Dokumentacja projektu z przedmiotu Analiza Obrazów

Optyczne rozpoznawanie znaków - rozpoznawanie logo firm

Jan Kwiatkowski, Aleksander Kopyto, Helena Jońca, Julia Przeździk

January 26, 2025

1 Założenia projektu

Celem projektu było utworzenie aplikacji umożlwiającej optyczne rozpoznawanie znaków, a dokładniej logo firm. System pozwala na wczytanie obrazu zawierającego tekst, który zostaje przetworzony w edytowalny format cyfrowy. Aplikacja została zaprojektowana w MATLAB.

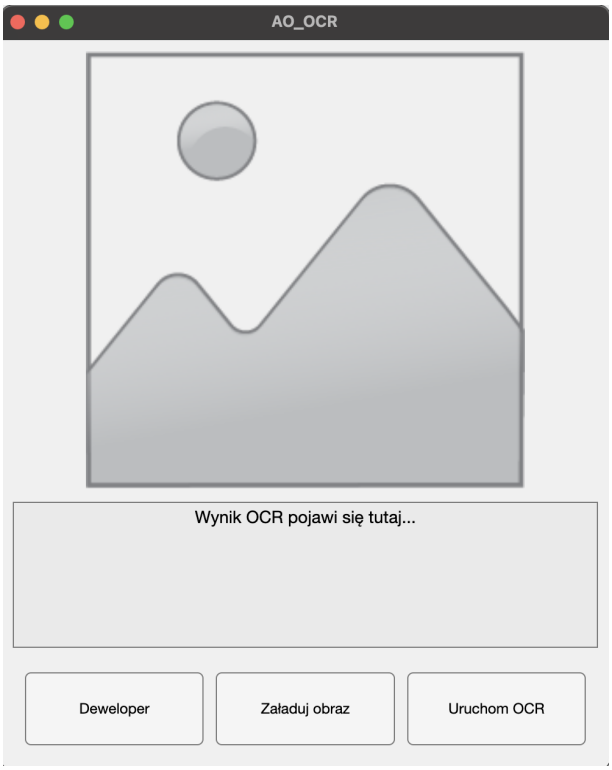
2 Opis projektu

Projekt składa się z graficznego interfejsu użytkownika, modułu przetwarzania obrazów oraz procesora OCR.

3 Interfejs użytkownika

Umożliwia interakcję użytkownika z systemem i zawiera następujące komponenty:

- Okno wyświetlające obraz wczytany przez użytkownika do aplikacji
- Pole tekstowe wyświetlające wyniki po przetworzeniu obrazu
- Przycisk "Deweloper" umożliwiający wybór opcji: trenowanie sieci oraz załadowanie modelu
- Przycisk pozwalający na załadowanie obrazu do apliakcji z własnego urządzenia
- Przycisk uruchamiający OCR



Główne okno aplikacji

Budowa interfejsu

Interfejs składa się z klasy **ORCGUI** , która została wykonana przy pomocy funkcji **uifigure**; jej zadaniem jest utworzenie głównego okna aplikacji, które podzielone na segmenty (wyświetlanie obrazu, obszar tekstowy prezentujący wynik działania OCR oraz układ przycisków) oraz obsłużenie ładowania obrazu do aplikacji. Klasa **DevWindow**, posiada przycisk trenowania (typu uibutton) oraz przycisk ładowania modelu (typu uibutton).

4 Sieć neuronowa

Sieć neuronowa użyta w projekcie OCR została zaprojektowana jako sieć konwolucyjna, dzięki czemu możliwe jest stopniowe filtrowanie różnych częsci danych uczących i wyodrębnienie cech, które umożliwią rozpoznawanie i klasyfikację wzorców. Posiada następującą budowę:

Warstwa wejściowa:
- **Rozmiar:** 32×28 pikseli, 1 kanał (skala szarości).
- Wprowadza obrazy 2D do sieci neuronowej i stosuje normalizację danych.

Pierwsza warstwa konwolucyjna:
- Warstwa typu **convolution2dLayer**
- Posiada 32 filtry
- **Funkcja:** Ekstrahuje podstawowe cechy, takie jak krawędzie i linie liter.

Warstwa normalizacji wsadowej:
- Warstwa typu **batchNormalizationLayer**
- **Funkcja:** Normalizuje dane wejściowe przed przekazaniem ich do kolejnych warstw.

Funkcja Aktywacji ReLU (1): • Warstwa typu **reluLayer**

- **Funkcja:** Stosuje funkcję ReLu, która operację progową dla każdego elementu wejścia; każda wartość mniejsza od zera jest ustawiana na zero.

Max Pooling: • Warstwa typu **maxPooling2dLayer**

- **Funkcja:** Dzieli dane wejściowe na prostokatne obszary poolingu oraz oblicza maksimum każdego z tych obszarów.

Druga Warstwa Konwolucyjna: • Warstwa typu **convolution2dLayer**

- **Funkcja:** Ekstrapoluje bardziej złożone cechy liter.

Batch Normalization 2: • Warstwa typu **batchNormalizationLayer**

- **Funkcja:** Stabilizuje aktywacje po drugiej warstwie konwolucyjnej.

Funkcja Aktywacji ReLU (2): • Warstwa typu **reluLayer**

- **Funkcja:** Wprowadza nieliniowość.

Max Pooling: • Warstwa typu **maxPooling2dLayer**

- **Funkcja:** Dalsza redukcja wymiarowości i konsolidacja cech.

Fully Connected Layer (1): • Warstwa typu **fullyConnectedLayer**

- **Funkcja:** Mnoży dane wejściowe przez macierz wag, a później dodaje wektor odchylenia.

Funkcja Aktywacji ReLU (3): • Warstwa typu **reluLayer**

- **Funkcja:** Wprowadza nieliniowość.

Fully Connected Layer (2): • Warstwa typu **fullyConnectedLayer**

- **Funkcja:** Przygotowuje dane do ostatecznej klasyfikacji na 36 klas (litery A-Z i cyfry 0-9).

Softmax Layer: • Warstwa typu **softmaxLayer**

- **Funkcja:** Przekształca wyjścia warstwy w prawdopodobieństwa dla każdej z 36 klas.

Classification Output: • **Funkcja:** Porównuje przewidywane etykiety z rzeczywistymi etykietami podczas treningu, umożliwiając optymalizację modelu.

Proces Działania Sieci

Na początku binarne obrazy liter są skalowane do rozmiaru 32x28 pikseli i wprowadzane do sieci. Dwie warstwy konwolucyjne oraz warstwy poolingowe wydobywają podstawowe i złożone cechy w budowie liter. Warstwy "Batch Normalization" stabilizują dystrybucję danych wejściowych do kolejnych warstw. Warstwy w pełni połączone integrują wyekstrahowane cechy, przygotowując je do klasyfikacji. Ostatecznie warstwa Softmax generuje prawdopodobieństwa dla każdej z 36 klas, wybierając klasę o najwyższym prawdopodobieństwie jako wynik klasyfikacji.

5 Procesor OCR

Klasa OCRProcessor służy do rozpoznawania liter na obrazach tekstowych. Przetwarza obraz zawierający tekst, wyodrębnia z niego litery, a następnie dokonuje klasyfikacji. Składa się z pól:

- **ImageMatrix** - przechowuje obraz wejścoiwy,
- **ExtractedLetters**- tablica wyekstraktowanych liter z obrazu; każda litera to macierz 32x28 pikseli typu single
- **LetterPositions** - macierz z pozycjami i rozmiarami wyekstraktowanych liter w obrazie
- **TrainedNet** - wczytany model sieci neuronowej.

Metody w klasie to:

- **setImage** - ustawia obraz do prztworzenia; dostarczony obraz ma być macierzą w formacie RGB lub skali szarości
- **loadNetwork** - wczytuje sieć z pliku .mat
- **performOCR** - wykonuje proces OCR: przetwarza obraz, wyodrębnia litery i klasyfikuje; zwraca przetworzony obraz
- **preprocessImage** - przetrwarza wstępnie obraz; na początku obraz wejściowy - imageMatrix zostaje przekonwertowany do skali szarości za pomocą funkcji rgb2gray(), następnie znormalizowany do zakresu [0,1], zbinaryzowany z odwróceniem i na końcu oczyszczony z szumu
- **visualizeExtractedLetters** - wizualizuje wyekstraktowane litery
- **extractLetters** - metoda ekstrakcji liter; przetwarza obraz linia po linii; przyjmuje jako dane wejściowe birnany obraz wejściowy, a zwraca listę wyekstraktowanych liter; w tej funkcji linie tekstu przetwarzane są do momentu zakończenia się obrazu, następnie wyniki są agregowane; na końcu usuwane są potencjalne puste komórki
- **lines** - funkcja dzieląca obraz na pierwszą linię tekstu i resztę obrazu; służy do przetwarzania obrazu linia po linii
- **clip** - funkcja zmniejszajaca obraz do minimalnego wymaganego prostokąta zawierającego tekst
- **processLine** - przetwarza pojedynczą linię tekstu; na początku spójne częsci są etykietowane; następnie za pomocą funkcji regionprops() obliczane są właściwości statystyczne; w instrukcji if obliczana jest średnia wysokość liter w linii, następnie każda część jest przetwarzana, później następuje scalanie elementów; na końcu litera jest ekstraktowana i normalizowana; wyniki zostają zapisane
- **sortComponentsByPosition** - sortuje części od lewej do prawej na podstawie pozycji X

- **mergeDotsWithLetters** - scala kroki z odpowiednimi literami; przyjmuje odpowiednie warunki identyfikacji kropki
- **extractSingleLetters** - wycina pojedynczą literę z obrazu linii na podstawie Bounding Box'a
- **mergeBoundingBoxes** - scala dwa Bounding boxy
- **resizeAndNormalizeLetter** - skaluje i normalizuje wyciętą literę do standardowych wymiarów 32x28; dla specyficznych liter stosuje odrębne skalowanie

6 Korzystanie z aplikacji

Aby uruchomić program, należy w środowisku MATLAB uruchomić skrypt o nazwie "main.m". Na ekranie powinno wyświetlić się okno aplikacji przedstawione na powyższym zdjęciu. Następnie należy dodać dane do przetworzenia za pomocą przycisku "Załaduj obraz". Rozpoznawanie rozpocznie się po naciśnięciu przycisku "Uruchom OCR". Jeśli dane zostaną poprawnie przetworzone i rozpoznane, wynik wyświetli się w polu tekstowym znajdującym się pod obrazkiem. W celu załadowania modelu lub trenowania sieci należy nacisnąć przysick "Deweloper".

7 Przykładowe dane

Nasza aplikacja umożliwia załadowanie oraz odczytanie logo firm; przykładowe dane zostały zamieszczone w folderze projektu.

Prezentacja rozpoznawania logo firmy - przykład nr 1

Prezentacja rozpoznawania logo firmy - przykład nr 2

8 Wymagania i ograniczenia

Aby uruchomić projekt, należy posiadać w programie MATLAB "Deep learning Toolbox" oraz "Image Processing Toolbox". Dodatkowo, aplikacja obsługuje pliki z rozszerzeniami .png, .jpg oraz .bmp.

9 Podział obowiązków

- **Graficzny interfejs użytkownika** - Jan Kwiatkowski
- **Sieć neuronowa** - Helena Jońca, Aleksander Kopyto
- **Dokumentacja** - Julia Przeździk

10 Możliwe usprawnienia aplikacji

Aplikację można przykładowo ulepszyć o rozszerzenie obsługiwanych formatów obrazu czy rozpoznawanie bardziej złożonych logo firm; na przykład zawierających nie tylko tekst, ale również grafiki.

11 Referencje

- Kaggle
- MATLAB - dokumentacja uifigure

- Opis warstw sieci w MATLAB