Politechnika Wrocławska
Wydział Informatyki i Telekomunikacji

Kierunek: _____ Informatyka techniczna (ITE)
Specjalność: _____ Systemy informatyki w medycynie (IMT)

PRACA DYPLOMOWA Inżynierska

Implementacja systemu optycznego rozpoznawania znaków oraz opracowanie środowiska eksperymentalnego opartego o samodzielnie skonstruowany zbiór danych

Krzysztof Zalewa

Opiekun pracy
Dr inż., Paweł Zyblewski

Słowa kluczowe: 3-6 słów kluczowych

WROCŁAW (2025)

Streszczenie

Dodaj streszczenie pracy w języku polskim. Staraj się uwzględnić wymienione na stronie tytułowej słowa kluczowe. Uwaga przedstawiony rekomendowany szablon dotyczy pracy dyplomowej pisanej w języku angielskim. W przeciwnym wypadku, student powinien samodzielnie zmienić nazwy „Chapter" na „Rodział" itp stosując odpowiednie pakiety systemu LaTeXoraz ustawienia w pliku *latex-settings.tex*.

Abstract

Streszczenie w języku angielskim.

Spis treści

1. Wstęp

1.1. Opis problemu

1.2. Cel pracy

Test

Celem pracy jest porównanie, w oparciu o eksperymenty komputerowe, działania wybranych algorytmów optycznego rozpoznawania znaków. Kluczowym elementem pracy jest pozyskanie i opracowanie autorskiego zbioru danych, składającego się z treści dostępnych za pośrednictwem publicznie dostępnego API serwisu wolnelektury.pl, który pozwoli na rzetelną ewaluację znanych z literatury algorytmów OSR. Opracowany zbiór zawierał będzie dokumenty o zróżnicowanej charakterystyce, uwzględniając m.in. różne kroje i stopnie pisma, a także modyfikacje utrudniające poprawne odczytanie treści.

2. Przegląd literatury

2.1. Narzędzia OCR

Do badań wybrano następujące cztery algorytmy optycznego rozpoznawania znaków. Badania były wykonywane na najnowszych dostępnych wersjach danego algorytmu.

2.1.1. Tesseract

Tesseract OCR to najstarszy z wybranych algorytmów optycznego rozpoznawania znaków. Został on stworzony przez firmę HP w latach 1984 - 1994. Obecnie Tesseract jest jednym z najpopularniejszych algorytmów OCR w kręgach akademickich [2, 6, 9] Do badań użyto wersji 5.3.4 jest to znaczące gdyż wcześniejsze wersje algorytmu(do wersji 3 włącznie) nie wykorzystywały sieci neuronowych. Algorytm ten działa w następujących krokach:

1. Analiza komponentów, gdzie zarys tych komponentów jest przechowywany. Takie podejście mimo że nakłada dodatkowe koszty obliczeniowe pozwala na łatwiejsze rozpoznawanie tekstu w odwróconych kolorach (biały tekst na czarnym tle) [10].

2. Wyszukiwanie lini w komponentach. Celem tego kroku była eliminacja potrzeby korekty przekrzywienia.

3. Podział linii na słowa.

4. Pierwsza iteracja rozpoznawania. Zaczynając na górze strony algorytm próbuje rozpoznać każde kolejne słowo. Jeżeli jest duże prawdopodobieństwo że słowo jest poprawne jest ono wykorzystywane do douczenia klasyfikatora. W ten sposób z każdym kolejnym słowem celność klasyfikatora powinna rosnąć.

5. Druga iteracja rozpoznawania. Po wykonaniu pierwszej iteracji jest duże prawdopodobieństow że klasyfikator uzyskałby lepsze wyniki. Więc po raz drugi algorytm próbuje rozpoznać tekst na stronie i aktualizuje słowa które były minej celnie rozpoznane.

2.1.2. Paddle OCR

Pomimo tego, że algorytm ten jest stosunkowo nowy (pierwsze wersje zostały wypuszczone w 2020 roku [3]). Paddle OCR jest drugim pod względem popularności algorytmem(zaraz po Tesserakcie). Początkowo pierwsze wersje algorytmu skupiały się na balansie między jakością wyniku a czasem jego otrzymania. Wraz z czasem w kolejnych wersjach udoskonalano wydajność algorytmu oraz rozszerzano jego umiejętności (Np. obsługa wielu języków, rozpoznawanie pisma ręcznego). Najnowsze wersje tego algorytmu(W pracy użyto wersji

3.2.0) składają się trzech głównych modułów. **PP-OCR** Rdzeń całego algorytmu służący do rozpoznawania znaków na obrazie. **PP-Structure** Moduł służący do rozpoznawania ustruktryzowanych obrazów (np. Zawierających tabele). **PP-ChatOCR** Moduł służący do ekstrakcji kluczowych informacji z obrazów przy pomocy dużego modelu językowego(z ang. Larg Language Model lub LLM). W tej pracy zastosowany został jedynie moduł PP-OCR. Działa on w następujących krokach:

1. **Preprocesowanie** - W celu uzyskania jak najlepszej jakości obrazu algorytm może usunąć niektóre zniekształcenia oraz problemy z orientacją obrazu.

2. **Wykrycie tekstu** - Algorytm tworzy mapę prawdopodobieństwa, w której każdy piksel ma przydzieloną wartość określającą jakie jest prawdopodobnieństwo, że piksel ten jest częścią obszaru tekstowego. Następnie przy pomocy binaryzacjci różniczkowalnej (ang. Differentiable Binarization) dynamicznie określany jest próg pomiędzy tekstem a tłem. Ostatecznie na podstawie tej mapy tworzone są wielokąty będące zarysem obszaru tekstowego.

3. **Wykrycie orientacji linii** - Wykryty tekst dzielony jest na linie. Następnie algorytm upewnia się, że wykryte linie tekstu są w prawidłowej orientacji

4. **Rozpoznanie tekstu** - Poprzez zastosowanie konwolucyjnej sieci neuronowej (z ang. Convolutional Neural Network lub CNN) wykrywane są charakterystyczne elementy tekstu jak pociągnięcia, krzywe i pętle. Elementy te podwane są do rekurencyjnej sieci neuronowej (z ang. Recurrent Neural Network lub RNN), która dzięki możliwości "zapamiętania" poprzednich elementów jest w stanie odróżnić poszczególne znaki. Na koniec koneksjonistyczna klasyfikacja czasowa (z ang. Connectionist Temporal Classification lub CTC) działa jako mechanizm wyrównywania i zwijania i znajduje najbardziej prawdopodobny napis (Np. "ccccczzzzzaaass" zostaje zmienione na "czas")

2.1.3. DocTR OCR

Algorytm ten jest skupiony na rozpoznawaniu dokumentów takich jak skany faktur,paragonów, formularzy czy listów. Z tąd też nazwa Document Text Recognition czy docTR w skrócie. Główną filozofią tego projektu jest "bezproblemowe optyczne rozpoznawanie znaków dostępne dla każdego" [8]. Algorytm ten stosuje dwuetapowe podejście do rozpoznawania tekstu:

1. **Wkrywanie tekstu** - DocTr pozwala na wykożystanie w tym celu wielu różnych modeli. Jednakże większość z nich,podobnie jak w Tesseract i Paddle OCR, oparta jest na konwolucyjnych sieciach neuronowych. Jednakże w przeciwieństwie do tych dwóch algorytmów docTr używa piramidy cech (z ang. Feature Pyramid Network) co pozwala na odczyt tekstu w wielu różnych rozmiarach (Przydatne na przykład do odróżnienia nagłówka od adnotacji itp.). Następnie algortym dla każdego piksela w obrazie przewiduje czy jest on w obszarze tekstowym i twoży wielokąty wokół wykrytego tekstu.

2. **Rozpoznanie tekstu** - Podobnie jak Paddle OCR docTr najpierw rozpoznaje cechy charakterystyczne tekstu przy pomocy CNN. Jednakże do odróżnienia znaków używana

jest dwukierunkowa rekurencyjna sieć neuronowa. Sieć ta różni się od zwykłej sieci RNN tym że czyta sekwencje znaków od lewej do prawej a następnie od prawej do lewej. W niektórych przypadkach ciąg liter "cl" może być bardzo zbilżony do "d". Dlatego też zabieg ten redukuje możliwość pomylenia znaków.

2.1.4. Easy OCR

EasyOCR jest jednym z nowszych algorytmów użytych w tej pracy (Pierwsza wersja pochodzi z 2019 roku [5]). Rozwojem tego projektu zajmuje się zespół Jaided AI specjalizujący się w wizji komputerowej i uczeniu maszynowym. Mimo stosunkowo krótkiej historii algorytm ten stał się poularny w śród deweloperów o raz w kręgach akademickich. Projekt ten priorytetyzuje prostotę, szybkość oraz wygodę w użytkowaniu. Podobnie jak DocTR OCR algorytm ten działa w dwóch krokach:

1. **Wkrywanie tekstu** - W przeciwieństwie do pozostałych algorytmów EasyOCR w tym kroku korzysta z modelu CRAFT (z ang. Character-Region Awareness For Text detection). Model ten tworzy dwie mapy prawdopodobieństwa. W pierwszej mapie zapisane jest prawdopodobieństwo tego że dany piksel znajduje się na środku znaku. Druga mapa zawiera pradopodobieństwo tego że dany piksel jest na środku przerwy między znakami. Nakładając na siebie te dwie mapy model jest w stanie precyzyjnie wyliczyć zarys każdego znaku a następnie wyrysować wielokąt zawierający w sobie dane słowo.

2. **Rozpoznanie tekstu** - Ten krok jest już znacznie bardziej standardowy. Przebiega podobnie jak analogiczny krok w algorytmie DocTR OCR.

2.2. Zbiory danych

Aby uzasadnić przydatność wykonanego zbioru danych wybrano zbiory IAM oraz old-books-dataset. Dla zbiorów tych zostały przeprowadzone badania [4, 6] w których wykonano testy dla algorytmu Tesseract.

2.2.1. IAM

IAM to zbiór ręcznie zapisanych tekstów w języku angielskim. Wykonany przez Instytut matematyki i informatyki na Uniwersytecie Breńskim. [7] Zbiór zawiera obrazy w rozdzielczości 300dpi zapisane w formacie PNG w 256 odcieniach szarości. Każdy pod katalog zawiera teksty zapisane przez jedną osobę.

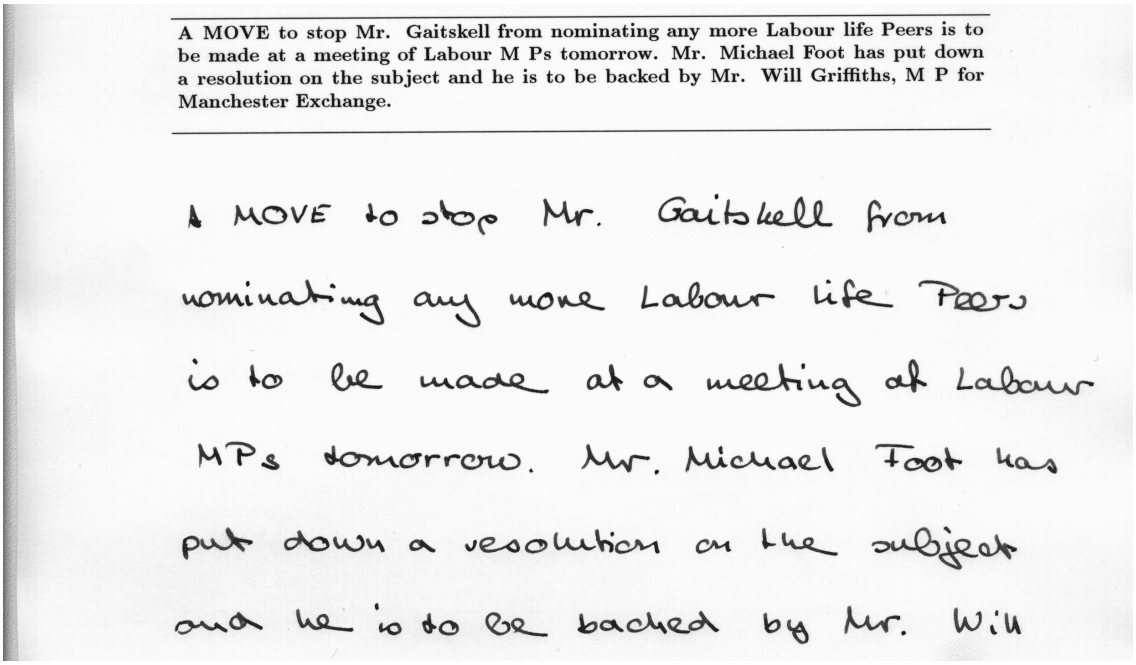

Figure 2.1: Przykładowy obraz ze zbioru danych IAM

2.2.2. oldbooksdataset

Zbiór udostępniony na platformie git hub zawierający skany książek w języku angielski. Książki zapisane są w formacie .tiff w rozdzielczości 300dpi oraz 500dpi [1].

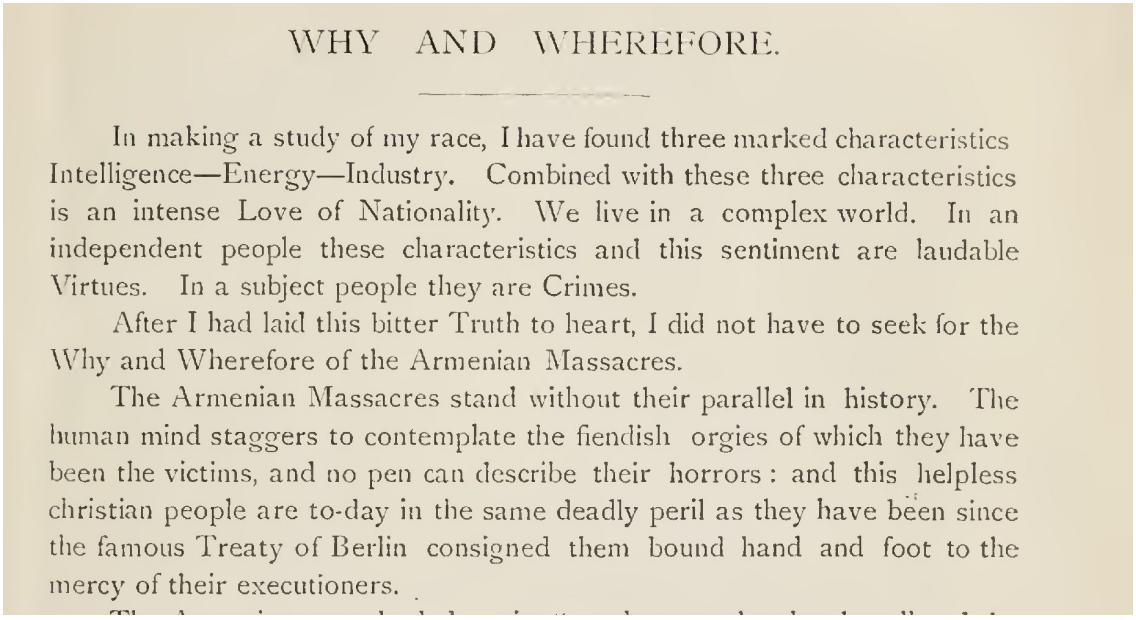

Figure 2.2: Przykładowy obraz ze zbioru danych old-books-dataset

2.3. Metryki

Do oceny wyników zastosowano dwie główne metryki. CER i WER zasadniczą różnicą między nimi jest celność porównań. CER porównuje na poziomie pojednyńczych znaków natomiast, z kolei WER porównuje na poziomie poszczególnych słów. Z tego powodu CER jest przydaniejsze w kontektstach w których nawet pojedyńczy znak może zmienić znaczenie słowa (np. w medycynie). Natomiast WER jest lepsze do porównywania spójności zdań itp.

2.3.1. CER

CER (Character Error Rate z ang. częstotliwość błędnych znaków) to metryka dzięki której możliwa jest ocena różnic między tekstem wytworzonym poprzez model OCR a tekstem rzeczywistym. W tym przypadku CER obliczane jest poprzez zsumowanie operacji(wstawień, usunięć oraz zamian znaków) potrzebnych do uzyskania tekstu rzeczywistego.

$$CER = \frac{S + D + I}{N_c}$$

Gdzie:

- S - Liczba zamian znaków (ang. Substitutions)
- D - Liczba usunięć znaków (ang. Deletions)
- I - Liczba wstawień znaków (ang. Inserts)
- N_c - Liczba znaków w tekście (ang. Number of characters)

Na przykład

Tekst orginalny: Życiem wschód, śmiercia południe;
Tekst wygenerowany przez model: Zyciem wschod, siercia poudniex;

Aby przekształcić tekst wygenerowany do tekstu orginalnego należy wykonać 4 zamiany (Brakujące znaki polskie), 1 wstawienie (Brakujące 'm' w tekście wygenerowanym) oraz 1 usunięcie ('x' nie występuje w tekście orginalnym). Więc $CER = 6/28 = 0.2141 \approx 21.4\%$

2.3.2. WER

WER (Word Error Rate z ang. częstotliwość błędnych słów) podobnie jak CER jest to metryka dzięki której możliwa jest ocena różnic między tekstem wytworzonym poprzez model OCR a tekstem rzeczywistym. Jak sama nazwa wskazuje WER porównuje tekst na poziomie poszczególnych słów.

$$WER = \frac{S + D + I}{N_w}$$

Gdzie:

- S - Liczba zamian słów (ang. Substitutions), czyli słowa które występują w obu tekstach ale te w tekście są różne od tych w tekście orginalnym.
- D - Liczba usunięć słów (ang. Deletions), czyli słowa które występują w tekście orginalnym jednakże nie ma ich w tekście wygenerowanym.
- I - Liczba wstawień słów (ang. Inserts), czyli słowa nadmiarowe których nie ma w tekście orginalnym.
- N_w - Liczba słów w tekście (ang. Number of words)

Na przykład

Tekst orginalny: Życiem wschód, śmiercia południe;
Tekst wygenerowany przez model: Zyciem wschod, siercia poudniex;

Aby przekształcić tekst wygenerowany do tekstu orginalnego należy wykonać 4 zamiany(Słowa zbliżone do orginału ale nie takie same). Więc WER=4/4=1=100%

3. Aspekt inżynierski

Implementacja wybranych, znanych z literatury metod optycznego rozpoznawania znaków (ang. optical character recognition <OSR>). Opracowanie autorskiego zbioru danych – na podstawie treści samodzielnie pozyskanych z serwisu wolnelektury.pl z wykorzystaniem dostępnego publicznie API – na potrzeby ewaluacji zaimplementowanych algorytmów. Całość implementacji zostanie wykonana w języku Python z wykorzystaniem właściwie dobranych bibliotek programistycznych. Dodatkowo, na potrzeby ewaluacji, opracowane oraz zaimplementowane zostanie odpowiednie środowisko eksperymentalne.

3.1. Wybór algorytmów OCR

3.2. Akwizycja danych

4. Aspekt badawczy

4.1. Opis problemu

5. Podsumowanie

Bibliografia

[1] P. Barcha. Old books dataset. https://github.com/PedroBarcha/old-books-dataset, 2024. Accessed: 2024.

[2] A. Chowdhury, A. A. Sami, S. M. P. Mamun, S. Absar, F. Biswas, and M. Kohinoor. Performance analysis of tesseract and easyocr for bangla optical character recognition on the novel bangla crosshair dataset. In *Proceedings of the International Conference on Computer and Communication Systems*, 01 2025.

[3] C. Cui, T. Sun, M. Lin, T. Gao, Y. Zhang, J. Liu, X. Wang, Z. Zhang, C. Zhou, H. Liu, Y. Zhang, W. Lv, K. Huang, Y. Zhang, J. Zhang, J. Zhang, Y. Liu, D. Yu, and Y. Ma. Paddleocr 3.0 technical report. *arXiv preprint*, 2025.

[4] T. Hegghammer. Ocr with tesseract, amazon textract, and google document ai: a benchmarking experiment. *Journal of Computational Social Science*, 5:861–882, 2022.

[5] JaidedAI. Ready-to-use ocr. https://github.com/JaidedAI/EasyOCR, 2024.

[6] Y. Li. Synergizing optical character recognition: A comparative analysis and integration of tesseract, keras, paddle, and azure ocr. *University of Sydney Technical Reports*, 45:45–60, 2023.

[7] U. Marti and H. Bunke. The iam-database: An english sentence database for off-line handwriting recognition. *International Journal on Document Analysis and Recognition*, 5:39–46, 2002.

[8] Mindee. doctr: Document text recognition. https://github.com/mindee/doctr, 2021.

[9] V. S and S. A. Performance comparison of ocr tools. *International Journal of UbiComp*, 6(3):19–30, July 2015.

[10] R. Smith. An overview of the tesseract ocr engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR)*, volume 2, pages 629–633, 2007.

Spis ilustracji

Spis tabel