Projekt z przedmiotu
Analiza i Przetwarzanie Obrazów

Jakub Kawka, Bartosz Gawron, Marek Dzierżawa, Kacper Karabasz,
Andrzej Brzyski, Zofia Stateczna, Grzegorz Gurdak, Kacper Duda

czerwiec 2025

Spis treści

1 Wstęp **2**

2 Dane Treningowe **2**

3 Wykrywanie Tekstu **2**
3.1 Przykładowe elementy obrazu wykorzystane przez algorytm 3
3.2 Problemy detekcji tekstu 3

4 Analiza Tekstu **4**

5 Opracowanie modelu ResNet **5**

6 Ocena Jakości Modelu **6**

7 Instrukcja Obsługi **7**

8 Podsumowanie **8**

9 Materiały Pomocnicze **9**

1 Wstęp

Projekt ma na celu stworzenie programu opierającego się na sztucznej inteligencji, który próbuje oszacować lokalizację na podstawie nagrania z dashcamu zgodnie z dokumentacją [1]. Ustalono, że przybliżenie lokalizacji będzie polegało na podaniu kraju, w którym powstało nagranie. Sama tematyka poruszana w projekcie jest podobna do motywu gry geograficznej GeoGuessr [2], gdzie na podstawie widoku z dashcamu należy zaznaczyć na mapie jak najdokładniejszą lokalizację.

2 Dane Treningowe

W ramach przygotowania danych treningowych do uczenia sieci neuronowej został opracowany dedykowany skrypt w języku Python, który automatycznie pobiera zdjęcia z serwisu Mapillary [3]. Mapillary to platforma należąca do firmy Meta, która udostępnia ogromne ilości zdjęć z poziomu ulicy, wykonanych głównie przez pojazdy poruszające się po drogach całego świata (w tym z kamer typu dashcam). Charakteryzuje się ona rozbudowanym API oraz dokładnymi metadanymi (współrzędne geograficzne, kierunek jazdy, czas wykonania). Dzięki temu Mapillary świetnie sprawdza się w roli źródła danych do zastosowań związanych z analizą przestrzenną i wizualną. Skrypt automatyzuje proces pobierania zdjęć, iterując po granicach wszystkich krajów świata na podstawie danych geoprzestrzennych z zestawu Natural Earth (plik o rozszerzeniu .shp). Dla każdego kraju obliczany jest bounding box czyli prostokątne obramowanie, które zawiera w sobie cały dany kraj. Na jego podstawie wysyłane jest zapytanie HTTP do API Mapillary. Pobieranych jest maksymalnie 100 zdjęć na kraj — z zachowaniem lokalnej równowagi geograficznej. Następnie skrypt filtruje zdjęcia, które faktycznie znajdują się w geometrycznych granicach danego państwa (nie tylko w jego bboxie), wykorzystując do tego bibliotekę shapely. Każde pobrane zdjęcie zapisywane jest lokalnie w katalogu `country_dataset/<nazwa_panstwa>/` , a metadane są zapisywane w pliku CSV o nazwie `country_dataset.csv`. Wiersze w pliku zawierają nazwę państwa, współrzędne geograficzne (szerokość i długość geograficzną), a także ścieżkę do lokalnego pliku graficznego. Taka struktura umożliwia późniejsze bezpośrednie wykorzystanie danych w procesie uczenia sieci neuronowej.

Przykładowy wpis w pliku CSV wygląda następująco:

Kraj	**Szerokość**	**Długość**	**Ścieżka do zdjęcia**
Poland	49.68597002	20.6561458	`country_dataset/Poland/4808510875842208.jpg`

Tabela 1: Przykładowy wpis w pliku CSV

Rysunek 1: Przykładowe zdjęcie z Polski pobrane z Mapillary

W przykładzie detekcji tekstu użyto także filmu znajdującego się na platformie Youtube [4]. Materiał znajduje się także jak przykład w repozytorium git.

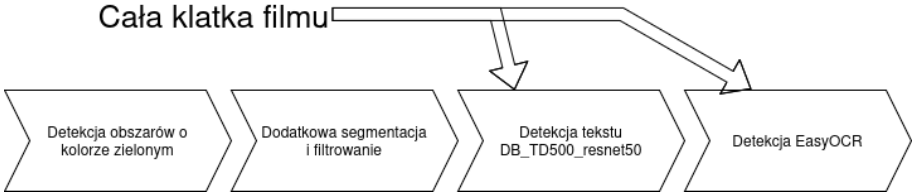
3 Wykrywanie Tekstu

Aby umożliwić analizę tekstu występującego na nagraniach, należy najpierw dokonać ekstrakcji tekstu z poszczególnych klatek filmu.

Moduł wykrywający tekst analizuje klatki filmu w poszukiwaniu znaków drogowych, reklam i innych form tekstu. Ograniczenie liczby analizowanych klatek ma na celu przyspieszenie działania algorytmu, przy zbyt częstym przetwarzaniu klatek rośnie liczba duplikatów znajdowanego tekstu. Analiza jest przeprowadzana globalnie dla całej klatki oraz dla obszarów tekstu które są zielone. Jest to motywowane kolorem znaków drogowych które wskazują nazwy miejscowości. Na rysunku 2 został przedstawiony poglądowy schemat działania algorytmu.

Dodatkowo wykorzystany został model detekcji tekstu oparty o Resnet 50 do wykrycia obszarów, które są następnie osobno przesyłane do modułu OCR w celu dalszej analizy. Należy zwrócić uwagę na to, że ten model jest w stanie tylko zaznaczyć obszar, który prawdopodobnie jest tekstem.

Jako silnik OCR został wykorzystany easyocr. Silnik ten potrafi wykryć obszary zawierające tekst oraz dokonać ekstrakcji tekstu z tych obszarów.



Rysunek 2: Schemat przebiegu detekcji tekstu

3.1 Przykładowe elementy obrazu wykorzystane przez algorytm

Zielone obszary są wykrywane poprzez utworzenie maski z kolorów, które mieszczą się w zakresie RGB [0, 60, 25] do [50, 255, 120]. Kolor ten odpowiada w przybliżeniu zielonemu odcieniowi znaków drogowych. Maska ta następnie jest odszumiana i powiększana za pomocą operacji morfologicznych. Po dokonaniu etykietowania za pomocą funkcji OpenCV2 wyznaczany jest minimalny prostokąt opisujący obszar zielony. Tak utworzony prostokąt jest następnie wykorzystany do wycięcia obszaru i przekazania go do modułu OCR. Przykładowy wycięty obszar jest zamieszczony na obrazie 3

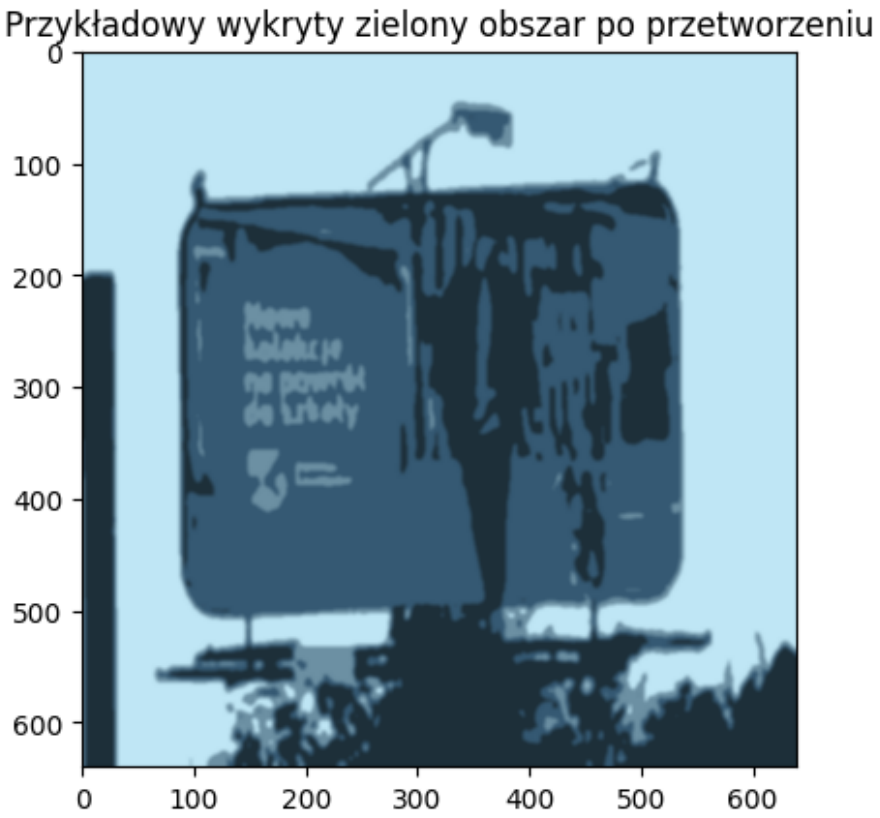
Mniejsze obszary, które nie posiadają wystarczająco dużej rozdzielczości, są segmentowane, aby uzyskać lepszy kontrast pomiędzy możliwie występującym tekstem a tłem. Obszary, które nie są kwadratowe, są powiększane tak, aby zmniejszyć wpływ skalowania do rozmiaru oczekiwanego przez model detekcji tekstu na czytelność czcionki. Wynik takiego przetworzenia przedstawiony jest na rysunku 4.

Rysunek 3: Przykładowy zielony obszar przypisany znakowi drogowemu

Po przetworzeniu obszarów są one wysyłane do modułu EasyOCR. Otrzymywana jest lista pól tekstowych, z których odrzucana jest część o zbyt niskiej pewności (¡= 0,4) oraz długości (¡= 3 znaków). Są to granice wybrane eksperymentalnie, które dawały najlepsze wyniki przy jednoczesnym minimalnym odrzucaniu rzeczywistego tekstu.

3.2 Problemy detekcji tekstu

Zaimplementowana metoda detekcji tekstu nie jest idealna i posiada pewne niedociągnięcia:



Rysunek 4: Przetworzony zielony obszar

- Granica która określa kolor zielony nie zależy od kolorystyki filmu. W zależności od balansu bieli jako kolor zielony może być wykrywana cała klatka. Zjawisko to marnuje czas obliczeniowy i nie wykrywa większej liczby pól tekstowych od analizy całej klatki. Znacznie lepsze byłoby wykrywanie koloru zielonego przez dynamiczny próg, określony np. analizą histogramu kanału zielonego.
- Tekst zawarty na przykład na szyldach sklepów jest widziany przez kamerę samochodową pod pewnym kątem. Użyte modele nie są w stanie wykryć takiego tekstu.
- Moduł wykrywania tekstu potrafi wielokrotnie wykryć to samo pole tekstowe jeżeli znajduje się na wielu klatkach, duplikując dane. Po wykonaniu OCR dokonywana jest deduplikacja, jednak nie jest ona wykonywana za pomocą podobieństwa tekstu a jedynie za pomocą identyczności.
- Czytającego kod programu mogą zaciekawić „na sztywno" ustawione wartości means w algorytmu detekcji tekstu - wynikają one ze sposobu w jaki trenowany był model, detekcja obrazu odbywa się najbardziej wydajnie gdy rozkład barw jest podobny do tego w zbiorze treningowym. Ponieważ nie jest przeprowadzana automatyczna korekta barw, może to powodować problemy w detekcji pozornie łatwo wykrywanego tekstu.

Podsumowując, w roli detekcji tekstu znacznie lepiej sprawdziłby się dedykowany model widzenia komputerowego, jednak trening tak dużego i zaawansowanego modelu wykracza poza zasoby sprzętowe dostępne w tym projekcie. Wymusza to użycie gotowych rozwiązań, które dają satysfakcjonujące wyniki po wykorzystaniu dodatkowej pomocy klasycznych metod analizy obrazu.

4 Analiza Tekstu

Aby możliwe było oszacowanie lokalizacji na podstawie tekstu odczytanego z nagrania, konieczne jest przeprowadzenie analizy semantycznej treści uzyskanej z modułu OCR. Analizowany tekst zawiera zwykle fragmenty nazw ulic, autostrad, punktów orientacyjnych, reklam lub nazw firm. Informacje te są

przekazywane do kolejnego etapu przetwarzania, którego zadaniem jest próba określenia przybliżonej lokalizacji pojazdu.

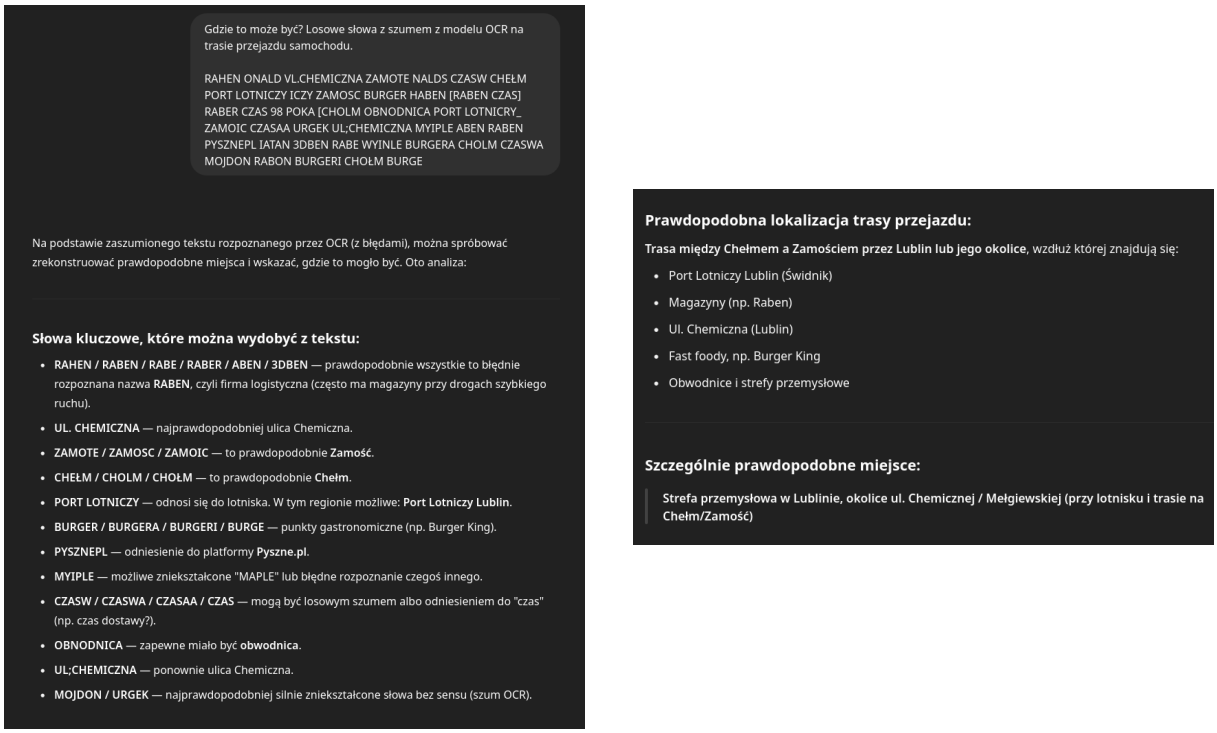
Moduł analizy tekstu wykorzystuje model językowy. Model otrzymuje tekst w formie ciągu znaków i próbuje na jego podstawie zidentyfikować najbardziej prawdopodobne miejsce, którego dotyczy. Przede wszystkim stara się wskazać nazwę miasta i kraju, a w przypadku niedostatecznej ilości informacji, przynajmniej kontynent. Dzięki temu system jest w stanie zwrócić przybliżoną lokalizację nawet w przypadku niepełnych lub niejednoznacznych danych. W przypadkach trudnych, gdzie model nie jest w stanie określić konkretnej lokalizacji, stosowany jest fallback na poziom kontynentu.

W obecnej wersji systemu wykorzystano model Mistral-7B-Instruct uruchamiany lokalnie. Model ten cechuje się dużą elastycznością i jest w stanie formułować odpowiedzi w spójnym, przewidywalnym formacie, co ułatwia późniejsze przetwarzanie wyników. Odpowiedź modelu jest następnie analizowana z użyciem wyrażeń regularnych w celu wyodrębnienia nazw miejscowości, krajów oraz kontynentów.

Warto zaznaczyć, że skuteczność działania modułu w dużej mierze zależy od jakości danych wejściowych. W przypadkach, gdy tekst jest nieczytelny, niepełny lub zawiera jedynie ogólne pojęcia (np. „Exit 5” lub „STOP”), model może mieć trudności z dokładnym określeniem lokalizacji. Niemniej jednak, w wielu przypadkach udaje się uzyskać satysfakcjonującą dokładność nawet przy minimalnej ilości danych.

W praktyce zauważono, że moduł text2loc napotyka ograniczenia podczas przetwarzania dużych ilości danych wejściowych, szczególnie gdy wykryty tekst jest zbyt długi lub pochodzi z wielu klatek filmu analizowanego bez odpowiedniego interwału czasowego. Wydłużony tekst może powodować problemy po stronie modelu językowego, prowadząc do spadku skuteczności lokalizacji lub nawet zakończenia programu przez niewystarczające zasoby. Eksperymentalnie wykazano, że zwiększenie odstępu pomiędzy przetwarzanymi klatkami (frame interval) oraz ograniczenie długości przekazywanego tekstu pozwala na uzyskanie lepszych i bardziej stabilnych wyników.

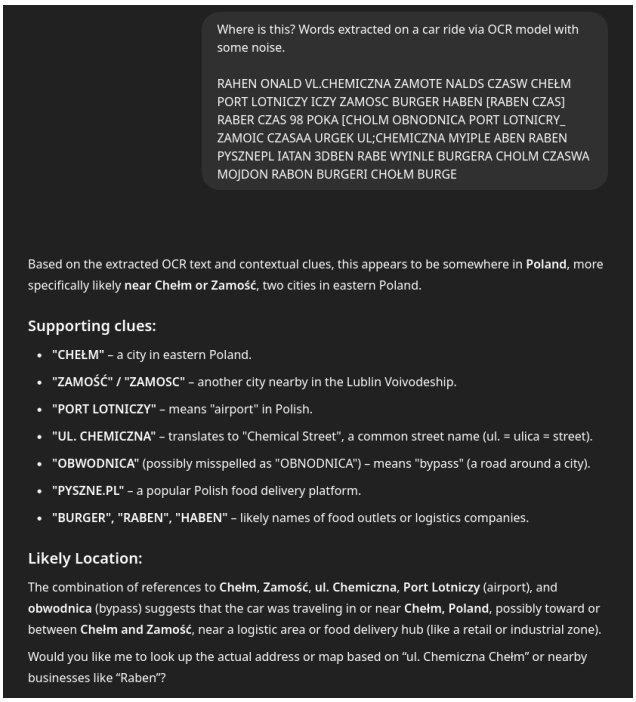
Dodatkowo, bardzo dobre rezultaty uzyskano przy zastosowaniu modelu ChatGPT zarówno w wersji polskiej (rys. 5), jak i angielskiej (rys. 6).



Rysunek 5: Predykcja lokalizacji przez ChatGPT - po polsku.

5 Opracowanie modelu ResNet

Opracowano model klasyfikacji obrazów, którego celem jest przewidywanie kraju na podstawie pojedynczego zdjęcia przedstawiającego daną lokalizację. Wstępnie dane zostały przefiltrowane tak, aby uwzględniały wyłącznie te kraje, które posiadają co najmniej dwie próbki. Nazwy krajów zakodowano numerycznie za pomocą LabelEncoder, co umożliwiło późniejsze użycie ich jako etykiet w klasyfikacji.



Rysunek 6: Predykcja lokalizacji przez ChatGPT - po angielsku.

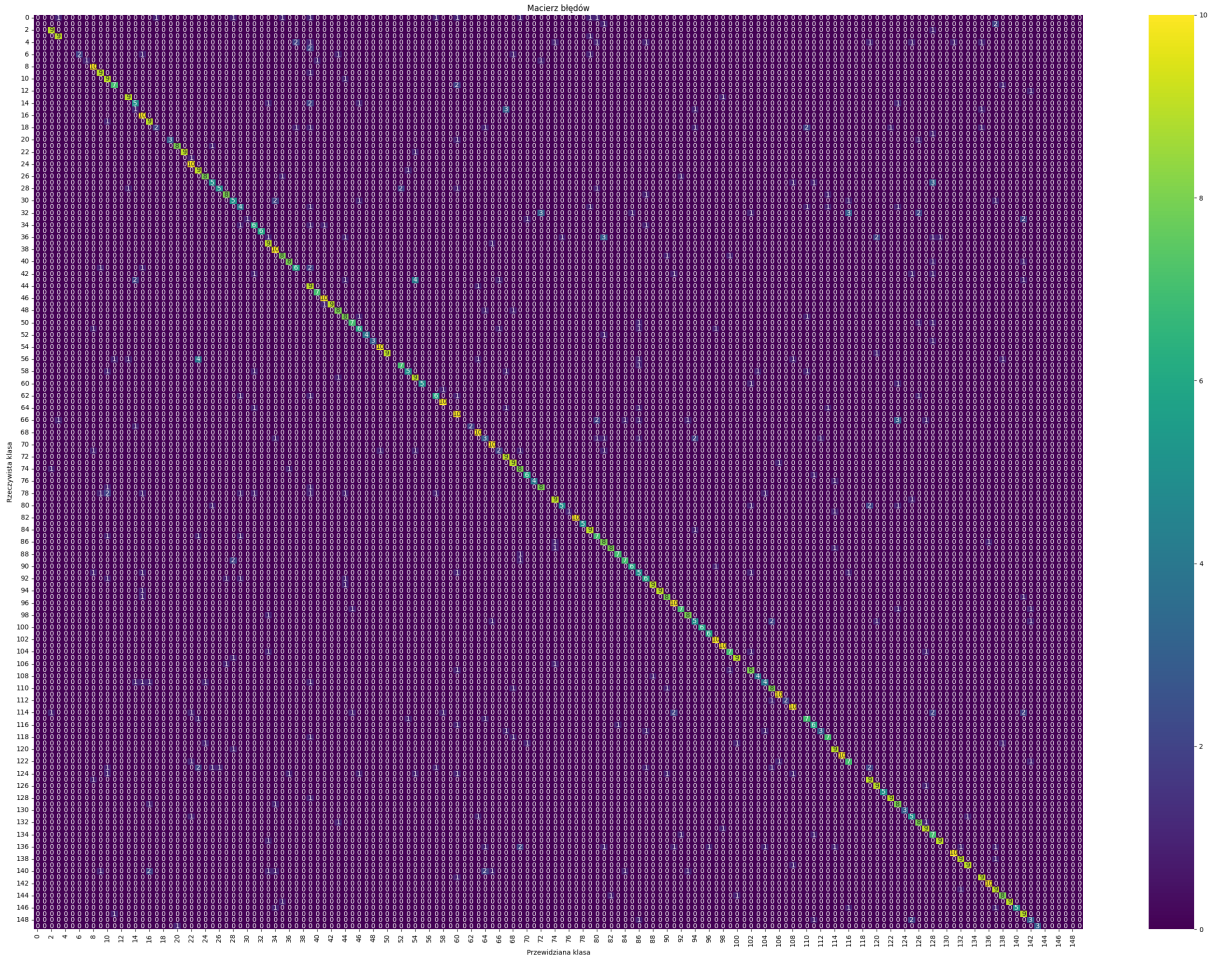
Dane zostały podzielone na trzy zbiory: treningowy (75%), walidacyjny (15%) i testowy (10%), przy czym podział został wykonany z zachowaniem proporcji klas. W celu zwiększenia różnorodności danych treningowych zastosowano augmentację obrazów, obejmującą m.in. losowe poziome odbicia, zmiany kontrastu i jasności oraz losowe przycięcia. Do klasyfikacji wykorzystano sieć konwolucyjną ResNet18, w której warstwę wyjściową dostosowano do liczby klas. Model był trenowany z wykorzystaniem funkcji straty CrossEntropyLoss, uwzględniającej wagi klas wyliczone na podstawie liczby przykładów w każdej klasie. Model został przetrenowany na 30 epokach, ale wybrany został ten, który miał najlepszą dokładność dla danych walidacyjnych. Osiągnął on dokładność 76.32% na danych treningowych, 54.63% podczas walidacji oraz 56.1% dla danych testowych.

6 Ocena Jakości Modelu

W ramach projektu opracowano model głębokiego uczenia ResNet, którego zadaniem była klasyfikacja zdjęć lub filmów przedstawiających lokalizacje geograficzne do odpowiedniego kraju. Etykiety krajów zostały zakodowane numerycznie przy użyciu LabelEncoder, umożliwiając bezpośrednie wykorzystanie ich jako klas w klasyfikacji wieloklasowej.

Wydajność modelu oceniono na podstawie kilku podstawowych metryk. Uzyskana dokładność (accuracy) wyniosła jedynie 5,49%, co oznacza, że tylko taki procent próbek został poprawnie sklasyfikowany. Równie niskie wartości osiągnięto dla średniego recallu (4,91%) oraz średniego F1-score (4,74%) w ujęciu makro, co wskazuje na trudności modelu zarówno w poprawnym rozpoznawaniu klas, jak i unikaniu błędnych klasyfikacji. Obliczona została również metryka top-5 accuracy (8.77%), która pozwoliła ocenić, czy prawidłowa etykieta znalazła się wśród pięciu najwyżej ocenionych przez model klas, jednak wynik wskazuje, że i ta miara nie przyniosła zadowalających wyników.

Dalsza analiza została przeprowadzona na podstawie macierzy błędów, która jednoznacznie wskazuje, że model myli się niemal we wszystkich klasach. Większość z nich nie została poprawnie sklasyfikowana ani razu, a nieliczne trafienia występują sporadycznie i zwykle ograniczają się do jednej lub dwóch prawidłowych klasyfikacji. Obserwuje się dużą asymetrię w liczbie pomyłek pomiędzy poszczególnymi klasami. Niektóre kraje są wybierane przez model częściej, nawet jeśli są nieprawidłowe, co może świadczyć o istnieniu klas dominujących w zbiorze treningowym. Brakuje również wyraźnych wzorców pomyłek w klasycznym rozumieniu, jednak da się zauważyć pewien schemat w wynikach modelu. Predykcje są często przesunięte względem poprawnych etykiet o jedną lub dwie klasy, co skutkuje złudzeniem przekątnej w macierzy błędów — choć nie jest to rzeczywista zgodność z klasami, może świadczyć o tym, że mo-



Rysunek 7: Macierz pomyłek klasyfikatora

del uchwycił pewne globalne cechy danych, lecz nie nauczył się ich precyzyjnego przyporządkowania do konkretnych klas.

Tak niska jakość klasyfikacji może mieć kilka źródeł. Po pierwsze, duża liczba klas przy niewielkiej liczbie przykładów dla wielu z nich sprawia, że model ma trudności z nauczeniem się odpowiednich wzorców. Po drugie, dane mogą być silnie niezbalansowane, co skutkuje tym, że model faworyzuje bardziej reprezentowane klasy. Dodatkowo, same obrazy mogą nie zawierać wystarczająco charakterystycznych cech pozwalających na precyzyjne rozróżnianie krajów. Problemy mogą również potęgować brak skutecznych technik augmentacji danych oraz zbyt szybkie dopasowanie modelu do dominujących klas (overfitting).

W celu poprawy wyników zaleca się kilka działań. Przede wszystkim warto ograniczyć liczbę klas do tych krajów, które mają wystarczającą liczbę próbek. Równocześnie należy rozważyć zastosowanie wag klasowych podczas treningu, aby zniwelować wpływ niezbalansowania danych. Istotnym krokiem może być także zwiększenie zbioru danych, szczególnie dla słabo reprezentowanych klas, lub wprowadzenie rozszerzonej augmentacji obrazów. Rekomenduje się także wykorzystanie technik transfer learningu i ponownego dostrajania (fine-tuningu) modelu ResNet z zamrożonymi warstwami początkowymi i dłuższym czasem treningu.

Podsumowując, opracowany model ResNet w obecnej postaci nie osiąga zadowalających wyników w zadaniu klasyfikacji krajów na podstawie zdjęć. Przyczyną są głównie ograniczenia wynikające z charakteru danych i ich reprezentacji. Kolejne iteracje projektu powinny koncentrować się na poprawie jakości i struktury zbioru danych oraz optymalizacji procesu uczenia, aby umożliwić modelowi lepsze odwzorowanie istotnych cech dla poszczególnych klas.

7 Instrukcja Obsługi

Sklonować repozytorium z Githuba

```
git clone git@github.com:AiPO-proj/AiPO.git
```

Utworzyć wirtualne środowisko python:

```
python3 -m venv venv
source venv/bin/activate
```

Zainstalować potrzebne biblioteki:

```
pip install -r requirements.txt
```

Wgrać plik do katalogu *input* nagranie i nazwać go *video.mp4* - w formacie MP4.
Uruchomić program:

```
python main.py
```

W przypadku posiadania niewystarczających zasobów program może nie wykonać się pomyślnie.

8 Podsumowanie

W ramach projektu opracowano program, którego celem jest szacowanie kraju pochodzenia z dashcam. Rozwiązanie polega na analizie zarówno tekstu, jak i wykrytego na nim tekstu, żeby najdokładniej oszacować kraj. Stworzono skrypt w Python do automatycznego pobierania danych z Mapillary. Ważnym elementem okazało się rozpoznawanie tekstu i OCR, przy wykorzystaniu EasyOCR. Pomimo wdrożonych udoskonaleń, takich jak filtracja kolorów czy segmentacja obrazu, system napotyka ograniczenia związane z różnorodnością jakości i kolorystyki analizowanych materiałów oraz obecnością duplikatów i błędów odczytu. Rozpoznane napisy są weryfikowane przez działający lokalnie model językowy Mistral-7B-Instruct, który ma na podstawie tego tekstu dobrać najbardziej prawdopodobną lokalizację. Model stanowi także filtr przed błędnie odczytanymi napisami oraz jest w stanie odzyskać poprawne nazwy z zapisów z błędami. Pomimo wdrożenia metod augmentacji danych i zbalansowania klas, uzyskano bardzo niską jakość predykcji, co wynika głównie z niewystarczającej liczby przykładów w wielu klasach, niezbalansowanych danych oraz małej liczby wyróżniających się cech charakterystycznych dla poszczególnych krajów na poziomie zdjęć ulicznych.

Projekt stanowi praktyczną analizę możliwości lokalizacji geograficznej na podstawie nagrań drogowych z wykorzystaniem zarówno przetwarzania obrazu, jak i informacji tekstowych. Choć wyniki klasyfikacji obrazów są niesatysfakcjonujące, użycie detekcji oraz analizy tekstu pozwala w licznych przypadkach na uzyskanie użytecznych przybliżeń lokalizacji.

Naszym największym problemem okazało się posiadanie niewystarczających zasobów obliczeniowych, co pokazuje, jak dobre rezultaty można by było uzyskać przy użyciu takich modeli językowych jak ChatGPT.

Wkład Autorów

- Jakub Kawka - Detekcja i ekstrakcja tekstu, detekcja znaków drogowych
- Bartosz Gawron - Przygotowanie danych, analiza obrazu bez uwzględnienia danych tekstowych, łączenie modeli w całość
- Zofia Stateczna - Ocena jakości modeli
- Kacper Duda - Poszukiwanie alternatywnych metod wykrywania obrazu i dokumentacja
- Kacper Karabasz - Opracowanie modelu ResNet
- Marek Dzierżawa - Opracowanie modelu ResNet
- Andrzej Brzyski - Analiza obrazu bez uwzględnienia danych tekstowych
- Grzegorz Gurdak - Analiza tekstu pod względem wyciągnięcie lokalizacji

9 Materiały Pomocnicze

Do opracowania algorytmu w większości zostały użyte informacje z części laboratoryjnej przedmiotu. Pomocne natomiast okazały się być przykłady online oraz biblioteki:

- Artykuł na blogu OpenCV o wykrywaniu teskstu na obrazie [5].
- Bliblioteka EasyOCR [6]. Prostsza w użyciu (jak sama nazwa wskazuje) od biblioteki Tesseract.

Literatura

[1] G. Gołaszewski, "Wytyczne dotyczące projektu z przedmiotu analiza i przetwarzanie obrazów," materiały dydaktyczne AGH, 2024, dostępne online: https://home.agh.edu.pl/~ggolasz/AiPO˙projekt.pdf.

[2] "Geoguessr – let's explore the world!" https://www.geoguessr.com/, dostęp 19 czerwca 2025.

[3] Mapillary AB / Meta Platforms, Inc., "Mapillary – platforma crowdsourcingowej, geotagowanej fotografii ulicznej," https://www.mapillary.com/, Mapillary AB (Malmö, Szwecja), 2025, dostęp 19 czerwca 2025.

[4] OSKEwan, "Rondo makro lublin skręt w lewo z ul. krańcowej," YouTube, – 2017, dostęp 17 czerwca 2025. [Online]. Available: https://www.youtube.com/watch?v=wI2NKMa9˙vk

[5] moukthika, "Text detection and removal using opencv," https://opencv.org/blog/text-detection-and-removal-using-opencv/, OpenCV Team, Mar. 2025, dostęp 17 czerwca 2025.

[6] R. Kittinaradorn, "Easyocr: Ready-to-use ocr with 80+ supported languages," https://github.com/JaidedAI/EasyOCR, Jaided AI, 2025, dostęp 17 czerwca 2025.