Raport

Projekt: KLASYFIKACJA CYFR

Wydział: Inżynierii Mechanicznej i Robotyki

Kierunek: Inżynieria Akustyczna

Rok: 3

Przedmiot: Technologia mowy

Grupa: 3

Imię i nazwisko:

Oliwia Chmura

Mateusz Juszczak

Henryk Kosała

1. Zakres i cel projektu

Celem projektu jest stworzenie systemu automatycznej klasyfikacji cyfr wypowiadanych przez różnych mówców i oszacowanie w teście krzyżowym N x (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. Każde nagranie zawiera wymówienie cyfry z zakresu od 0 do 9. System jest zaprojektowany modułowo, co pozwala na przejrzysty przepływ danych oraz możliwość łatwej rozbudowy. Dzięki zastosowaniu interfejsu użytkownik sam wybiera oraz przechodzi do kolejnych etapów. Cała architektura systemu tworzy spójny przepływ danych, który przebiega od surowych nagrań przez ich przetworzone reprezentacje numeryczne, dalej poprzez proces trenowania i optymalizacji modeli, aż po finalną klasyfikację.

2.1. Przygotowanie danych

Najpierw system wyszukuje i wczytuje wszystkie pliki .csv, .wav. Wczytywanie danych zostało zaimplementowane z wykorzystaniem Google Api z uwagi na przetwarzanie dużej ilości danych, a także konieczności zapewnienia wydajnego i niezawodnego dostępu do zasobów zewnętrznych. 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 danych defaultowych. Po wyznaczeniu cech dane są zapisywane w zorganizowanej strukturze słownikowej i następnie przechodzą przez moduł walidacji, którego celem jest sprawdzenie poprawności struktury przygotowanych danych wejściowych.

2.2. Eksperymenty i optymalizacja

W kolejnym etapie dane dzielone są na zbiory treningowe i testowe, a następnie następuje optymalizacja parametrów MFCC i GMM oraz testy krzyżowe w celu oceny stabilności i skuteczności modeli. Na podstawie tych operacji wybierane są optymalne parametry do treningu finalnego modelu.

2.3. Trening końcowego modelu

W trzecim etapie, po zakończonej optymalizacji, odbywa się trening końcowego modelu z wykorzystaniem najlepszych parametrów MFCC i GMM. W projekcie zastosowano oddzielny 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. Następnie wartości te dzielone są na dane treningowe i testowe, które służą do trenowania modeli GMM.

2.4. Ewaluacja systemu

Po wytrenowaniu modeli system dokonuje klasyfikacji nowych próbek, przy czym wynik klasyfikacji wskazuje cyfrę o najwyższym prawdopodobieństwie. Na tej podstawie obliczane są metryki jakości klasyfikacji, m.in (Accuracy %), co pozwala ocenić skuteczność i niezawodność systemu. Kolejno dokonuje ewaluacji na finalnym modelu zwracając ostateczne wyniki.

3. Status pracy

3.1. Status wykonanej pracy na dzień 19.11.2025

Zrealizowane zadania:

1. Stworzono repozytorium oraz zaplanowanie planu pracy w pierwszym etapie projektu.
2. Utworzono funkcję load_mfcc_params(), która zwraca wczytane z pliku dane.
3. Utworzono funkcję w etapie ekstrakcji cech MFCC:

- get_mfcc(), która oblicza i zwraca współczynniki MFCC
- load_train_files_and_determine_mfcc(), która wyznacza współczynniki MFCC poprzez wywołanie funkcji get_mfcc() z przekazanymi parametrami przez mfcc_params.
- load_mfcc_data(), która pozwala na wczytanie wcześniej zapisanych danych z pliku pickle.

4. Utworzono funkcję w etapie Przygotowania danych do treningu:
   - prepare_training_data(), która przygotowuje dane treningowe do GMM łącząc cechy MFCC pochodzące od wszystkich mówców dla każdej cyfry w osobne macierze
   - split_train_test_by_speaker(), która dzieli mówców na zbiór treningowy i testowy.
5. Utworzono funkcję classifier(), która porównuje log-prawdopodobieństwa dla każdego modelu i zwraca przewidywaną cyfrę oraz wynik.
6. Utworzono funkcję test(), która przeprowadza przykładowy test wsobny i wykorzystuje classifier()

3.2. Status wykonanej pracy na dzień 26.11.2025.

Zrealizowane zadania:

1. Plan pracy drugiego etapu.
2. Poprawiono czytelność kodu i zwracanych wartości w konsoli
3. Zaktualizowano funkcje:
   - get_mfcc()
   - load_mfcc_params()
   - load_ load_train_files_and_determine_mfcc()
   - prepare_training_data()
   - train_gmms()
   - classifier()
4. Dodano nową funkcję do etapu ekstrakcji cech MFCC:
   - load_gmm_params(), która odpowiada za wczytanie parametrów GMM
5. Usunięto funkcję load_mfcc_data()
6. Dodano nową funkcję do etapu przygotowania danych:
   - save_processed_dataset(), która zapisuje przetworzony dataset do pliku

- validate_data_quality(), która sprawdza dane i je zwraca w terminalu
- save_gmm_models(), która zapisuje do pliku pickle
- load_gmm_models(), która wczytuje gmm z pliku pickle
- save_best_parameters(), która zapisuje parametry do pliku csv
-load_best_parameters(), która wywołuje funkcje load_mfcc_params() i load_gmm_params()

7. Zaktualizowano funkcję test() i zmieniono jest nazwę na calculate_accuracy()
8. Dodano nową funkcję do etapu kroswalidacji - cross_validation(), która wykorzystuje KFold, a następnie trenuje modele i zwraca otrzymane wyniki
9. Dodano funkcje do etapu analizy:
   - evaluate_system, która zwraca otrzymane dane z ewaluacji
   - print_confusion_matrix(), która macierz pomyłek
10. Dodano etap optymalizacji – funkcja optimize_parameters_full().
11. Dodano funkcje etapowe, które wywołują zaimplementowane wcześniej funkcje dla danych etapów:

- ETAP: 1 – prepare_data_stage()
- ETAP: 2 – quick_prototype(), optimize_parameters_stage(), cross_validation_stage()
- ETAP: 3 – train_final_model()
- ETAP: 4 – evaluate_system_stage()

12. Przetworzono main.
13. Dogranie zbioru ewaluacyjnego

3.3. Status wykonanej pracy na dzień 7.12.2025.

Zrealizowane zadania:

1. Zaplanowano pracy na ostatni etap projektu.
2. Zaimplementowano pobieranie danych z zewnętrznego dysku za pomocą Google API.
3. Poprawa krytycznego błędu - początkowa implementacja klasyfikowała cyfrę na podstawie pojedynczej ramki MFCC (fragmentu dźwięku), co było błędem, ponieważ ignorowało to kontekst czasowy całego nagrania i drastycznie obniżało dokładność.
4. Usunięcie zbędnych funkcji load_mfcc_params() oraz load_gmm_params(), zastąpiono je wczytywaniem danych z pliku config.py

5. Rozdzielenie całego programu na moduły, które odpowiadają za poszczególne etapy przetwarzania danych wejściowych:
   - audio_processing
   - config
   - evaluation
   - g_loader
   - gmm_manager
   - main
   - optimization
   - utils
   - workflow
6. Stworzono raport końcowy z analizą otrzymanych wyników.

4. Wyniki i wnioski

4.1. Wyniki na dzień 26.11.2025

Otrzymane wyniki przy zastosowaniu funkcji optimize_parameters_full(), w której za pomocą pętli tworzymy zbiór do pliku csv z najlepszymi parametrami. (Rysunek 2.1.). Optymalna konfiguracja uzyskała dokładność 80%, przy parametrach podanych poniżej.

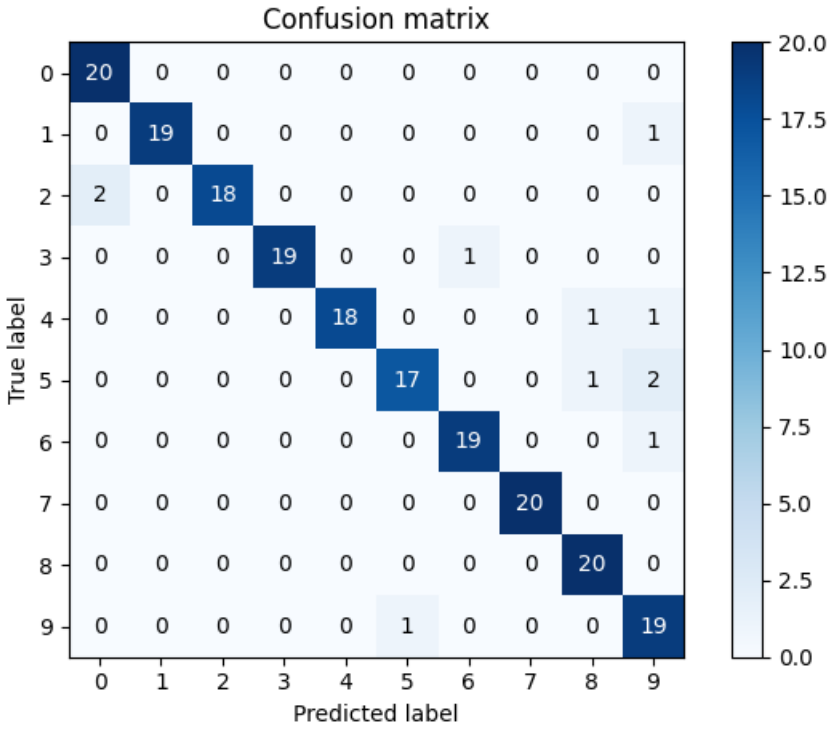
```
=== NAJLEPSZE PARAMETRY ===
Skuteczność: 80.00%
Parametry MFCC: {'n_mfcc': 13, 'n_fft': 1024, 'win_length': 512, 'hop_length': 200, 'n_mels': 32, 'count_delta': True, 'count_delta_delta':
 True}
Parametry GMM: {'n_components': 12, 'covariance_type': 'tied', 'max_iter': 200, 'random_state': 42}
Najlepsze parametry zapisano do plików:
  - best_mfcc_params.csv
  - best_gmm_params.csv

=== TOP 10 KOMBINACJI ===
+--------+---------+---------+-----------+-----------+---------+---------+---------+-----------+-----------+------------+
|   Rank |   n_mfcc |   n_fft |   win_len |   hop_len |   n_mels | delta   | delta2   |   GMM_comp | GMM_cov   |   Accuracy% |
+========+=========+=========+===========+===========+=========+=========+=========+===========+===========+============+
|      1 |       13 |    1024 |       512 |       200 |       32 | ✓       | ✓        |         12 | tied      |       80   |
+--------+---------+---------+-----------+-----------+---------+---------+---------+-----------+-----------+------------+
|      2 |       13 |    1024 |       512 |       160 |       32 | ✓       | X        |         12 | diag      |       76.67 |
+--------+---------+---------+-----------+-----------+---------+---------+---------+-----------+-----------+------------+
|      3 |       20 |    1024 |       512 |       160 |       32 | ✓       | X        |         12 | diag      |       76.67 |
+--------+---------+---------+-----------+-----------+---------+---------+---------+-----------+-----------+------------+
|      4 |       13 |     512 |       400 |       160 |       32 | ✓       | ✓        |         12 | diag      |       73.33 |
+--------+---------+---------+-----------+-----------+---------+---------+---------+-----------+-----------+------------+
|      5 |       13 |     512 |       512 |       200 |       32 | ✓       | ✓        |          8 | tied      |       73.33 |
+--------+---------+---------+-----------+-----------+---------+---------+---------+-----------+-----------+------------+
|      6 |       13 |    1024 |       512 |       200 |       32 | ✓       | ✓        |          4 | diag      |       73.33 |
+--------+---------+---------+-----------+-----------+---------+---------+---------+-----------+-----------+------------+
|      7 |       16 |     512 |       400 |       160 |       32 | ✓       | X        |         12 | tied      |       73.33 |
+--------+---------+---------+-----------+-----------+---------+---------+---------+-----------+-----------+------------+
|      8 |       16 |    1024 |       400 |       200 |       26 | ✓       | X        |          4 | tied      |       73.33 |
+--------+---------+---------+-----------+-----------+---------+---------+---------+-----------+-----------+------------+
|      9 |       20 |     512 |       400 |       200 |       32 | ✓       | ✓        |          8 | diag      |       73.33 |
+--------+---------+---------+-----------+-----------+---------+---------+---------+-----------+-----------+------------+
|     10 |       20 |    1024 |       512 |       200 |       26 | ✓       | ✓        |         12 | diag      |       73.33 |
+--------+---------+---------+-----------+-----------+---------+---------+---------+-----------+-----------+------------+
```

Rysunek 4.1. Otrzymane wyniki

4.2. Wyniki na dzień 7.12.2025

Po dokonaniu zmian poprawiono skuteczność na zbiorze testowym z 80% na 100%. Testując model na danych ewaluacyjnych, przy użyciu wytrenowanego modelu, otrzymano skuteczność wynoszącą 94,5%, która wynika z zastosowanej implementacji. W celu dokładniejszej analizy wyników klasyfikacji przedstawiono macierz pomyłek (Rys. 4.2). Na przekątnej macierzy znajdują się liczby poprawnie sklasyfikowanych przykładów dla poszczególnych klas. W większości przypadków wartości te wynoszą 18–20, co świadczy o tym, że model zazwyczaj prawidłowo rozpoznaje dane klasy. Pojawiające się błędy dotyczą wyłącznie pojedynczych przypadków.



Rysunek 4.2. Macierz błędów dla zbioru ewaluacyjnego

4.3. Analiza hiperparametrów

W trakcie procesu trenowania dostrajano hiperparamtery w celu znalezienia możliwie najwydajniejszej konfiguracji. W tym celu zastosowano funkcję _generate_param_combinations(), która wygenerowała wszystkie parametry użyte w późniejszej optymalizacji w funkcji optimize_parameters_full(). Zastosowano przetwarzanie

danych w sposób równoległy, dzięki czemu przyspieszono proces optymalizacyjny - funkcja _worker_optimize_single().

Brano pod uwagę następujące parametry MFCC i ich wartości:

- Liczba współczynników MFCC - n_mfcc = [13, 16, 20]
- Okno FFT - n_fft = [512, 1024]
- Długość okna - win_len = [400, 512]
- Kroki analizy - hop_lenght = [160, 200]
- Liczba filtrów melowych - n_mels = [26, 32]
- Możliwości delty – delta = [(False, False), (True, False), (True, True)]

Parametry GMM:

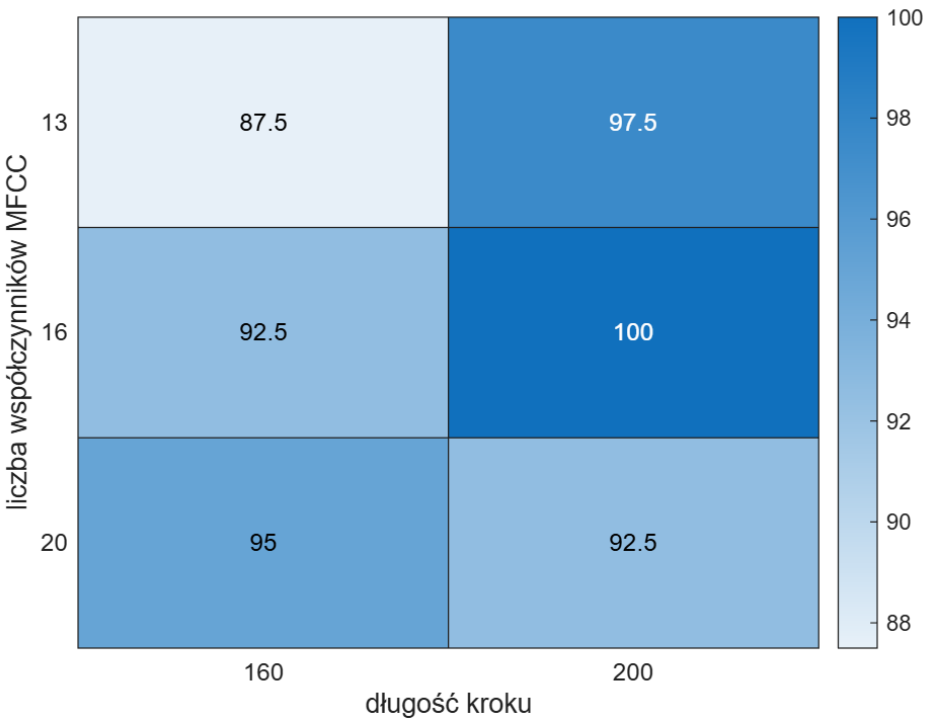
- Liczba współczynników – gmm_comp = [4, 8, 12]
- Typ macierzy – ['diag', 'tied']

Łącznie otrzymano 864 kombinacji.

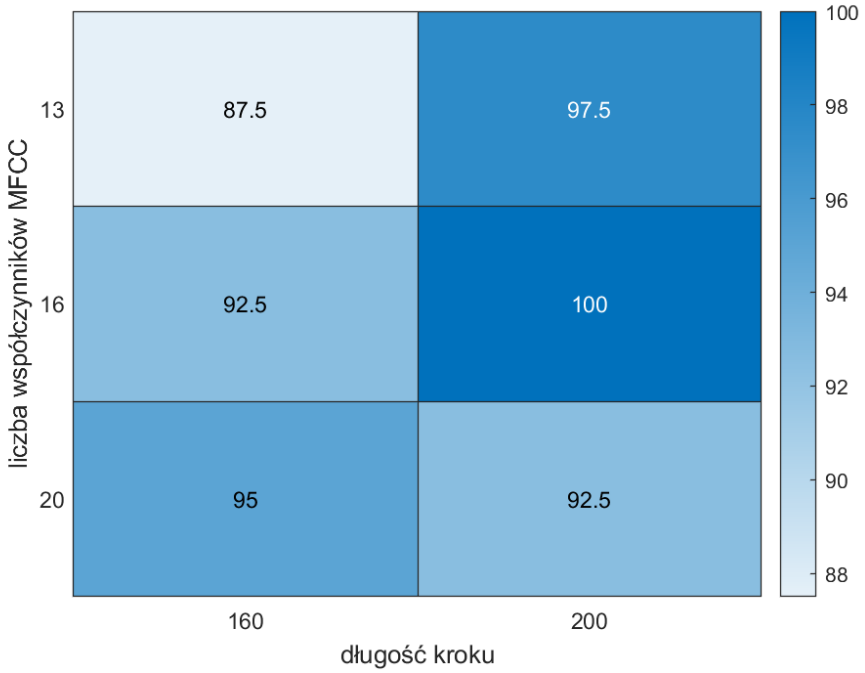
Najlepsze wartości accuracy otrzymano dla danych podanych poniżej (Tabela 1).

Tabela 1. Otrzymane najlepsze wartości accuracy przy zadanych parametrach

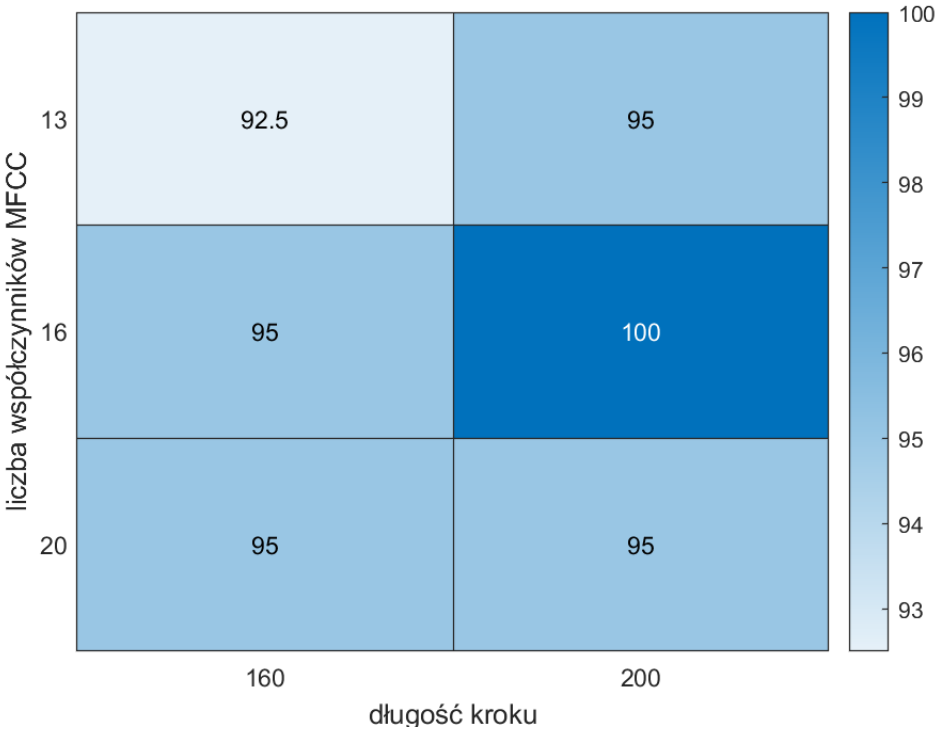
rank	accuracy	n_mfcc	n_fft	win_len	hop_len	n_mels	delta	delta2	gmm_comp	gmm_cov
1	100	13	512	512	200	26	prawda	prawda	4	diag
2	100	13	1024	512	200	26	prawda	prawda	4	diag
3	100	16	512	400	200	26	prawda	prawda	4	diag
4	100	16	512	400	200	32	prawda	prawda	4	diag
5	100	16	512	512	200	26	prawda	prawda	4	diag
6	100	16	1024	400	200	26	prawda	prawda	4	diag
7	100	16	1024	400	200	32	prawda	prawda	4	diag
8	100	16	1024	512	200	26	prawda	prawda	4	diag
9	97,5	13	512	400	200	26	prawda	prawda	4	diag
10	97,5	13	512	512	200	32	prawda	prawda	4	diag
11	97,5	13	1024	400	200	26	prawda	prawda	4	diag
12	97,5	13	1024	512	200	32	prawda	prawda	4	diag
13	97,5	16	512	400	200	26	prawda	fałsz	4	diag
14	97,5	16	512	400	200	32	prawda	fałsz	4	diag
15	97,5	16	512	512	160	32	prawda	fałsz	4	diag
16	97,5	16	512	512	200	26	prawda	fałsz	4	diag
17	97,5	16	512	512	200	32	prawda	fałsz	4	diag
18	97,5	16	512	512	200	32	prawda	prawda	4	diag
19	97,5	16	1024	400	160	32	prawda	fałsz	4	diag
20	97,5	16	1024	400	200	26	prawda	fałsz	4	diag



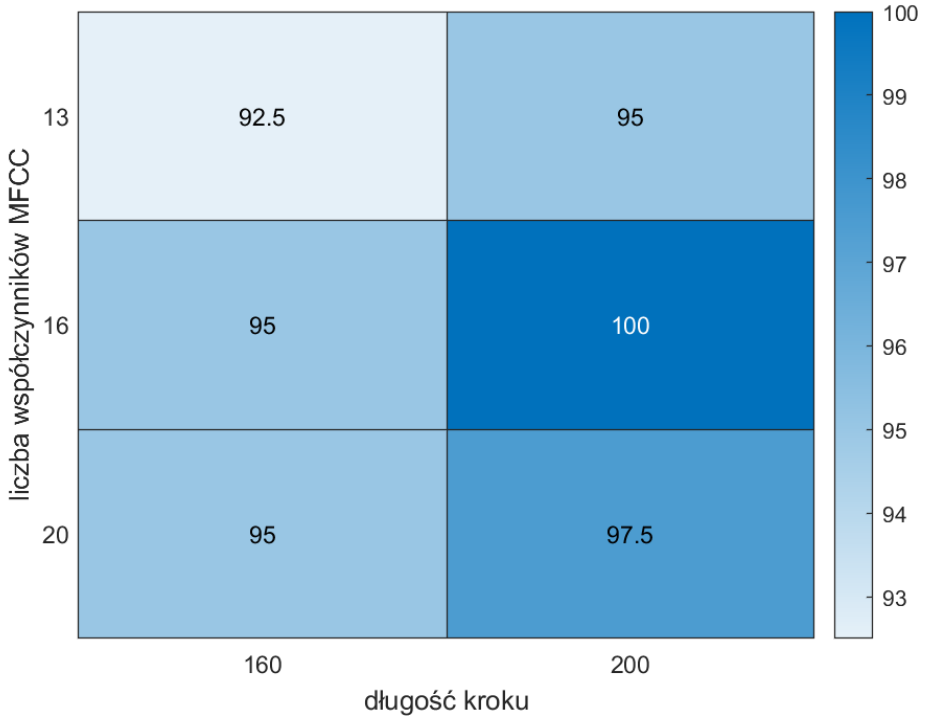
Rysunek 4.3. Mapa ciepła porównująca skuteczność przy liczbie współczynników MFCC z długością kroku analizy, przy pozostałych parametrach n_fft = 512, win_length = 400, n_mels = 26, gmm_comp = 4, macierz diagonalna



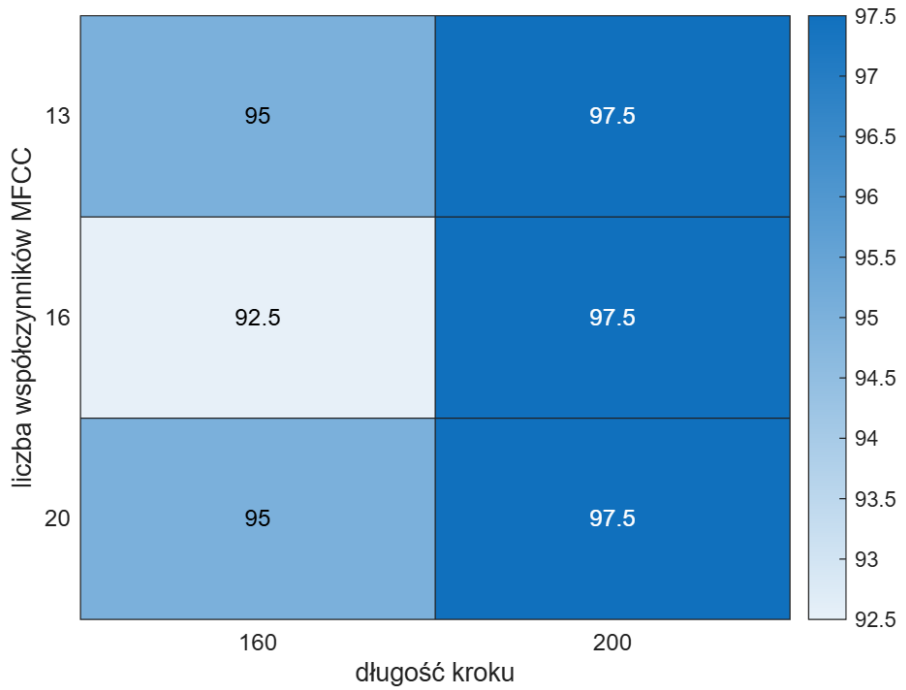
Rysunek 4.4. Mapa ciepła porównująca skuteczność przy liczbie współczynników MFCC z długością kroku analizy+, przy pozostałych parametrach n_fft = 1024, win_length = 400, n_mels = 26, gmm_comp = 4, macierz diagonalna



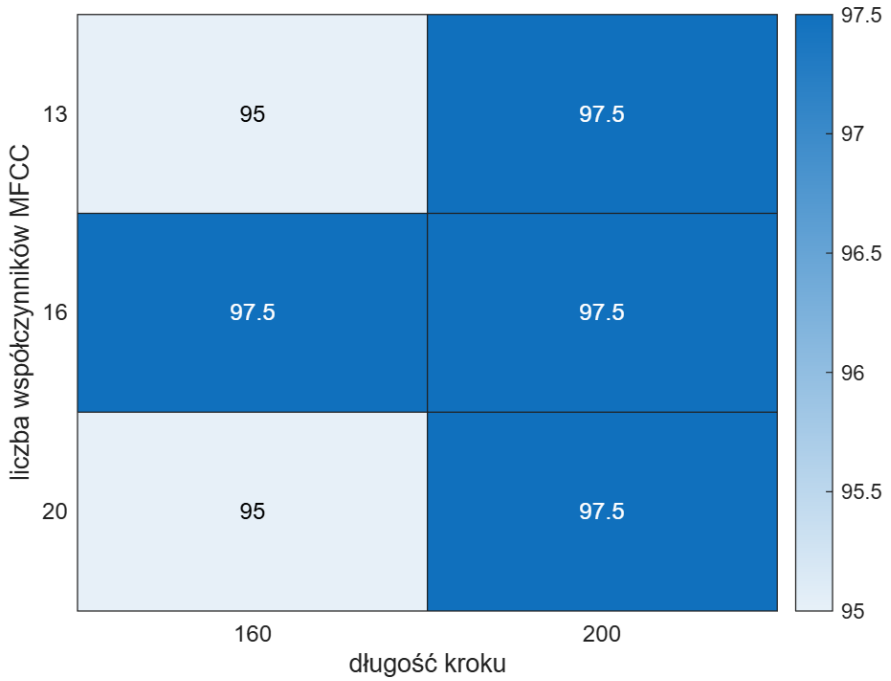
Rysunek 4.5. Mapa ciepła porównująca skuteczność przy liczbie współczynników MFCC z długością kroku, przy pozostałych parametrach n_fft = 512, win_length = 400, n_mels = 32, gmm_comp = 4, macierz diagonalna



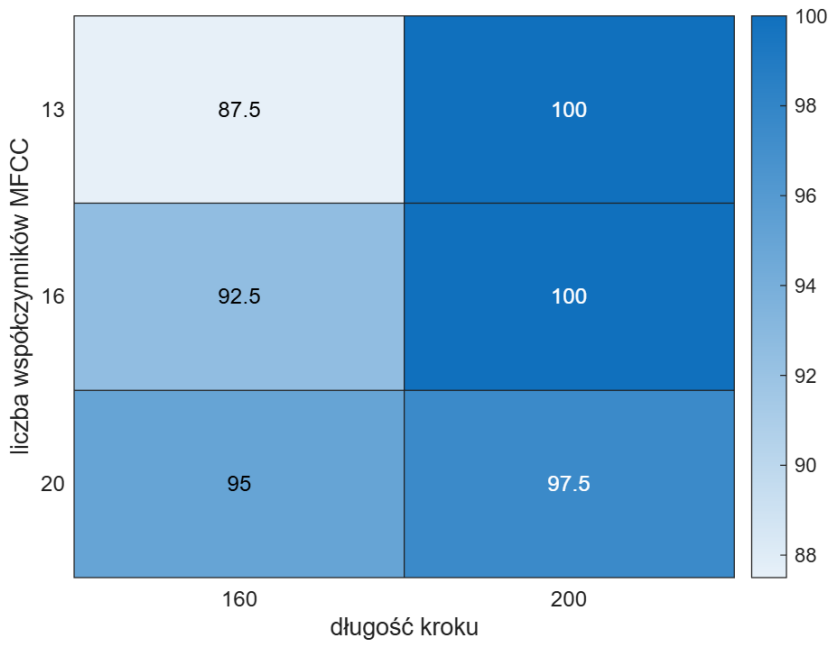
Rysunek 4.6. Mapa ciepła porównująca skuteczność przy liczbie współczynników MFCC z długością kroku, przy pozostałych parametrach n_fft = 1024, win_length = 400, n_mels = 32, gmm_comp = 4, macierz diagonalna



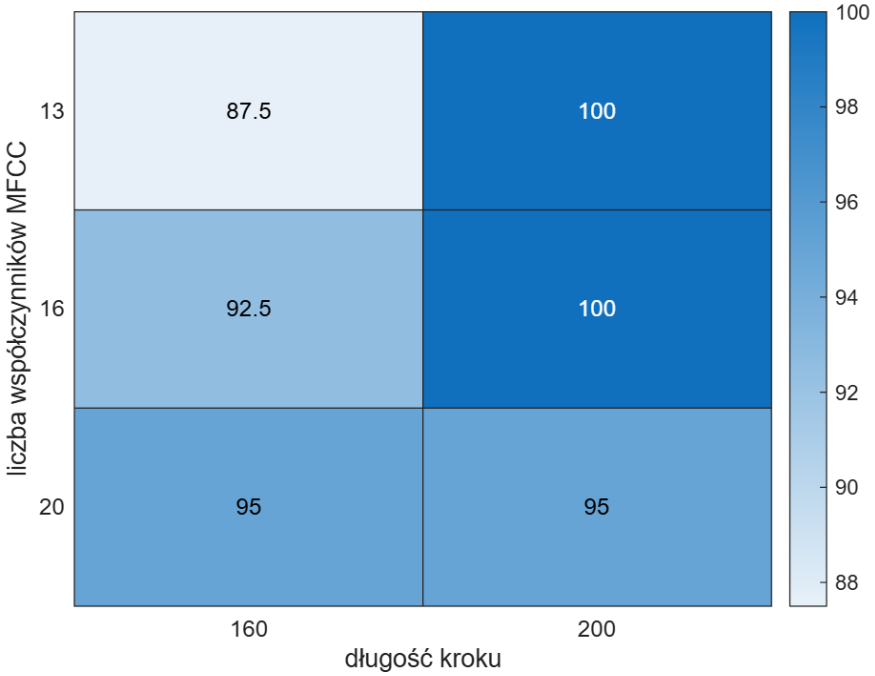
Rysunek 4.7. Mapa ciepła porównująca skuteczność przy liczbie współczynników MFCC z długością kroku, przy pozostałych parametrach n_fft = 512, win_length = 512, n_mels = 32, gmm_comp = 4, macierz diagonalna



Rysunek 4.8. Mapa ciepła porównująca skuteczność przy liczbie współczynników MFCC z długością kroku, przy pozostałych parametrach n_fft = 1024, win_length = 512, n_mels = 32, gmm_comp = 4, macierz diagonalna



Rysunek 4.9. Mapa ciepła porównująca skuteczność przy liczbie współczynników MFCC z długością kroku, przy pozostałych parametrach n_fft = 1024, win_length = 512, n_mels = 26, gmm_comp = 4, macierz diagonalna



Rysunek 4.10. Mapa ciepła porównująca skuteczność przy liczbie współczynników MFCC z długością kroku, przy pozostałych parametrach n_fft = 512, win_length = 512, n_mels = 26, gmm_comp = 4, macierz diagonalna

Analiza uzyskanych wyników pokazuje, że najwyższą skuteczność klasyfikacji, równą 100%, osiągnięto dla modeli GMM wykorzystujących diagonalną macierz kowariancji oraz cztery komponenty. Parametry te występują we wszystkich konfiguracjach o najwyższej skuteczności, co wskazuje, że mają one kluczowy wpływ na poprawne rozpoznawanie mówców.

Najniższe wartości skuteczności, równe 72,5%, odnotowano w przypadkach, w których nie zastosowano delt i delt-2, a jednocześnie użyto macierzy diagonalnej. Pokazuje to, że informacje dynamiczne (delta i delta-delta) są istotnym elementem reprezentacji cech MFCC i wyraźnie poprawiają skuteczność klasyfikacji. Szczególnie ważna okazuje się delta, która w każdej konfiguracji podnosi skuteczność co najmniej do poziomu 82,5%.

Istotnym czynnikiem okazał się również parametr hop_length. Wartość 200 konsekwentnie pojawiała się w modelach o najwyższej skuteczności, natomiast krótszy krok analizy równy 160 obniżał skuteczność do poziomu około 97,5%. Oznacza to, że dłuższy krok czasowy sprzyja stabilniejszej i bardziej skutecznej klasyfikacji w analizowanym zbiorze danych. Można to również zauważyć na Rys. 4.3. – Rys. 4.10., gdzie w każdej konfiguracji parametrów przy kroku 160 osiągana skuteczność była niższa niż w przypadku dłuższego kroku. Dodatkowo, zastosowanie 16 współczynników MFCC wraz z hop_len = 200 w analizowanych konfiguracjach konsekwentnie prowadziło do uzyskania najwyższej skuteczności. Oznacza to, że zwiększona liczba współczynników lepiej odwzorowywała charakterystykę widmową mówców, co przekładało się na najlepsze wyniki klasyfikacji. Pozostałe parametry MFCC takie jak liczba filtrów melowych, długość okna oraz długość transformaty FFT nie wykazywały stałej, jednoznacznej tendencji i ich wpływ zależał jedynie od pozostałych elementów konfiguracji.