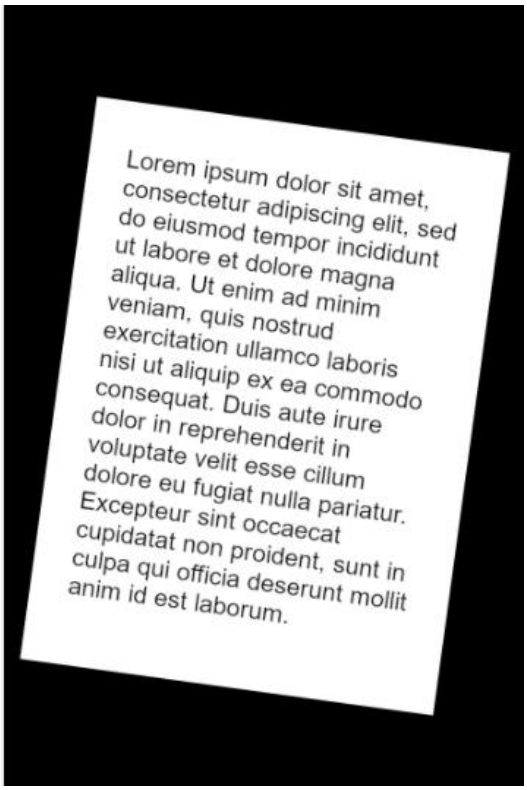
- **selectButton, prepareButton, converButton** – obiekty reprezentujące przyciski wywołujące funkcje po naciśnięciu
- **sliderSensitivity, sliderRotation, sliderOpen** – suwaki, którymi użytkownik może ustawić wartości zmiennych „sensitivity”, „openSize” oraz kąt rotacji
- **checkboxBadLighting –** pole wyboru czy wczytane zdjęcie ma złe oświetlenie
- **regionprops** – funkcja zwracająca informacje/właściwości regionów znalezionych na obrazie binarnym przekazanym jako argument
- **to128Image** – funkcja skalująca obrazy do rozmiaru 128x128
- **mergeObjects–** funkcja przyjmująca obraz binarny z obiektami (znakami) i scalająca nachodzące na siebie obiekty, co pozwala na łatwiejsze przetwarzanie tekstu

8. Testowanie

Program testowano manualnie, sprawdzając kolejne zaimplementowane funkcjonalności.
Sprawdzono poprawność działania wczytania obrazu z pliku PNG lub JPG, przygotowanie zdjęcia do rozpoznawania tekstu oraz konwersję obrazu na tekst. Testowano przetwarzanie oraz konwersję obrazów dla różnych przykładowych zdjęć oraz przy różnych ustawieniach suwaków. Na podstawie tego, przygotowano przyjęte założenia wstępne, opisujące jaki powinien być obraz wejściowy, aby poprawnie rozpoznano tekst.

9. Wdrożenie, raport i wnioski

Stwierdzono, że wczytywanie zdjęcia jest wykonywane bezproblemowo. Zachowując przyjęte założenia wstępne, przy poprawnym ustawieniu parametrów oraz dobrej rozdzielczości rozpoznawanie tekstu działa całkiem dobrze. Nie jest ono niestety idealne i zdarza się, że pojedyncze litery są rozpoznawane niepoprawnie lub małe litery są mylone z wielkimi.



Obraz wczytany w celu rozpoznania tekstu

Lnrom jpSum dOlOr Sjt amet'
COnSeCtetUr adipisCing eljt! sgd
dO ejusmOd tempOf jncididuQt
ut labOFe et dOlOre magna
aljqua. Ut epjm ad mjnlm
VOnjam' qUjs nOSlrud
eXgrcitatjOn u!lamOC !aboriS
njSi ut aljqujp eX ea cOmmOdO
cOGSsquat' DujS aute irure
dOlor jn Feprehenderit in
VOluptate Veljt esse Cjllum
dGlOfg eu fugiat nulla parjatuF.
EXceptsuF sint OCcaeCa!
cupjdatat nCn prOldentl SLInt jn
Cu!pa qui Orj0ia deSeFunt mn!ljl
anim id sSt laborUm.

Tekst rozpoznany przez aplikację