

Raport: 1

Projekt: KLASYFIKACJA CYFR

Wydział: Inżynierii Mechanicznej i Robotyki

Kierunek: Inżynieria Akustyczna

Rok: 3

Przedmiot: Technologia mowy

Grupa: 3

Imię i nazwisko:

1. Oliwia Chmura
2. Mateusz Juszcak
3. Henryk Kosała

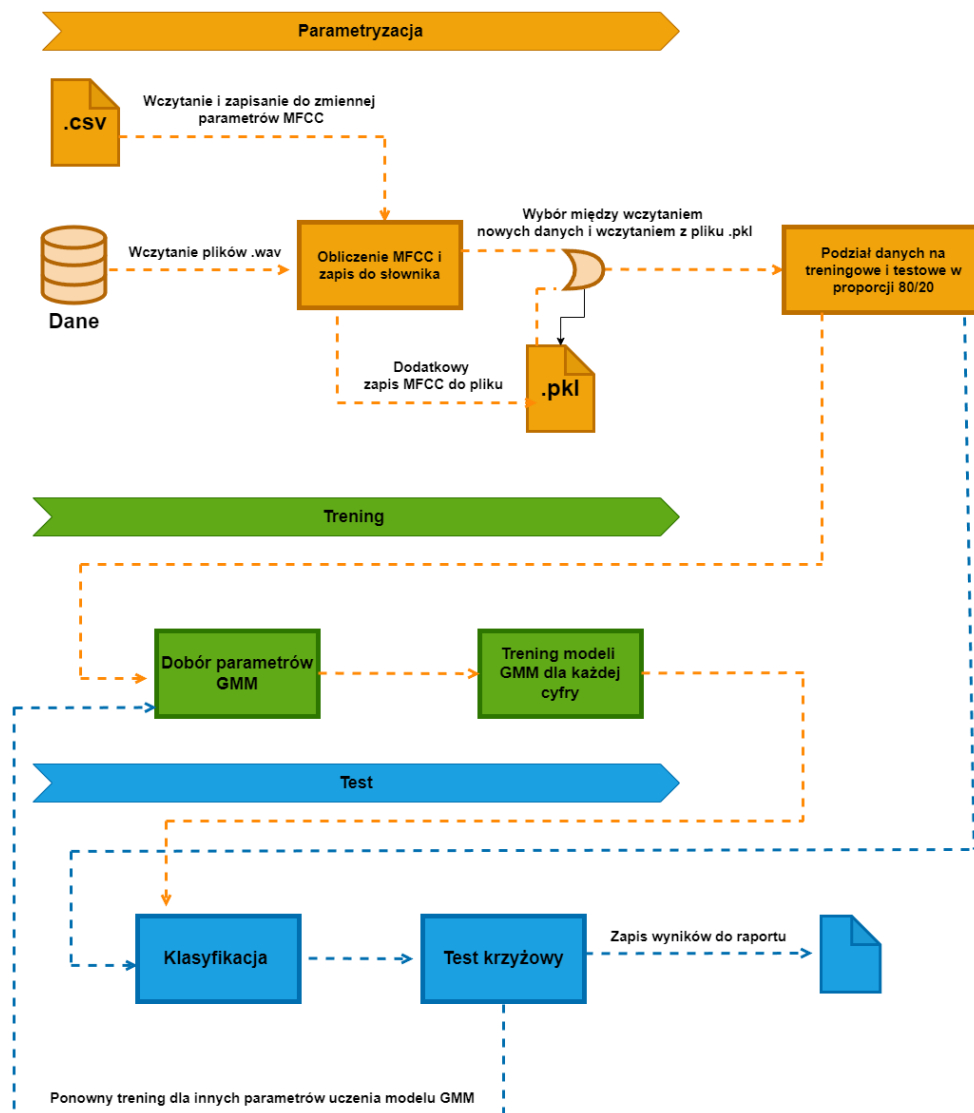
1. Zakres i cel projektu

Celem projektu jest stworzenie systemu automatycznej klasyfikacji cyfr wypowiedzianych przez różnych mówców i oszacowanie w teście krzyżowym $N \times N$ (80% vs 20%) jego skuteczności wyrażonej miarą Accuracy (%) wraz z wartością odchylenia standardowego wyniku z wyznaczeniem macierzy błędów (confusion matrix).

2. Architektura systemu i przepływu danych

Danymi wejściowymi do systemu są nagrania audio w formacie .wav, przygotowane przez 22 mężczyzn. Każde nagranie zawiera wymówienie cyfry z zakresu od 0 do 9. Pliki są uporządkowane według mówców.

System jest zaprojektowany modułowo, co pozwala na przejrzysty przepływ danych oraz możliwość łatwej rozbudowy. Najpierw system wyszukuje i wczytuje wszystkie pliki .csv, .wav. Każdy plik .wav przekazywany jest do modułu ekstrakcji cech, gdzie wyznaczane są współczynniki MFCC. Parametry tego procesu pobierane są z pliku mfcc_params.csv, a w przypadku błędów z pliku default_mfcc_params.csv. Po wyznaczeniu cech dane są zapisywane w zorganizowanej strukturze słownikowej. Następnie dane przekazywane są do modułu trenowania modeli. W projekcie zastosowano jeden model GMM dla każdej z cyfr 0–9. Cechy MFCC pochodzące z nagrań tej samej cyfry, zebrane od wszystkich mówców, są łączone w jeden wspólny zbiór treningowy. Kolejno dzielimy otrzymane wartości na zbiór testowy i treningowy, aby przejść do treningu modelu GMM. Wyniki poddajemy klasyfikacji, której wynik zwraca nam wartość etykiety cyfry o najwyższym prawdopodobieństwie, co pozwala na wyznaczenie metryk jakości Accuracy (%) oraz macierzy błędów (confusion matrix).



Rysunek 2.1. Koncepcja systemu.

2.1. Wczytywanie parametrów MFCC

2.1.1. Funkcja `load_mfcc_params()`

Funkcja wczytuje dane z pliku `mfcc_params.csv`. W przypadku braku pliku korzysta z domyślnych wartości z pliku `default_mfcc_params.csv`. Następnie konwertuje pobrane wartości, a następnie wyświetla tabelę zawierającą odczytane parametry.

2.2. Ekstrakcja cech MFCC

2.2.1. Funkcja `get_mfcc()`

Funkcja oblicza i zwraca współczynniki MFCC wyznaczone z wykorzystaniem biblioteki `librosa` na podstawie wcześniej pobranych danych, a następnie wyznacza także współczynniki delta oraz delta-delta.

2.2.2. Funkcja `load_train_files_and_determine_mfcc()`

Funkcja odpowiada za wczytanie danych oraz walidację ich poprawności. Dane wczytywane są za pomocą biblioteki `os` i zapisywane w utworzonym słowniku. Następnie dla każdego pliku wyznaczane są współczynniki MFCC poprzez wywołanie funkcji `get_mfcc()` z przekazanymi parametrami przez `mfcc_params`.

2.2.3. Funkcja `load_mfcc_data()`

Funkcja pozwala na wczytanie wcześniej zapisanych danych z pliku `pickle`. Dzięki temu można szybko rozpocząć proces przygotowania danych i trening modeli bez ponownego przetwarzania nagrań audio.

2.3. Przygotowanie danych do treningu

2.3.1. Funkcja `prepare_training_data()`

Funkcja przygotowuje dane treningowe do GMM łącząc cechy MFCC pochodzące od wszystkich mówców dla każdej cyfry w osobne macierze.

2.3.2. Funkcja `split_train_test_by_speaker()`

Funkcja dzieli mówców na zbiór treningowy i testowy.

2.4. Trenowanie klasyfikatora

2.4.1. Funkcja `train_gmms()`

Funkcja trenuje osobny model GMM dla każdej cyfry. Modele są trenowane na macierzach cech MFCC przygotowanych w poprzednim etapie. Każdy model uczy się charakterystycznych cech danej cyfry w oparciu o nagrania wszystkich mówców.

2.5. Klasyfikacja

2.5.1. Funkcja `classifier()`

Funkcja porównuje log-prawdopodobieństwa dla każdego modelu i zwraca przewidywaną cyfrę oraz wynik.

2.6. Testy

2.6.1. Funkcja test()

Funkcja przeprowadza przykładowy test wsobny, używając danych ze zbioru. Funkcja wykorzystuje klasyfikator zdefiniowany w `classifier()`, a wynik testu zwraca dokładność klasyfikacji dla poszczególnych cyfr. Testy pozwalają ocenić skuteczność systemu oraz wskazują, które cyfry są najtrudniejsze do rozpoznania.

3. Status pracy

3.1. Status wykonanej pracy na dzień 19.11.2025

Zrealizowane zadania:

1. Stworzenie repozytorium oraz planu pracy pierwszego etapu projektu.
2. Implementacja funkcjonalności parametryzującej zbiór audio zgodnie z zaplanowaną architekturą i strukturami danych.
3. Implementacja funkcjonalności umożliwiającej trening klasyfikatora (modeli klas/modelu) zgodnie z zaplanowaną architekturą i strukturami danych.
4. Implementacja funkcjonalności klasyfikatora oraz przykładowego testu wsobnego na kilku plikach różnych klas ze zbioru treningowego.

3.2. Zadania do wykonania

Zadania niezrealizowane:

1. Plan pracy drugiego i trzeciego etapu projektu.
2. Poprawa czytelności zwracanych wartości w konsoli.
3. Implementacja pętli testu krzyżowego - pętli treningu/testowania
4. Implementacja treningu klasyfikatora
5. Dogranie zbioru ewaluacyjnego
6. Wykonanie testów i optymalizacja parametrów klasyfikatora,
7. Test na danych ewaluacyjnych.