Audio Analyser - aplikacja do analizy dźwięku

Natalia Choszczyk

Marzec 2025

Spis treści

1 Wprowadzenie

Audio Analyser to aplikacja umożliwiająca zaawansowaną analizę plików audio. Oferuje funkcje takie jak: załadowanie pliku w formacie .wav, odtwarzanie dźwięku, wizualizacja przebiegu czasowego audio na wykresie, detekcja ciszy oraz dźwięcznych głosek. Aplikacja generuje również wykresy parametrów dźwięku (np. volume, ZCR) oraz umożliwia ich eksport do plików CSV lub TXT. Wykresy są interaktywne, co umożliwia wygodne przeglądanie oraz analizowanie danych.

Aplikacja jest dostępna publicznie pod linkiem: `https://audioanalyser.streamlit.app/`

2 Specyfikacja techniczna

Aplikacja została napisana w języku Python z wykorzystaniem biblioteki Streamlit do stworzenia interfejsu graficznego użytkownika oraz do stworzenia aplikacji internetowej. Program jest zaprojektowany w sposób modularny, co ułatwia dodawanie nowych funkcji w przyszłości. Kod implementujący poszczególne części aplikacji znajduje się w oddzielnych plikach.
Aplikacja obsługuje pliki audio w formacie .wav oraz umożliwia eksport parametrów do plików CSV lub TXT.

2.1 Wykorzystane biblioteki

- **Streamlit** - do intefejsu graficznego użytkownika (GUI) oraz do stworzenia aplikacji internetowej.
- **NumPy** - do operacji matematycznych i obliczeniowych.
- **Librosa** - do załadowania pliku audio.
- **Pandas** - do tworzenia ramek danych, które są wykorzystywane do eksportu parametrów lub wyświetlania ich w tabeli.
- **Plotly** - do generowania interaktywnych wykresów takich przebieg czasowy audio, czy parametry dźwięku.

3 Interfejs użytkownika

Aplikacja jest dostępna w formie aplikacji internetowej. Po uruchomieniu aplikacji należy załadować plik audio w formacie .wav. Po załadowaniu pliku, użytkownik może odtworzyć dźwięk za pomocą wbudowanego widgetu. Pod wykresem przebiegu czasowego audio można wybrać zaznaczenie fragmentów ciszy lub głosek dźwięcznych. Po lewej stronie aplikacji znajduje się panel sterowania (sidebar), w którym użytkownik może regulować parametry analizy, takie jak:

- rozmiar ramki (frame size),
- krok ramki (frame step),
- progi głośności dla ciszy i głosek dźwięcznych,
- progi parametru ZCR dla ciszy i głosek dźwięcznych.

Poniżej wykresu przebiegu czasowego audio znajduje się wykres przedstawiający parametry dźwięku na poziomie poszczególnych ramek. Dostępnych jest pięć typów wykresów:

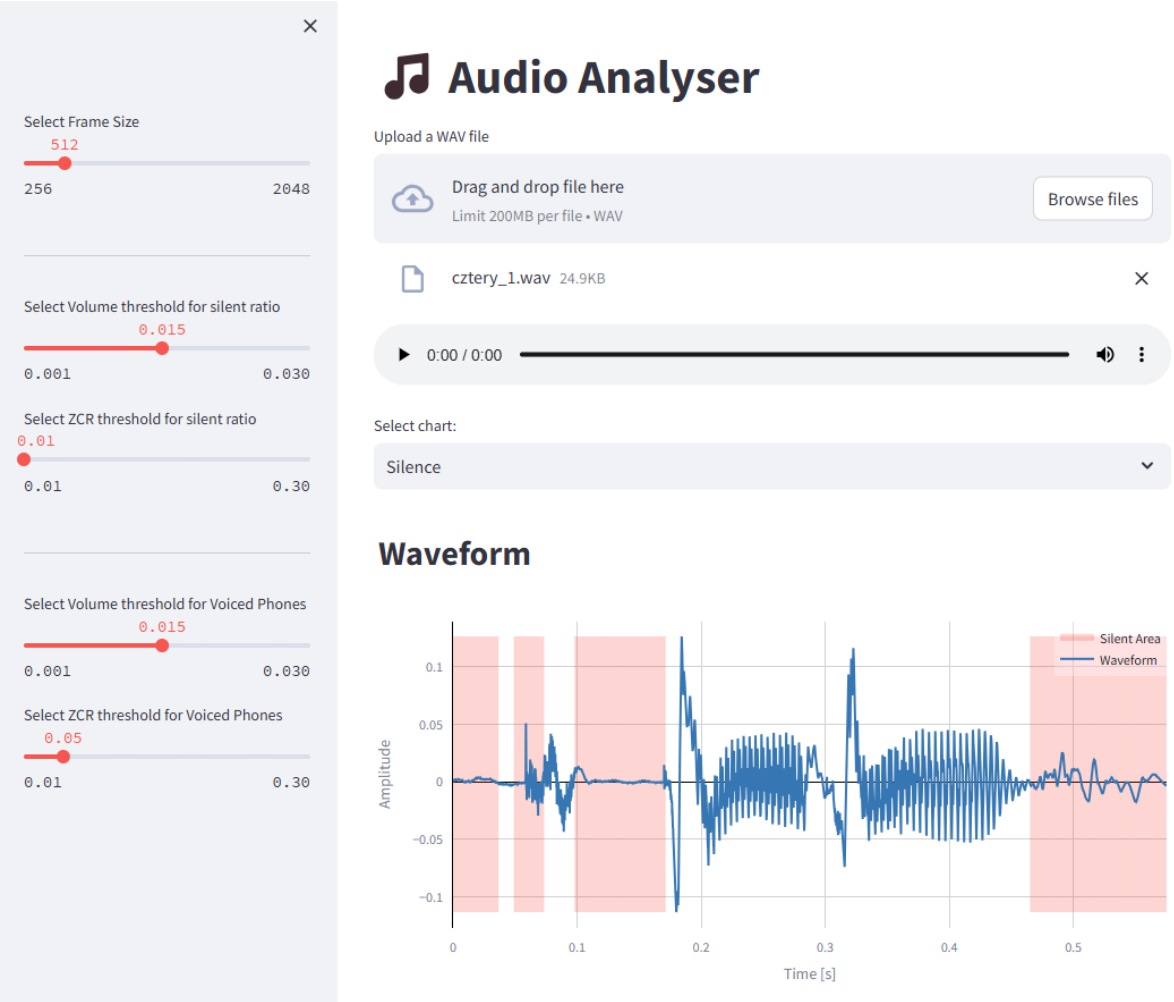
- Volume
- Short Time Energy (STE)
- Zero Crossing Rate (ZCR)
- Fundamental Frequency (F0) - Autocorrelation
- Fundamental Frequency (F0) - AMDF

Kolejna sekcja to analiza parametrów na poziomie klipu (fragmentu audio o długości jednej sekundy). W tabeli znajdują się następujące parametry:

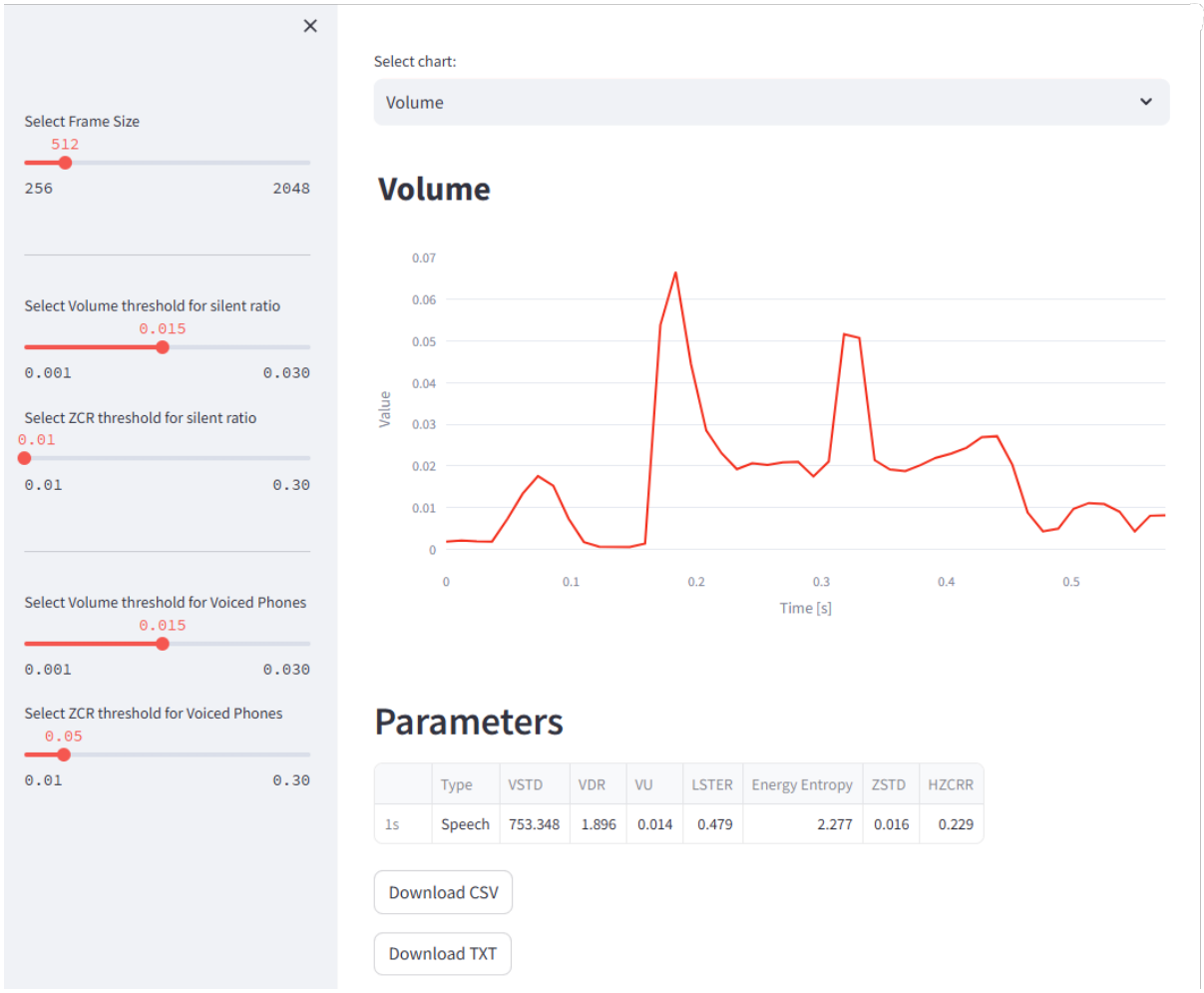
- VSTD - współczynnik zmienności,
- VDR - współczynnik rozpiętości,
- VU - średnia wartość amplitudy,
- LSTER - wskaźnik krótkoterminowej energii,
- Energy Entropy - entropia energii,
- ZSTD - zmienność ZCR,
- HZCRR - współczynnik wykrywania zmian ZCR.

Aplikacja próbuje również rozpoznać, czy dany klip to mowa, czy muzyka, w oparciu o wyliczone parametry. W przypadkach, gdy analiza nie daje jednoznacznych wyników, klip może zostać oznaczony jako „Unknown". Tabelę parametrów na poziomie klipu można wyeksportować do pliku CSV.

Ostatnią częścią aplikacji są przyciski eksportu. Podstawowe informacje na temat nagrania oraz parametry dla poszczególnych ramek możemy wyeksportować do pliku CSV lub TXT.



Rysunek 1: Wygląd aplikacji dla przykładowego audio cz. 1



Rysunek 2: Wygląd aplikacji dla przykładowego audio cz. 1

4 Struktura aplikacji

Aplikacja składa się z pliku głównego aplikacji: `app.py` oraz z folderu `tools`, w którym znajdują się wszystkie pomocnicze pliki. Poniżej znajdują się opisy poszczególnych plików i ich funkcjonalności.

4.1 app.py

Plik `app.py` zawiera główną klasę **AudioAnalyzerApp**, która odpowiada za cały interfejs aplikacji. Zawiera on kod odpowiedzialny za ładowanie pliku audio za pomocą biblioteki `librosa`, która od razu normalizuje dźwięk do zakresu `[-1, 1]`, obsługuje również odtwarzanie audio. Następnie definiuje suwaki w panelu bocznym, rysuje wykresy oraz pola wyboru do nich, wyświetla parametry, a także dodaje przyciski do eksportowania danych. Aplikacja jest w pełni interaktywna dzięki bibliotece Streamlit.

Rysunek 3: Panel ładowania oraz odtwarzania audio

4.2 tools

Folder `tools` zawiera pliki pomocnicze, które realizują poszczególne funkcjonalności.

4.2.1 audio_params.py

Plik `audio_params.py` zawiera funkcje odpowiedzialne za obliczanie parametrów dźwięku na poziomie ramek audio. Należą do nich: Volume, Short Time Energy (STE), Zero Crossing Rate (ZCR), Fundamental Frequency (F0) obliczane za pomocą autokorelacji oraz AMDF (Average Magnitude Difference Function) oraz detekcja ciszy i głosek dźwięcznych na podstawie wcześniej wymienionych parametrów. Poniżej znajuje się opis każdego z tych parametrów.

Głośność (Volume)

Głośność jest wyliczana jako pierwiastek średniej energii sygnału audio w klatce, za pomocą wzoru:

$$p(n) = \sqrt{\frac{1}{N}\sum_{i=0}^{N-1} s_n^2(i)} \tag{1}$$

Gdzie N to liczba próbek w ramce, a $s_n(i)$ to amplituda i-tej próbki w n-tej ramce.

Short Time Energy (STE)

Jest to głośność podniesiona do kwadratu:

$$STE(n) = \frac{1}{N}\sum_{i=0}^{N-1} s_n^2(i) \tag{2}$$

Zero Crossing Rate (ZCR)

ZCR mierzy, jak często sygnał zmienia znak w danej ramce. Obliczony według wzoru:

$$Z(n) = \frac{1}{2N}\sum_{i=1}^{N-1} |\text{sign}(S_n(i)) - \text{sign}(S_n(i-1))| \tag{3}$$

Tutaj `sign()` to finkcja signum.

Fundamental Frequency (F0)

Częstotliwość podstawowa $F0$ określa wysokość dźwięku. Jest ona wyliczana na dwa sposoby, za pomocą funkcji autokorelacji:

$$R_n(l) = \sum_{i=0}^{N-l-1} S_n(i)S_n(i+l) \tag{4}$$

lub za pomocą funkcji AMDF:

$$A_n(l) = \sum_{i=0}^{N-l-1} |S_n(i) - S_n(i+l)| \tag{5}$$

Silent Ratio

Jest to parametr, który przyjmuje wartość `True`, jeśli dana klatka jest zakwalifikowana jako cisza oraz `False` w przeciwnym przypadku. Klasyfikacja następuje za pomocą parametru Volume oraz Zero Crossing Rate. Za ciszę uznajemy klatki dla których jest spełniona zależność:

$$SR = (V < V_{threshold}) \wedge (ZCR > ZCR_{threshold}) \tag{6}$$

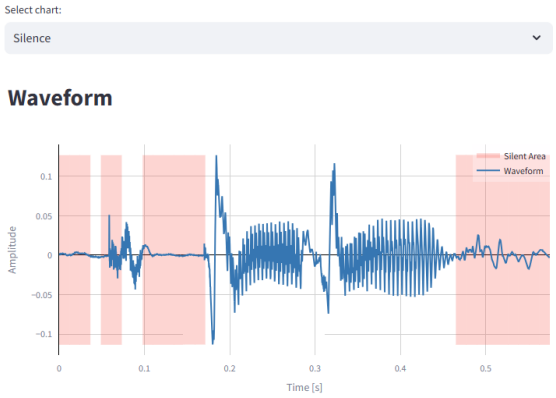
gdzie dla głośności domyślnie $V_{threshold} = 0.008$, ale możemy regulować tę wartość za pomocą suwaka w zakresie $0.001 - 0.03$, a dla ZCR domyślnie $ZCR_{threshold} = 0.07$ z suwakiem o zakresie $0.01 - 0.3$.

Voiced Ratio

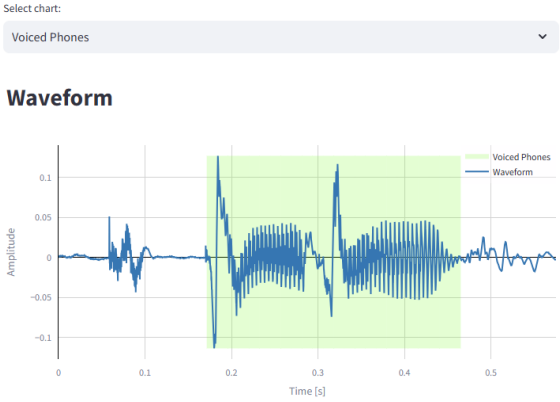
Jest to parametr, który przyjmuje wartość `True`, jeśli dana klatka jest zakwalifikowana jako fragment dźwięczny oraz `False` dla fragmentu bezdźwięcznego. Klasyfikacja następuje za pomocą parametru Volume oraz Zero Crossing Rate. Za fragmenty dźwięczne uznajemy klatki dla których jest spełniona zależność:

$$VR = (V > V_{threshold}) \wedge (ZCR < ZCR_{threshold}) \tag{7}$$

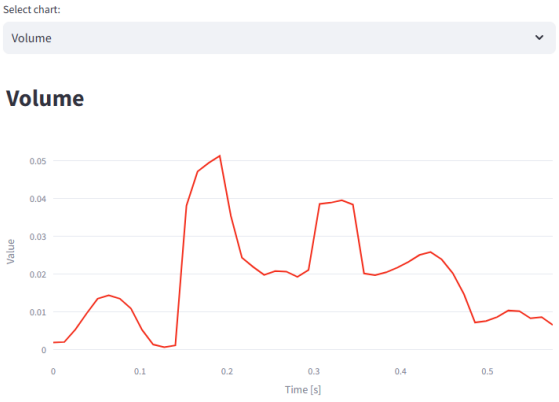
gdzie dla głośności domyślnie $V_{threshold} = 0.015$, ale możemy regulować tę wartość za pomocą suwaka w zakresie $0.001 - 0.03$, a dla ZCR domyślnie $ZCR_{threshold} = 0.05$ z suwakiem o zakresie $0.01 - 0.3$.



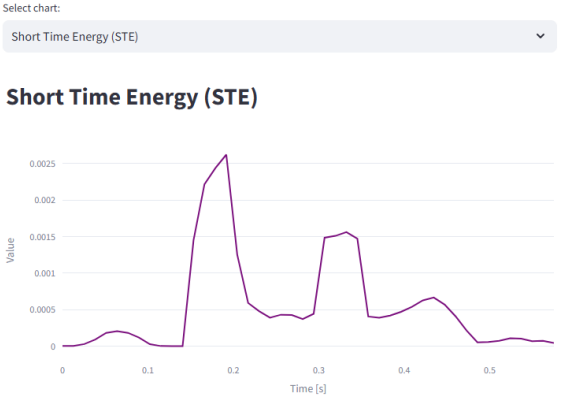
Rysunek 4: Wykres Waveform z zaznaczonymi fragmentami ciszy



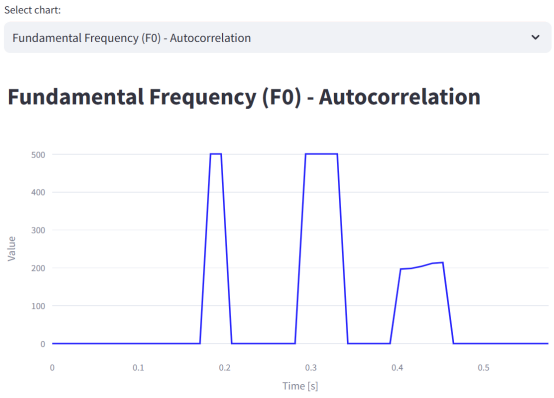
Rysunek 5: Wykres Waveform z zaznaczonymi fragmentami dźwięcznymi



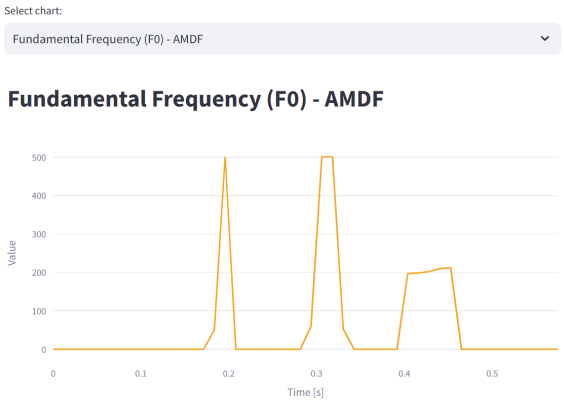
Rysunek 6: Wykres parametru Volume



Rysunek 7: Wykres parametru STE



Rysunek 8: Wykres parametru F0 (autokorelacja)



Rysunek 9: Wykres parametru F0 (AMDF)

Rysunek 10: Wykres parametru ZCR

4.2.2 clip_params.py

Plik `clip_params.py` odpowiedzialny jest za obliczanie parametrów dźwięku na poziomie klipu (fragmentu dźwięku o długości jednej sekundy). Parametry obejmują zmienność amplitudy (VSTD), rozpiętość amplitudy (VDR), średnią amplitudę (VU) oraz inne parametry, takie jak ZSTD (zmienność ZCR) oraz HZCRR (współczynnik detekcji ZCR). Ponadto, na podstawie tych parametrów, aplikacja próbuje klasyfikować fragment audio jako mowa lub muzyka. Poniżej znajuje się opis każdego z tych parametrów.

VSTD

Jest to miara określająca zmienność głośności w klipie. Obliczana za pomocą wzoru:

$$VSTD = \frac{\sigma_{\text{clip}}}{\mu_{\text{clip}}}, \quad \text{dla} \quad \mu_{\text{clip}} \neq 0, \tag{8}$$

gdzie σ_{clip} to odchylenie kwadratowe sygnału, a μ_{clip} to średnia wartość sygnału.

VDR

Określa różnicę między maksymalną a minimalną wartością sygnału, znormalizowaną względem wartości maksymalnej:

$$VDR = \frac{\max(\text{clip}) - \min(\text{clip})}{\max(\text{clip})}, \quad \text{dla} \quad \max(\text{clip}) \neq 0 \tag{9}$$

VU

Jest to średnia bezwzględna wartość amplitudy.

$$VU = \frac{1}{N} \sum_{i=1}^{N} |\text{clip}_i| \tag{10}$$

LSTER

Określa proporcję ramek o energii mniejszej niż połowa średniej energii w oknie 1 sekundowym.

$$LSTER = \frac{1}{2N} \sum_{n=0}^{N-1} \left(\text{sign}(0.5 * avSTE - STE(n)) + 1\right), \tag{11}$$

gdzie N jest liczbą ramek, $STE(n)$ to STE w n-tej ramce i $avSTE$ jest średnią wartością STE w 1 sekundowym oknie.

Energy Entropy

Opiera się na znormalizowanej energii sygnału na poziomie poszczególnych ramek, które oznaczamy jako σ_i.

$$I = -\sum_{i=1}^{J} \sigma_i^2 \log_2 \sigma_i^2 \tag{12}$$

ZSTD

Mierzy odchylenie standardowe częstości przejść sygnału przez zero w ramkach:

$$ZSTD = \sigma(ZCR) \tag{13}$$

HZCRR

Określa odsetek ramek, w których częstość przejść przez zero przekracza 1.5-krotność średniej wartości:

$$HZCRR = \frac{1}{2N} \sum_{n=0}^{N-1} \left(\text{sign}(ZCR(n) - 1.5 * avZCR) + 1\right), \tag{14}$$

gdzie N jest liczbą ramek, $ZCR(n)$ to ZCR w n-tej ramce i $avZCR$ jest średnią wartością ZCR w 1 sekundowym oknie.

Klasyfikacja mowa/muzyka

Na podstawie wartości LSTER i ZSTD określany jest rodzaj dźwięku:

$$\text{Typ} = \begin{cases} \text{Speech}, & LSTER > 0.3 \text{ i } ZSTD > 0.01 \\ \text{Music}, & LSTER < 0.3 \text{ i } ZSTD < 0.01 \\ \text{Unknown}, & \text{w przeciwnym razie} \end{cases} \tag{15}$$

Parameters

	Type	VSTD	VDR	VU	LSTER	Energy Entropy	ZSTD	HZCRR
1s	Speech	-407.276	2.125	0.017	0.477	2.888	0.053	0.244
2s	Speech	374.753	2.486	0.023	0.349	2.884	0.024	0.07
3s	Speech	286.317	2.234	0.016	0.407	2.774	0.016	0.163
4s	Unknown	-82.074	2.476	0.004	0.292	2.961	0.006	0.167

Rysunek 11: Tabela parametrów dla klipów

4.2.3 waveform_plot.py

Plik `waveform_plot.py` zawiera funkcje do rysowania wykresu przebiegu czasowego sygnału audio. Dodatkowo, do wyboru mamy zaznaczenie fragmentów ciszy lub fragmentów dźwięcznych, opartych na wcześniej obliczonych parametrach Volume i ZCR.

4.2.4 params_plot.py

Plik `params_plot.py` zawiera funkcje do rysowania wykresów parametrów dźwięku na poziomie ramek. Użytkownik może wybierać wykresy przedstawiające różne parametry, takie jak: Volume, Short Time Energy (STE), Zero Crossing Rate (ZCR), Fundamental Frequency (F0) obliczane za pomocą autokorelacji oraz AMDF.

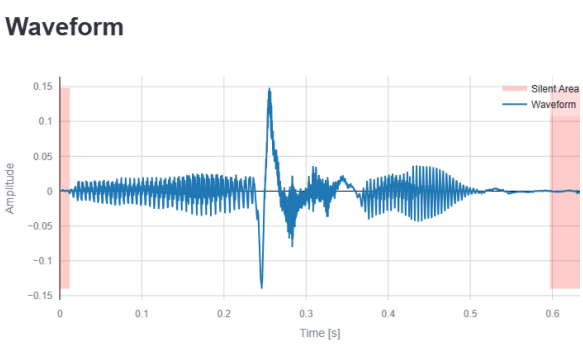
4.2.5 export_data.py

Plik `export_data.py` zawiera funkcje odpowiedzialne za eksportowanie danych. Parametry audio na poziomie ramek oraz podstawowe informacje o pliku audio mogą być wyeksportowane do pliku CSV lub TXT. Dodatkowo każdy wykres możemy pobrać w formacie PNG.

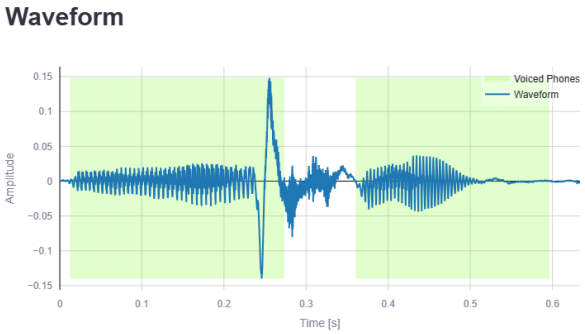
5 Przykłady i porównania

5.1 Wykrywanie ciszy oraz głosek dźwięcznych na podstawie nagrań głosów

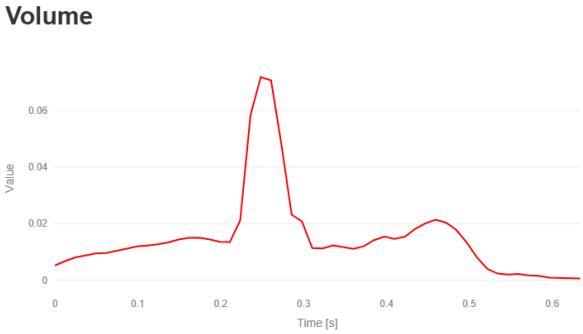
Jako przykład przedstawię wyniki dla słowa **"alesza"**, gdzie głoska "sz"jest bezdźwięczna, a pozostałe są dźwięczne.
Możemy zauważyć, że aplikacja pozwoliła wykryć fragmenty ciszy na początku i końcu nagrania oraz poprawnie wykryła głoski dźwięczne. Analizując wykresy Volume oraz ZCR możemy zauważyć, że na początku i końcu nagrania mamy niższą wartość Volume i wyższą wartość ZCR, co wspazuje na ciszę, a w okolicy 0.3s mamy niższą wartość Volume i bardzo wysoką wartość ZCR, odpowiada to bezdźwięcznej głosce "sz".



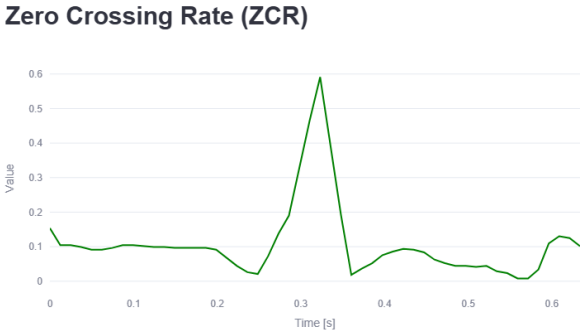
Rysunek 12: Detekcja ciszy dla słowa alesza



Rysunek 13: Detekcja fragmentów dźwięcznych dla słowa alesza



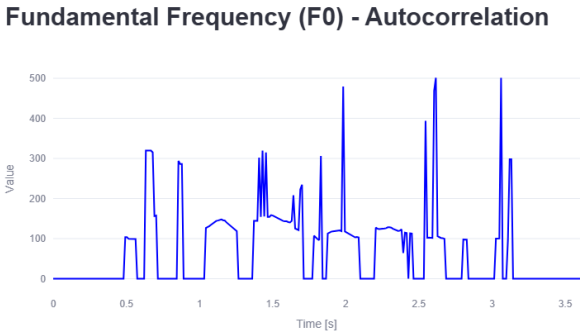
Rysunek 14: Wykres Volume dla słowa alesza



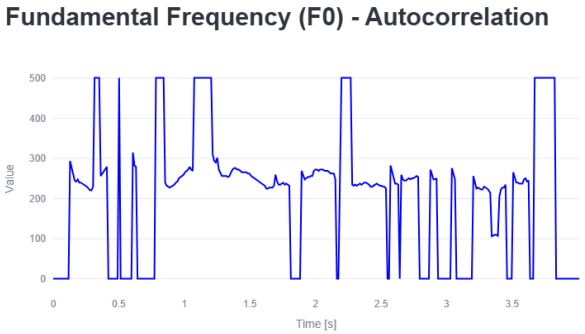
Rysunek 15: Wykres ZCR dla słowa alesza

5.2 Porównanie: głos męski i żeński

Porównując głos męski i żeński powinniśmy przede wszystkim zwrócić uwagę na parametr Fundamental Frequency. Głos żeński powinien mieć wyższą częstotliwość. To też możemy zaobserwować na poniższych wykresach.



Rysunek 16: Wykres dla głowy męskiego



Rysunek 17: Wykres dla głosu żeńskiego

5.3 Porównanie: mowa i muzyka

W celu porównania parametrów dla muzyki i mowy, wykorzystam przykładowe nagranie dźwięku dla mowy (`https://www.youtube.com/watch?v=VmAYsDAhEBU`) i przykładową muzykę (`https://pixabay.com/sound-effects/search/background%20music/`). Program odpowiednio zakwalifikował audio jako muzyka i mowa. Najważniejszą obserwacją jest fakt, że dla muzyki mamy zauważalnie niższy współczynnik LSTER i ZSTD niż dla mowy. Na tej podstawie przebiega klasyfikacja.

	Type	VSTD	VDR	VU	LSTER	Energy Entropy	ZSTD	HZCRR
1s	Unknown	-116.11	1.793	0.089	0.233	3.061	0.099	0.07
2s	Music	-665.31	2.037	0.098	0.262	2.991	0.007	0.163
3s	Music	352.383	2.135	0.103	0.233	2.978	0.007	0.192
4s	Music	-976.554	2.16	0.096	0.208	3.079	0.008	0.15
5s	Music	-173.041	1.695	0.094	0.163	3.082	0.008	0.157
6s	Music	-379.33	1.98	0.092	0.256	3.104	0.008	0.192
7s	Music	-140.145	2.101	0.094	0.203	3.088	0.007	0.14
8s	Unknown	233.838	2.192	0.095	0.246	3.183	0.01	0.175

Rysunek 18: Parametry dla fragmentu muzyki

	Type	VSTD	VDR	VU	LSTER	Energy Entropy	ZSTD	HZCRR
1s	Speech	-1,348.1479	2.591	0.071	0.599	2.397	0.038	0.256
2s	Speech	937.165	2.378	0.059	0.337	2.856	0.035	0.128
3s	Speech	1,758.307	2.249	0.04	0.523	2.235	0.022	0.285
4s	Speech	3,888.823	2.087	0.068	0.486	2.862	0.037	0.179
5s	Speech	-978.961	2.33	0.081	0.483	2.771	0.025	0.128
6s	Speech	-1,822.615	2.288	0.041	0.651	2.562	0.032	0.256
7s	Speech	-1,523.642	2.562	0.055	0.523	2.286	0.044	0.209
8s	Speech	-1,543.756	1.9	0.046	0.738	1.734	0.052	0.213

Rysunek 19: Parametry dla fragmentu mowy

6 Źródła

- https://pages.mini.pw.edu.pl/~rafalkoj/www/?Dydaktyka:2024%2F2025:-_Analiza_i_przetwarzanie_d%C5%BAwi%C4%99ku
- Audacity
- https://www.youtube.com/watch?v=VmAYsDAhEBU
- https://pixabay.com/sound-effects/search/background%20music/
- ChatGPT
- https://docs.streamlit.io/