STRESZCZENIE

Określanie dźwięków instrumentów bywa często nazywane kwestia subiektywną, ale istnieją pewne cechy charakterystyczne pozwalające na dywersyfikacje ich brzmień. Różnice pomiędzy nimi wywodzą się z właściwości fizycznych tych fal (drgań). Łatwy oraz szybki dostęp do określania tych brzmień mógłby zagwarantować lepszy dobór instrumentów do nastroju i charakterystyki dzieła muzycznego. Praca ma na celu znalezienie tych różnic pomiędzy brzmieniami, oraz przygotowaniu urządzenia które pozwoliłoby na określenie ich rodzaju. Do wykonania projektu wybrano instrument klawiszowy, a następnie zebrano próbki klawisza A4 (a razkreślne) o częstotliwości 440 Hz dla dwóch powszechnie i umownie określonych brzmień: „jasny”, „miękki”. Po znalezieniu zależności, polegających na intensywnościach oraz liczby składowych drgań harmonicznych dźwięku, przygotowano MEL-spektrogramy danych treningowych. Użyto ich w tworzeniu modelu konwolucyjnej sieci neuronowej zdolnej do rozróżnienia tych klas. Ze względu na ograniczenia czasowe oraz porównaniu modeli tworzonych własnoręcznie, postanowiono skorzystać z frameworku „Teachable Machines” autorstwa Google, udostępnionego do użytku publicznego, pozwalającego na stworzenie modelu będącego w stanie dokonać predykcji z zadowalającą dokładnością. Następnie przygotowano przenośne urządzenie pozwalające na użytkowanie modelu, Wyniki projektu wykazują, że różnice występujące w dźwiękach nie tylko pozwalają na analityczne rozróżnienie barw, ale także na stworzenie modelu będącego w stanie je określać.

Słowa kluczowe:

MEL-spektrogramy, muzyka, analiza dźwięku, sztuczna inteligencja, konwolucyjne sieci neuronowe, programowanie, Python, komputer jednopłytkowy, RaspberryPi, urządzenia embedded.

ABSTRACT

Defining the sounds of instruments is often called a subjective matter, but there are certain characteristics that allow for the diversification of their sounds. The differences between them are derived from the physical properties of these waves (vibrations). Easy and quick access to defining these sounds could guarantee a better selection of instruments for the mood and characteristics of a musical work. The work aims to find these differences between sounds, and to prepare a device that would allow for determining their type. A keyboard instrument was selected for the project, and then samples of the A4 (middle A) key with a frequency of 440 Hz were collected for two commonly and conventionally defined sounds: "bright", "soft". After finding the relationships, consisting in the intensities and the number of components of harmonic vibrations of the sound, MEL-spectrograms of training data were prepared. They were used to create a convolutional neural network model capable of distinguishing these classes. Due to time constraints and the comparison of models created by hand, it was decided to use the Google "Teachable Machines" framework, made available for public use, which allows the creation of a model capable of making predictions with satisfactory accuracy. Then, a portable device was prepared to allow the use of the model. The results of the project show that the differences occurring in sounds not only allow for analytical distinction of colors, but also for the creation of a model capable of defining them.

Keywords:

MEL-spectrograms, music, sound analysis, artificial intelligence, convolutional neural networks, programming, Python, single-board computer, RaspberryPi, embedded devices.

Spis treści

[WYKAZ WAŻNIEJSZYCH SŁÓW I SKRÓTÓW 6](#_Toc193717552)

[1. CEL PRACY 7](#_Toc193717553)

[2. WSTĘP TEORETYCZNY 8](#_Toc193717554)

[2.1. Dźwięk oraz jego analiza 9](#_Toc193717555)

[2.1.1. Charakterystyka dźwięku oraz badanego instrumentu 9](#_Toc193717556)

[2.1.2. MEL-spektrogramy 15](#_Toc193717557)

[2.1.3. Transformata Fouriera 18](#_Toc193717558)

[2.2.1. Działanie oraz zastosowanie sztucznych sieci neuronowych 20](#_Toc193717559)

[2.2.2. Konwolucyjne sieci neuronowe i ich zastosowanie w projekcie 21](#_Toc193717560)

[2.3.1. Charakterystyka języka 22](#_Toc193717561)

[2.3.2. Wykorzystane biblioteki 23](#_Toc193717562)

[2.4. Urządzenia embedded 24](#_Toc193717563)

[2.4.1. Ogólna charakterystyka 25](#_Toc193717564)

[2.4.2. RaspberryPi 26](#_Toc193717565)

[3. PRZEBIEG BADANIA I METODOLOGIA 27](#_Toc193717566)

[3.1. Dane 28](#_Toc193717567)

[3.1.1. Pozyskiwanie danych 29](#_Toc193717568)

[3.1.2. Analiza 30](#_Toc193717569)

[3.1.3. Przygotowanie zbiorów treningowych 37](#_Toc193717570)

[3.2. Stworzenie modelu sieci neuronowej 38](#_Toc193717571)

[3.3. Aplikacja do klasyfikacji dźwięku 39](#_Toc193717572)

[3.3.1. Struktura i działanie aplikacji 40](#_Toc193717573)

[3.3.2. Elementy graficzne 41](#_Toc193717574)

[3.3.3. Implementacja i funkcjonalność modelu sieci 42](#_Toc193717575)

[4. PODSUMOWANIE 43](#_Toc193717576)

[WYKAZ LITERATURY 44](#_Toc193717577)

[WYKAZ RYSUNKÓW 46](#_Toc193717578)

[MATERIAŁY DODATKOWE 47](#_Toc193717579)

# WYKAZ WAŻNIEJSZYCH SŁÓW I SKRÓTÓW

Hz – Herc, jednostka miary częstotliwości.

CNN – konwolucyjna sieć neuronowa (od ang. Convolutional Neural Network).

zmienna – w języku programowania oznacza to nośnik danych w programie bądź funkcji .

funkcja – w języku programowania jest to definiowalna metoda, która przyjmuje zmienne oraz wykonuje operacje logiczne.

epoka – jedno kompletne przejście zestawu danych treningowych przez algorytm

barwa dźwięku – inne określenie na charakterystykę brzmienia instrumentu.

dynamika – w kontekście muzycznym jest to opisanie sposobów gry na fortepianie (np. głośno, cicho), polega na kontrolowaniu siły użytej podczas przyciskania klawiszy.

MFCC – (od ang. Mel-Frequency Cepstrum Coefficients) współczynniki cepstralne skali MEL.

STFT – (od and. Short Time Fourier Transform) to jedna z technik analizy sygnału zmiennego w czasie.

Wartość znormalizowana – wartość znajdująca się w zbiorze

RGB – (od ang. Red, Green, Blue) jest to przestrzeń barw w której za pomocą modulacji nasycenia trzech składowych kolorów; czerwonego, zielonego oraz niebieskiego można otrzymać wiele innych kolorów.

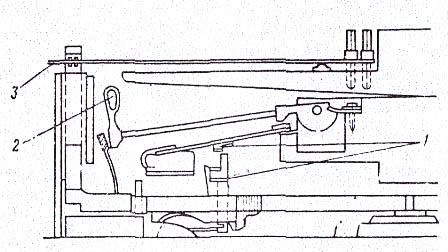
A4 – A razkreślne, dźwięk instrumentu klawiszowego, którego częstotliwość dźwięku to 440Hz

# 1. CEL PRACY

Praca ma na celu stworzenie urządzenia zdolnego do wykrywania różnic w drganiach harmonicznych dźwięku A4, którego częstotliwość to 440 Hz. Wykrywanie odbywa się za pomocą modelu sieci neuronowej. Dzięki przeprowadzonym badaniom znaleziono różnice pomiędzy intensywnościami i występowaniem harmonicznych pomiędzy poszczególnymi brzmieniami instrumentu. Dzięki wytrenowaniu modelu sieci w celu klasyfikacji tych brzmień, możliwe jest stworzenie przenośnego urządzenia pozwalającego na łatwiejsze określanie brzmienia, co może ułatwić wybór i przygotowania instrumentów do odtwarzania specyficznych utworów, bądź dodawania detali w nowych kompozycjach.

# 2. WSTĘP TEORETYCZNY

Muzyka towarzyszy człowiekowi od bardzo dawna, a jej prymitywne korzenie szacuje się na czasy pradawne. Mimo tak rozległego czasu istnienia, forma oraz instrumenty muzyczne są rozwijane po dzień dzisiejszy. Dźwięk nie jest niczym innym, niż falą mechaniczną, która jest odbierana poprzez człowieka za pomocą wibracji[1]. Dzięki rozwojowi instrumentów oraz sposobów wydobywania z nich brzmień, powstała możliwość stworzenia isntrumentu jakim jest fortepian. Działanie tego instrumentu można uprościć do prostego modelu młoteczka uderzającego w strunę o zdefiniowanym naprężeniu (nastrojeniu), powodując różnicę w amplitudzie drgań tej struny. Na poniższym rysunku przedstawiono uproszczony przekrój prostego mechanizmu dla 1 klawisza:



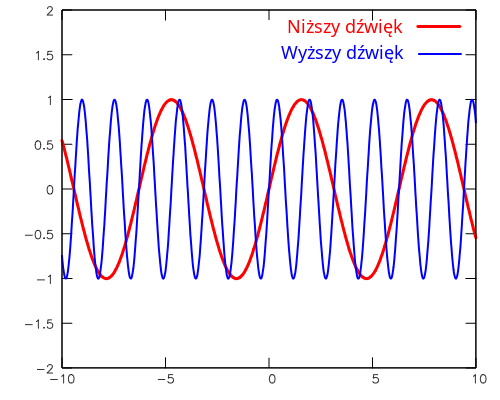
Rys 2.1 – Schemat mechanizmu, gdzie 1 to dźwignie, 2 to młoteczek a 3 to struna

Pomimo faktu podobnego, a wręcz ideowo identycznego, mechanizmu zawartego we wszystkich instrumentach fortepianowych możemy usłyszeć różnice w odniesieniu do postrzegania wydobywających się z nich dźwięków. Możliwe jest skategoryzowanie ich na subiektywnie stworzone kategorie; niektóre brzmienia są bardziej miękkie oraz delikatniejsze, z kolei inne bardziej wyraziste, jasne. Co prawda brzmienie te można uzyskiwać za pomocą kontrolowanej wartości siły podczas naciskania klawiszy, lecz możliwe jest spreparowanie lub stworzenie instrumentu, który cechowałby się specyficznym brzmieniem bez względu na dynamikę grania (bez względu na dynamikę gry cechy charakterystyczne barwy dźwięku dla danego instrumentu będą zawsze słyszalne).

## 2.1. Dźwięk oraz jego analiza

### 2.1.1. Charakterystyka dźwięku oraz badanego instrumentu

W fizyce dźwięk to wibracja, która rozprzestrzenia się jako fala akustyczna przez medium transmisyjne, takie jak gaz, ciecz lub ciało stałe[2]. Fala akustyczna jest opisana funkcja sinusoidalną, a wysokość dźwięku jest jednoznaczny z jej częstotliwością wprost proporcjonalnie; zwiększając częstotliwość podwyższamy dźwięk.



Rys 2.2 – Przykład fali akustycznej wraz z zilustrowaną zależnością

Poza główną częstotliwością, występują także częstotliwości harmoniczne. W fizyce i akustyce harmoniczna to fala sinusoidalna o częstotliwości będącej dodatnią całkowitą wielokrotnością częstotliwości podstawowej fali. Częstotliwość podstawowa jest również nazywana pierwszą harmoniczną; pozostałe harmoniczne są znane jako wyższe harmoniczne. Zestaw harmonicznych tworzy szereg harmoniczny[3], czyli w przypadku akustyki, słyszany przez człowieka dźwięk. Dokładniej zjawisko to zostanie opisane w podrozdziale 2.1.3.

W tym przypadku, wytwarzaną falę dźwiękową można opisać jako wypadkową falę, powstająca na wskutek wygaszania i wzmacniania fal powstających podczas odbijania się od punktu zamocowania drugiego końca struny (2.1) [4]:

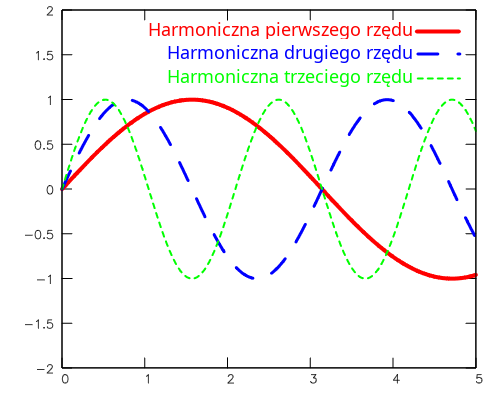
|  |  |
| --- | --- |
|  | (2.1) |

Gdzie to amplituda, k jest liczbą falowa (), x to położenie na strunie, jest częstością kołową, t jest równe czasowi a to przesunięcie fazowe.

Dla dowolnej częstotliwości, wizualizacje harmonicznych można opisać prostym wzorem funkcji sinusoidalnej, pomijając czynniki fizyczne (2.2):

|  |  |
| --- | --- |
|  | (2.2) |

Gdzie to rząd harmonicznej.



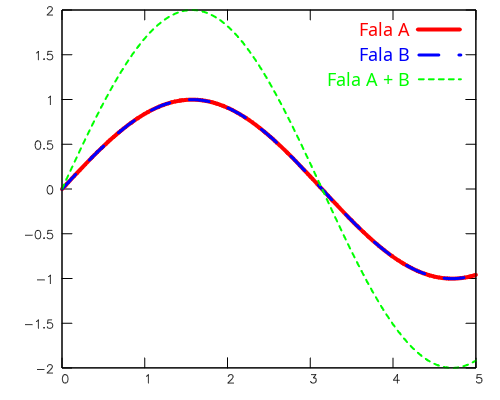
Rys 2.3 – Wizualizacja pierwszej harmonicznej wraz z dwoma wyższymi harmonicznymi dla dowolnej częstotliwości x

Zjawisko to jest kluczowe w zrozumieniu charakterystyk dźwięku instrumentu, ponieważ uderzana struna fortepianowa tworzy taką właśnie falę, która po głębszej analizie okazuje się być szeregiem harmonicznym. Ma to olbrzymi wpływ na znalezienie oraz zrozumienie różnic pomiędzy poszczególnymi brzmieniami. Dane wstępne zebrano z dostępnego instrumentu cyfrowego Yamaha Arius YDP-143.

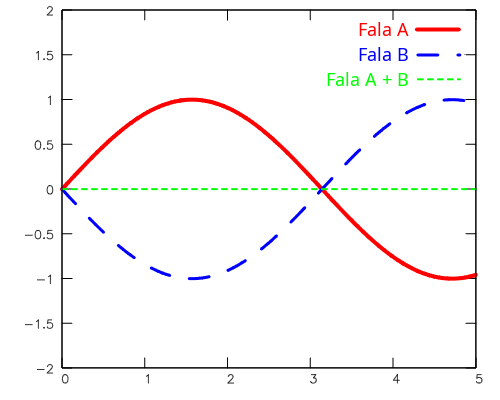


Rys 2.4 – Instrument którym posłużono się do zebrania wstępnych danych dźwiękowych

Fundamentalną różnicą pomiędzy instrumentem analogowym a cyfrowym jest sposób generowania dźwięku. W fortepianie polega to na fizycznym zjawisku wzbudzania struny w drgania. Najczęściej instrument ten posiada więcej strun przypadających na młoteczek, co przekłada się na ilości oraz intensywności harmonicznych. Bardzo ważną różnicą jest także obecność pudła rezonansowego, które amplifikuje dźwięk i pozwala na zachodzenie zjawisk fizycznych fal, takie jak wzmacnianie i wygaszanie [5]. Uproszczona wizualizacja tych interferencji została przedstawiona na rysunkach 2.5 oraz 2.6 poniżej.

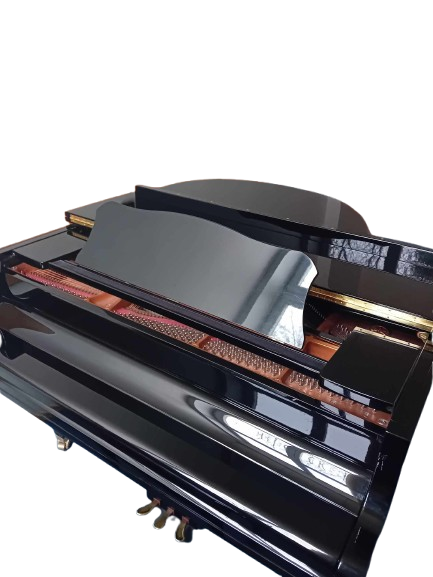


Rys 2.5 – Wizualizacja zjawiska wzmacniania fal stojących



Rys 2.6 – Wizualizacja zjawiska wygaszania fal stojących

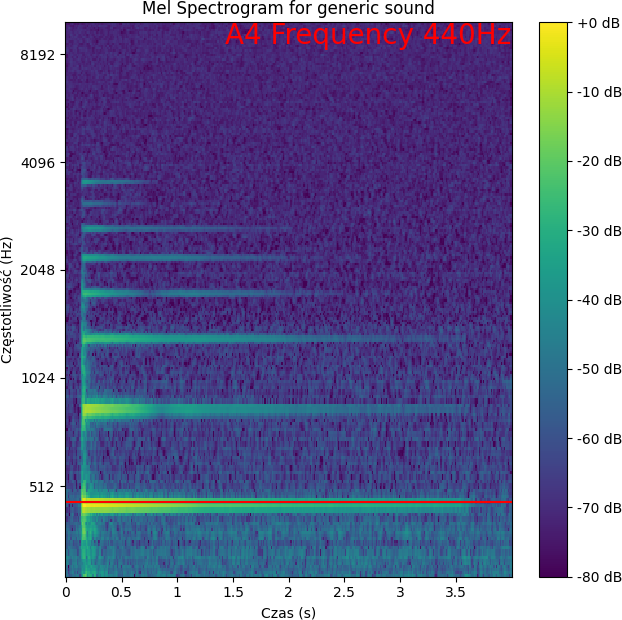
W instrumencie cyfrowym zjawiska te nie mają miejsca, ponieważ dźwięk wydobywa się przez głośniki urządzenia, co powoduje brak zachodzenia modulacji fal dźwiękowych. Skutkuje to gorszą jakością danych, ponieważ wydobywane są tylko niektóre harmoniczne, charakteryzujące specyfikację danego brzmienia. Dzięki udostepnieniu fortepianów przez Akademię Muzyczną im. Stanisława Moniuszki w Gdańsku zebrane dane dźwiękowe z instrumentów akustycznych, w celu poprawienia precyzji modelu. Są to mniejsze instrumenty fortepianowe Yamaha GB1 a stroju a1 = 442 HZ, posiadające po trzy struny tworzące dźwięki wyższe, w tym badany dźwięk A4.



Rys 2.7 – Zdjęcie jednego z fortepianów, na którym zebrano dokładniejsze dane.

### 2.1.2. MEL-spektrogramy

Spektrogramy MEL różnią się od zwykłych spektrogramów tym, że są przedstawiane na skali logarytmicznej.

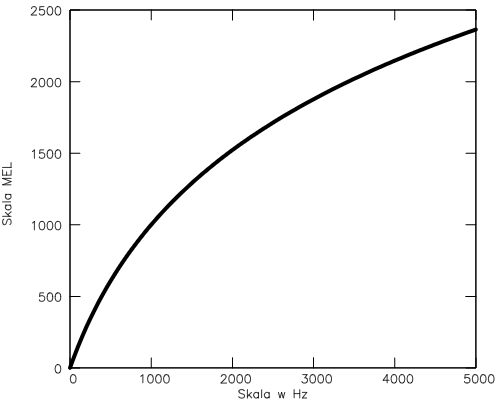


Rys 2.8 – przykład spektrogramu MEL stworzony z danych treningowych dla barwy „generic”

Skala ta jest wyliczona ze wzoru (2.3) [6]:

|  |  |
| --- | --- |
|  | (2.3) |

Gdzie jest skalą logarytmiczną normalnej skali częstotliwości .



Rys 2.9 – przykład funkcji skali melowej

Spektrogramy MEL są niezwykle skuteczne w rozpoznawaniu dźwięku oraz modelowaniu subiektywnej zawartości wysokości i częstotliwości sygnałów audio. Dzieje się tak za pomocą przedstawionej powyżej skali logarytmicznej, która pozwala na przedstawienie dźwięku za pomocą spektrogramu w klarowniejszy sposób. Dzięki temu zawrzeć możemy na wykresie więcej intensywniejszych częstotliwości harmonicznych, a same dane nie nachodzą na siebie, co nie będzie powodowało artefaktów w wyodrębnionych cechach charakterystycznych podczas treningu modelu CNN.

Posiadając skalę MEL spektrogramu jesteśmy w stanie zbadać występujące w sygnale cechy MFCC. Cepstrum to odwrotna transformata Fouriera widma sygnału. Opiera się na Fourierowskiej analizie sygnału wyrażonego w skali decybelowej (logarytmiczna skala Mel przedstawiona wzorem numer 3) [8] [9]. Sygnał analizowany jest za pomocą transformacji Fouriera, opisanej w podrozdziale 2.1.3. Cepstrum postrzegane jest jako informacja o prędkości zmian w poszczególnych pasmach częstotliwości widma o znanej funkcji. Służy do analizy widma akustycznego [10]. W przypadku tego projektu analiza jest oparta o występujące częstotliwości, ponieważ w każdym przypadku występuje dźwięk bazowy o tej samej częstotliwości, a mianowicie 440Hz. Przez wygaszanie fal w instrumencie dźwięk jest tłumiony po pewnym czasie, więc zebrane częstotliwości nie nachodzą na siebie. Dzięki cepstrum możliwe jest sporządzenie wykresu nasilenia występujących częstotliwości w zależności od czasu trwania dźwięku.

### 2.1.3. Transformata Fouriera

Analiza Fourierowska sygnału pozwala na analizę jego składowych, których wypadkowa funkcja będzie tożsama z badaną. Jest to niezbędny etap projektów, ponieważ mimo iż częstotliwość dźwięków jest taka sama, to różnice w ich brzmieniu są spowodowane różnymi intensywnościami występujących składowych harmonicznych. Różnice te zostały dokładnie opisane w podrozdziale 2.1.4.

Sygnały mogą zostać podzielone za pomocą kryterium opisujące ich zmienność w czasie. Za pomocą takiej segregacji możemy je opisać jako stacjonarne oraz niestacjonarne. Sygnały stacjonarne nie podlegają zmianom w trakcie ich trwania, a z kolei niestacjonarne ulegają różnym zjawiskom, takim jak: wzmacnianie, wygaszanie oraz inne zjawiska powodujące możliwość określenia składowych sygnałów polegając jedynie na prawdopodobieństwie [11]. Tradycyjna transformata Fouriera może zostać opisana za pomocą poniższego równania (2.4) dla ilości próbek sygnału [12]:

|  |  |
| --- | --- |
|  | (2.4) |

Gdzie jest jednym z zespolonych pierwiastków sygnału a jest jednostką urojoną. Dla i indeksy oraz należą do przedziału .

Dzięki takiej operacji, istnieje możliwość zbadania cech charakterystycznych składowych oryginalnego sygnału. Badany sygnał jest sygnałem niestacjonarnym, ponieważ w trakcie jego trwania podlega wygaszaniu; z powodu sił fizycznych działających na strunę zmniejsza ona w czasie amplitudę swoich drgań. Aby móc dokładnie zanalizować zmienny sygnał, konieczne jest wzięcie pod uwagę parametru czasowego dzięki któremu jesteśmy w stanie przeprowadzić analizę, biorąc pod uwagę zmianę sygnału w czasie. Stosując takie rozwiązanie możemy użyć krótkookresowej transformacji Fouriera (STFT), dzięki której jesteśmy poznać parametry sygnału w ramach chwili . Równanie opisujące tą transformatę to [11]:

|  |  |
| --- | --- |
|  | (2.5) |

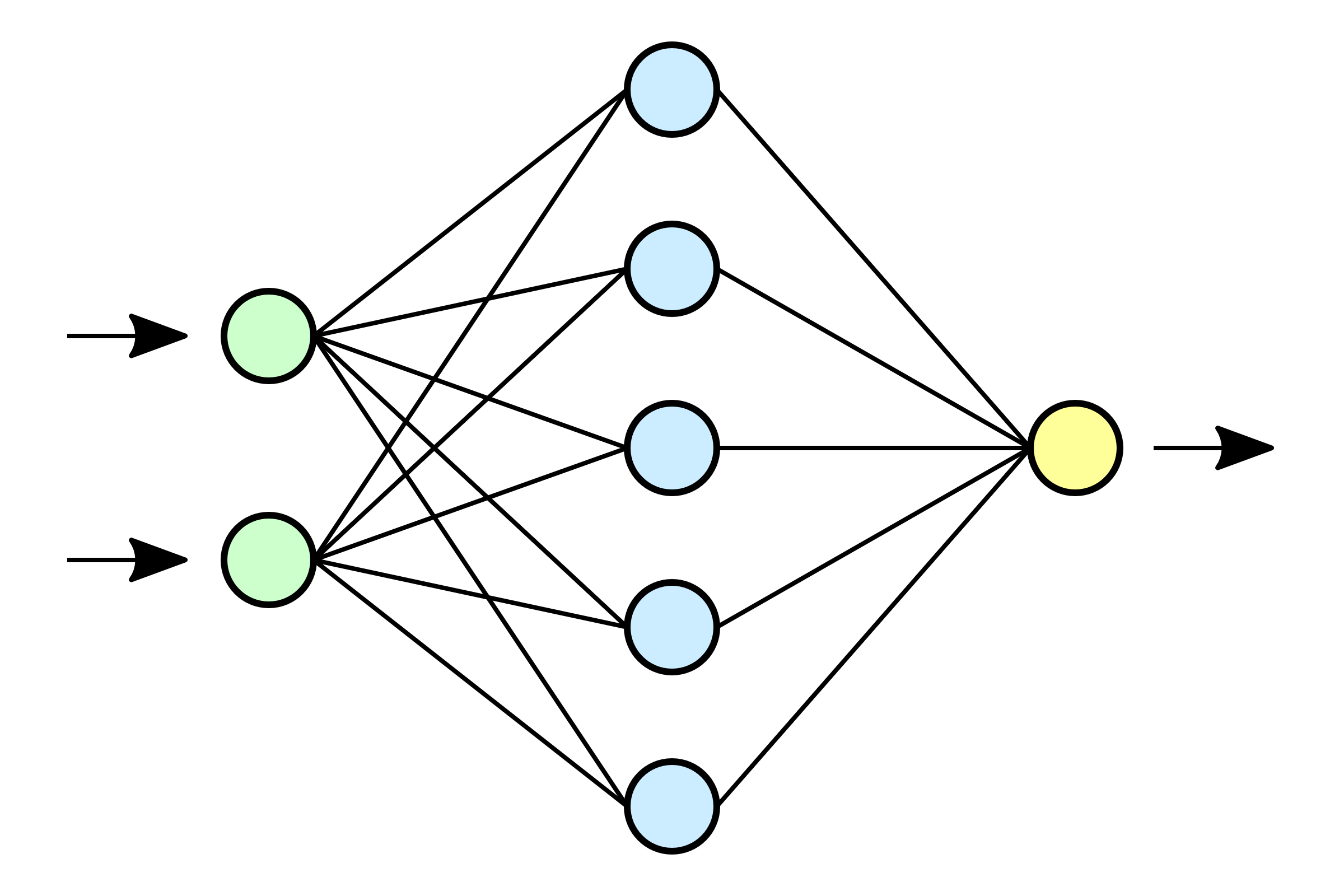
Gdzie to sygnał w chwili obserwowalny w wybranej ramie czasowej, z czego funkcja pozwala na anihilację funkcji znajdującej się poza danym oknem czasowym. Opisana dokładniej w podrozdziale 2.3.2, biblioteka języka python librosa pozwala na tworzenie spektrogramów melowych za pomocą STFT [13] dostosowując parametry funkcji.

2.2. Sztuczne sieci neuronowe

### 2.2.1. Działanie oraz zastosowanie sztucznych sieci neuronowych

Sieci neuronowe to algorytmy będące w stanie rozpoznawać i uczyć się schematów. Wykorzystywane są w wielu dziedzinach, zaczynając od przedmiotów codziennego użytku a kończąc na medycynie. Nieskomplikowane modele przedstawiane są za pomocą trzech warstw.

Wejściowa, ukryta oraz wyjściowa.

**

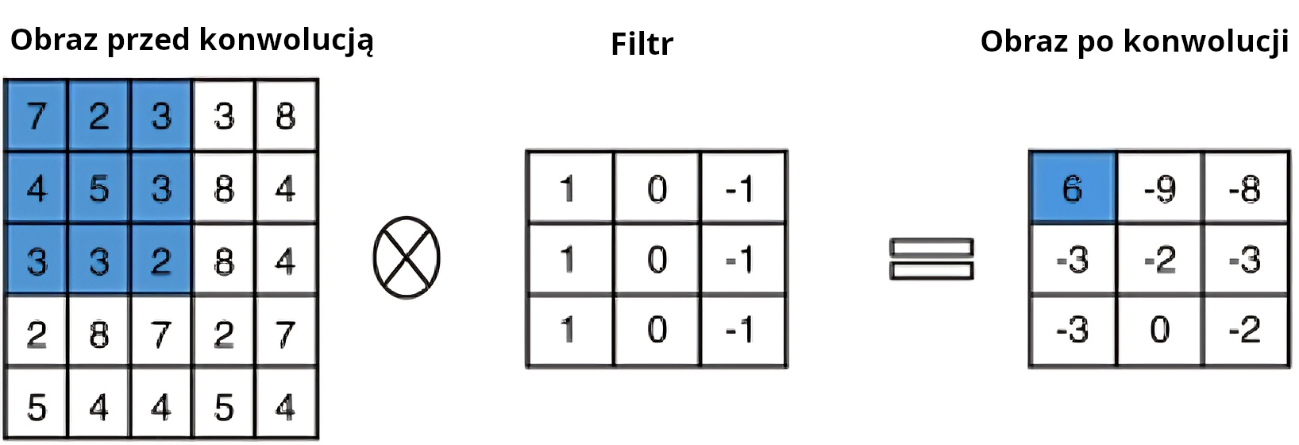
Rys 2.10 – Przykład budowy sztucznej sieci neuronowej, zielonym kolorem zostały zaznaczone warstwa wejściowa, błękitnym warstwa ukryta a żółtym wyjściowa

Każda z poszczególnych warstw pełni inną rolę. Warstwa wejściowa jest odpowiedzialna za odczytanie danych i przekazaniu jej do warstwy ukrytej, w której dokonywany jest rachunek wagowy. Ostatnia warstwa dokonuje predykcji na podstawie wyliczonego w warstwie ukrytej rachunku prawdopodobieństwa [14]. Warto dodać że wartości, na których działa algorytm, są znormalizowane. Praca opisuje działanie sieci neuronowych pobieżnie, skupiając się tylko na podstawach oraz konwolucyjnej sieci neuronowej. W przypadku uczenia maszynowego nadzorowanego, stosowanego w tym projekcie, model szuka cech charakterystycznych obrazów dostarczonych w zbiorze treningowych. Uczenie trwa przez określoną liczbę epok, dzięki którym algorytm stara się zoptymalizować wartości wag, aby dobra klasyfikacja dążyła swoją wartością do 1. Gdy wartość predykcji lub klasyfikacji jest zbyt idealna i występuje zjawisko przeuczenia sieci, model nauczył się stuprocentowo rozwiązywać przykłady ze zbioru walidacyjnego, ale ma olbrzymie trudności, a w ręcz niemożliwe jest, aby poprawnie sklasyfikował przypadek nienależący do tego zbioru.

### 2.2.2. Konwolucyjne sieci neuronowe i ich zastosowanie w projekcie

Konwolucyjna sieć neuronowa to specyficzny algorytm, który został stworzony w celu radzenia sobie z analizą zdjęć. Dzięki rozwojowi i optymalizacji modeli, bardzo częstym przypadkiem w analizie dźwięku miejsce znajdują właśnie te algorytmy sieci [15]. Przy zamianie dźwięku na spektrogram melowy, bardzo łatwo stworzyć bazę danych obrazów.

CNN, dzięki warstwom konwolucyjnym, jest w stanie dostrzegać cechy charakterystyczne obrazu. Każdy obraz może zostać przedstawiony jako macierz liczbowych wartości kolorów RGB. Ekstrakcja cech charakterystycznych może być przedstawiona w prosty sposób jako translacja macierzy kolorów, w celu znalezienia schematów oraz kompresji obrazu.



Rys 2.11 – Przykładowe działanie warstwy konwolucyjnej wyodrębniającej cechy charakterystyczne obrazu

Dzięki takiej analizie dwuwymiarowego obrazu model jest w stanie doprowadzić obraz do analizy wagowej wartości w końcowej macierzy tak, jak w przypadku zwykłych sieci neuronowych.

Postanowiono wprowadzić optymalizację polegająca na używaniu spektrogramach w skali szarości, ponieważ model analizuje tylko jeden znormalizowany parametr zamiast trzech. Przez fakt badania tylko jednego dźwięku bazowego o częstotliwości 440Hz, jest to optymalne rozwiązanie, ponieważ w większości przypadków ulegać zmianie będą ilości oraz długości harmonicznych. Korelację tą przedstawiono w podrozdziale 3.1.2. Do wizualizacji modelu stworzonego na potrzeby projektu posłużył program o nazwie „Netron”.

2.3. Język programowania python

### 2.3.1. Charakterystyka języka

Język programowania python cechuje się możliwością stosowania go w wielu przypadkach. Od pisania aplikacji do operowania na danych. Cechuje się bardzo duża elastycznością, a dzięki jego popularności, zostało stworzone wiele rozszerzających jego działanie bibliotek [18]. Został wybrany jako język do napisania oprogramowania dlatego, że dzięki tym właśnie bibliotekom istnieje możliwość stworzenia funkcji analizującej sygnał i tworzącej spektrogramy, obsługującej model sieci neuronowej oraz prosty interfejs graficzny w granicach mono językowych skryptów. Co prawda istnieje możliwość programowania w nim obiektowo, skupiono się jednak na integracji wszystkich funkcjonalności bazując w większości na proceduralnym charakterze wykonywania instrukcji.

Co prawda język programowania python nie jest najszybszym z dostępnych alternatyw, dzięki przystępności jest bardzo dobrą bazą do stworzenia prototypu. Pozwala to na skupieniu się na funkcjonalności urządzenia, pomijając na czas obecny kwestie optymalizacyjne.

W poniższym rozdziale zostały krótko opisane wykorzystane biblioteki, oraz ich funkcjonalności. Dokładny opis użytych funkcji i metod wraz z kodem został zamieszczony w podrozdziale 3.3.

### 2.3.2. Wykorzystane biblioteki

Poniżej zostały opisane zewnętrzne biblioteki, dzięki którym możliwe było stworzenie aplikacji integrujące wszystkie potrzebne funkcjonalności. Moduły i wbudowane opcje zostały pominięte.

1. Kivy.

2. NumPy.

3. Librosa.

4. Tensorflow.

5. MatPlotLib.

6. Pillow.

1. Kivy – Framework umożliwiający tworzenia interfejsu graficznego dla urządzeń dotykowych [19]. Został on wybrany dzięki możliwości w miarę prostego modyfikowania właściwości interfejsu, takich jak rozdzielczość aplikacji co było ważnym elementem, biorąc pod uwagę zaimplementowania w urządzeniu ekranu o niestandardowej rozdzielczości 480 na 320 pikseli. Wykorzystano także opcje okien informacyjnych w celu zaimplementowania stworzonych w programie Blender animacji ekranu powitalnego oraz animowanej instrukcji korzystania z urządzenia.

2. NumPy – Jest to biblioteka pozwalająca na operacje macierzowe oraz implementację skomplikowanych funkcji matematycznych [20]. Nie tylko to rozszerza możliwości analizowania dźwięku, to pozwala także na optymalizację procesów, poprzez obliczanie za pomocą tej biblioteki. Dzięki wbudowanym funkcjom, możliwe są operacje matematyczne ,takie jak wybieranie argumentu maksymalnego z danego zbioru liczbowego, co posłuży do klaryfikacji wyniku uzyskanego przez model sieci.

3. Librosa – Biblioteka stworzona w celu analizy dźwięku [21]. Jest to najważniejszy element aplikacji, ponieważ dzięki funkcjom dostępnym w zakresie librosy możliwa jest analiza sygnału dźwiękowego za pomocą STFT. Tworzenie spektrogramów melowych również znajduje się w dziedzinie tej biblioteki, co skutkuje mniejszą procedurą obróbki danych.

4. Tensorflow – Jedna z najpopularniejszych bibliotek pozwalająca na tworzenie modeli uczenia maszynowego [22]. Z powodu użytkowania mniejszego modelu operującego na skalach szarości o rozszerzeniu „tflite” dominującego na mikrokontrolerach, do operowania na funkcjach obsługujących działanie modelu nie potrzebna jest dodatkowa biblioteka, taka jak na przykład „Keras”.

5. MatPlotLib – Biblioteka umożliwiająca tworzenie wykresów [23]. Jest to równie ważny element jak Librosa, ponieważ pozwala na stworzenie graficznej reprezentacji danych, w celu stworzenia zbioru treningowego oraz spektrogramu analizowanego brzmienia. Co więcej, używany jest również spektrogram melowy wyświetlany w interfejsie aplikacji, pozwalający zaobserwować właściwości szeregu harmonicznego.

6. Pillow – Python Image Library, biblioteka służąca do przeprowadzania operacji na obrazach [24]. Dzięki jej zastosowaniu możliwe jest dynamiczne dodawanie i zmiana wyświetlanych obrazów w interfejsie stworzonej aplikacji.

## 2.4. Urządzenia embedded

### 2.4.1. Ogólna charakterystyka

Urządzenia embedded, a dokładniej systemy embedded, są tą specyficzne systemy stworzone do działania w pewnym określonym zakresie lub dziedzinie [25]. Jednym z przykładów takich urządzeń jest kalkulator, którego funkcja ogranicza się tylko do przeprowadzania operacji matematycznych. Dziedzina embedded, wraz z rozwojem technologii informacyjnej, również ulega zmianom. W dniach dzisiejszych takie miano mogą uzyskać urządzenia bazowane na telefonach dotykowych, będące o wiele bardziej skomplikowane oraz posiadające większy potencjał funkcjonalności, niż np. wcześniej wspomniany kalkulator. Dzięki temu ten projekt magisterski może zostać zaliczony do tej kategorii, mimo iż został stworzony bazując na jednopłytkowym minikomputerze, a nie na mikrokontrolerze. W tym przypadku przedstawiono możliwości takich urządzeń w tej dziedzinie, nie ograniczając urządzenia do wykonywania wyłącznie operacji na prototypowej aplikacji. Dzięki temu możliwe jest rozszerzenie zastosowania urządzenia do większej ilości analizy dźwięku, tworząc dodatkowe programy oraz modele.

### 2.4.2. RaspberryPi

RaspberryPi jest to nazwa fundacji, odpowiedzialnej za tworzenie małych, jednopłytkowych komputerów [26]. Dzięki temu że posiadają większa moc obliczeniową w porównaniu do mikrokontrolerów, oraz bazują na linuksowym systemie operacyjnym, pozwalają na bardzo wszechstronne zastosowanie, a ich niewielki rozmiar pozwala na tworzenie przenośnych urządzeń. Wyżej wspomniana większa moc obliczeniowa w zestawieniu z mikrokontrolerami, pozwala na używanie pythona w celu tworzenia aplikacji. Przez charakterystykę optymalizacyjną tego języka, używanie go na mikrokontrolerach skutkowałoby o wiele dłuższym czasem operacji matematycznych oraz funkcjonowania modelu. Tak jak wspomniano we wcześniejszym podrozdziale, urządzenie oparte o RaspberryPi nie podlega tradycyjnej definicji systemu embedded ze względu na swoją elastyczność i moc obliczeniową, ale pozwala na stworzenie mobilnych i małych urządzeń będących w stanie dokonywać więcej trudniejszych operacji obliczeniowych niż mikrokontrolery.

Używanym modelem jednopłytkowego komputera jest RaspberryPi 4B posiadająca 4 gigabajtów pamięci RAM. Używany system operacyjny to PiOS bazujący na systemie Debian 12 „bookworm”.

# 3. PRZEBIEG BADANIA I METODOLOGIA

Gdyby podczas badań bądź użytku pojawiła się inna częstotliwość niż badana postanowiono dodać trzecia kategorię brzmienia pod nazwą „niekreślone”. Po zebraniu danych dźwiękowych zamieniono je na dane liczbowe używając STFT, a następnie sporządzono z nich MEL-spektrogramy. Po zauważeniu różnic w występujących częstotliwościach harmonicznych oraz ich intensywnościach, zebrano próbki dźwiękowe z pianina cyfrowego dla każdego z brzmień, przygotowano model konwolucyjnej sieci neuronowej a następnie wytrenowano go przy użyciu MEL-spektrogramów, w ilości 240 plików dla każdego z przypadków. Dla przygotowanego modelu w języku Python została napisana aplikacja analizująca dźwięk instrumentu klawiszowego, informująca o predykcji klasy oraz pewności modelu podana w procentach. Na końcu stworzono oddzielny model posługując danymi zebranymi na akustycznych instrumentach po 843 na każdą z klas. Została również zaimplementowana graficzna pomoc w obsłudze oraz wyświetlanie MEL-spektrogramu próbki. Jako urządzenie obliczeniowe, które pozwoliłoby na budowę przenośnego urządzenia o możliwości wyświetlania graficznych funkcjonalności aplikacji, wykorzystano jednopłytkowy komputer RaspberryPi model 4B.

## 3.1. Dane

Dane na których przeprowadzono analizę oraz które są baza operowania modelu to czterosekundowe fragmenty dźwięku klawisza A razkreślnego. Z danych tych zostały stworzone spektrogramy melowe, a następnie zostały one przeanalizowane pod kątem występujących korelacji oraz przygotowane jako zbiór treningowy. Zebrana próbka dźwiękowa została skryptowo podzielona na czterosekundowe fragmenty (końcowy fragment mniejszy niż ta długość został usunięty). Kod skryptu dzielącego dane oraz tworzącego spektrogramy melowe w celu aplikacji w modelu (przedstawiono i opisano tylko ten, ponieważ dla większych rozdzielczości oraz koloru zmianie ulegają jedynie parametry funkcji) został przedstawiony poniżej. Opis został zamieszczony w formie komentarzy kodu (zawarte między ''' na górze oraz dole opisu) zaznaczone kolorem zielonym).

Kod skryptu odpowiedzialnego za rozdzielenie nagrania na poszczególne próbki:

import os  
import librosa  
import soundfile as sf  
  
def split\_audio(file\_path, sample\_duration=4.0, output\_folder="WAV/bright"):  
 audio, sr = librosa.load(file\_path, sr=None)  
 samples\_per\_segment = int(sample\_duration \* sr)  
 os.makedirs(output\_folder, exist\_ok=True)  
 total\_samples = len(audio)  
 num\_segments = total\_samples // samples\_per\_segment  
 print(f"Rozbijam plik na {num\_segments} próbek po {sample\_duration}

sekundy")

for i in range(num\_segments):  
 start\_sample = i \* samples\_per\_segment  
 end\_sample = start\_sample + samples\_per\_segment  
 segment = audio[start\_sample:end\_sample]  
 output\_filename = os.path.join(output\_folder, f"sample\_{i + 1}.wav")  
 sf.write(output\_filename, segment, sr)  
 print(f"Zapisano próbkę: {output\_filename}")  
  
 if total\_samples % samples\_per\_segment != 0:  
 remainder\_segment = audio[num\_segments \* samples\_per\_segment:]  
 output\_filename = os.path.join(output\_folder, f"sample\_{num\_segments + 1}.wav")  
 sf.write(output\_filename, remainder\_segment, sr)  
 print(f"Zapisano ostatnią próbkę (mniejsza niż 4 sekundy): {output\_filename}")

|  |
| --- |
| '''  Podział pliku audio na segmenty i zapis ich jako osobne pliki WAV.  Funkcja iteruje przez nagranie audio, dzieląc je na segmenty o określonej długości i zapisuje je jako osobne pliki w odpowiadającym folderzer. Jeśli pozostaje niepełny segment, zapisuje go osobno.  Parametry i zmienne:  - num\_segments: liczba pełnych segmentów audio  - samples\_per\_segment: liczba próbek w jednym segmencie  - start\_sample, end\_sample: indeksy początkowy i końcowy segmentu  - segment: wycięty fragment audio  - output\_folder: folder do zapisu plików  - output\_filename: nazwa pliku wynikowego  - sr: częstotliwość próbkowania audio  ''' |

audio\_file = 'Bright.wav'  
split\_audio(audio\_file, sample\_duration=4.0, output\_folder="WAV/bright")

|  |
| --- |
| '''  Podczas operowania manualnie zmieniono nazwy badanych brzmień w danych fragmentach skryptu.  ''' |

Skrypt tworzenia spektrogramów:

import os  
import librosa  
import numpy as np  
import matplotlib.pyplot as plt  
from PIL import Image  
  
source\_audio\_folder = "WAV"  
mel\_output\_folder = "Train\_data/96Mels"  
  
soft\_folder = os.path.join(mel\_output\_folder, "soft")  
bright\_folder = os.path.join(mel\_output\_folder, "bright")  
generic\_folder = os.path.join(mel\_output\_folder, "generic")  
  
os.makedirs(soft\_folder, exist\_ok=True)  
os.makedirs(bright\_folder, exist\_ok=True)  
os.makedirs(generic\_folder, exist\_ok=True)

|  |
| --- |
| ''' Powyższy fragment kodu odpowiada za stworzenie odpowiednich podfolderów oraz sprecyzowanie lokalizację rozdzielonych danych dźwiękowych. Foldery podzielonych próbek zostały stworzone za pomocą wyżej opisanego skryptu, zachowując również konwencje nazywnictwa. ''' |

def save\_mel\_spectrogram(audio\_file, output\_path):  
 y, sr = librosa.load(audio\_file, sr=None)  
 mel\_spectrogram = librosa.feature.melspectrogram(y=y, sr=sr, n\_mels=96)  
  
 mel\_spectrogram\_db = librosa.power\_to\_db(mel\_spectrogram, ref=np.max)  
  
 mel\_spectrogram\_db\_flipped = np.flipud(mel\_spectrogram\_db)  
  
 mel\_min = mel\_spectrogram\_db\_flipped.min()  
 mel\_max = mel\_spectrogram\_db\_flipped.max()  
 mel\_normalized = 255 \* (mel\_spectrogram\_db\_flipped - mel\_min) / (mel\_max - mel\_min)  
 mel\_normalized = mel\_normalized.astype(np.uint8)  
  
 plt.imsave(output\_path, mel\_normalized, cmap='gray', dpi=96)  
  
 img = Image.open(output\_path).resize((96, 96))  
 img.save(output\_path)

|  |
| --- |
| ''' |
| Aby uzyskac obraz o większej rozdzielczości należy zmienić liczbę dpi oraz w „resize” na odpowiadający rozmiar. W celu posługiwania się inna skalą niż szarości, należy zmienić tez parametr „cmap”. Kolorowe spektorgramy melowe korzystają z mapy „viridis”. |
| ''' |

label\_mapping = {'bright': bright\_folder, 'soft': soft\_folder, 'generic': generic\_folder}  
  
for label in ['bright', 'soft', 'generic']:  
 source\_folder = os.path.join(source\_audio\_folder, label)  
  
 if os.path.exists(source\_folder):  
 for audio\_file in os.listdir(source\_folder):  
 if audio\_file.endswith(".wav"):  
 audio\_path = os.path.join(source\_folder, audio\_file)  
 filename = os.path.splitext(audio\_file)[0]  
  
 mel\_output\_path = os.path.join(label\_mapping[label], f"{filename}\_mel.png")  
 save\_mel\_spectrogram(audio\_path, mel\_output\_path)  
  
 print(f"Processed {audio\_file}: Mel-spectrogram saved.")  
  
print("Processing completed. Mel-spectrograms saved as 96x96 PNG images.")

|  |
| --- |
| '''Funkcja przekształca plik audio na spektrogram melowy; analizuje sygnał za pomocą STFT zawartej w bibliotece librosa oraz konwertuje go do skali decybelowej, normalizuje i zapisuje jako obraz PNG o rozdzielczości 96x96 pikseli w skali szarości.Parametry i zmienne:- audio\_file: ścieżka do pliku audio- output\_path: ścieżka zapisu obrazu spektrogramu melowego- y, sr: dane audio oraz częstotliwość próbkowania- mel\_spectrogram: mel-spektrogram utworzony z sygnału audio- mel\_spectrogram\_db: spektrogram w skali decybelowej- mel\_normalized: znormalizowany obraz spektrogramu w skali 0-255- label\_mapping: mapowanie etykiet na foldery wyjściowe- source\_folder: folder zawierający pliki audio dla danej etykiety- mel\_output\_path: ścieżka zapisu wygenerowanego obrazu mel-spektrogramu''' |

### 3.1.1. Pozyskiwanie danych

### 3.1.2. Analiza

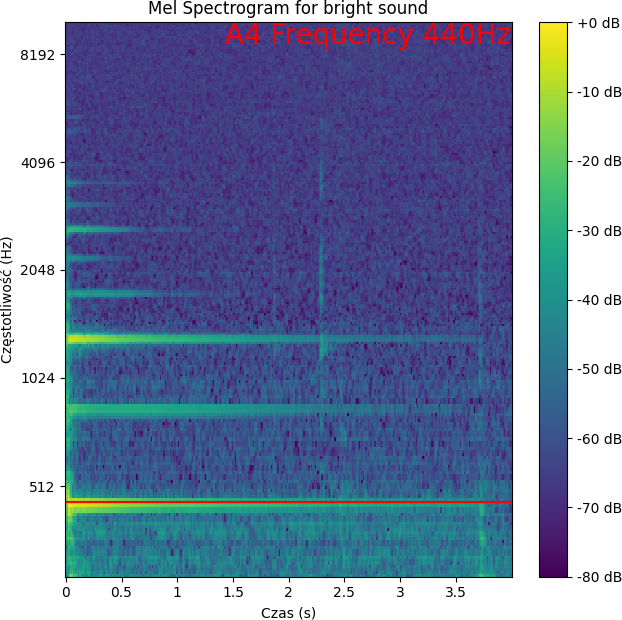
Analiza dźwięku polegała na znalezieniu różnic w cechach występujących częstotliwości. Dokonano tego analizując dane wstępne, zebrane z instrumentu cyfrowego. Zestawiono ze sobą wykresy i porównano ich cechy charakterystyczne. Poniżej przedstawiono te spektrogramy melowe dla trzech klas.

|  |  |  |
| --- | --- | --- |
|  |  |  |

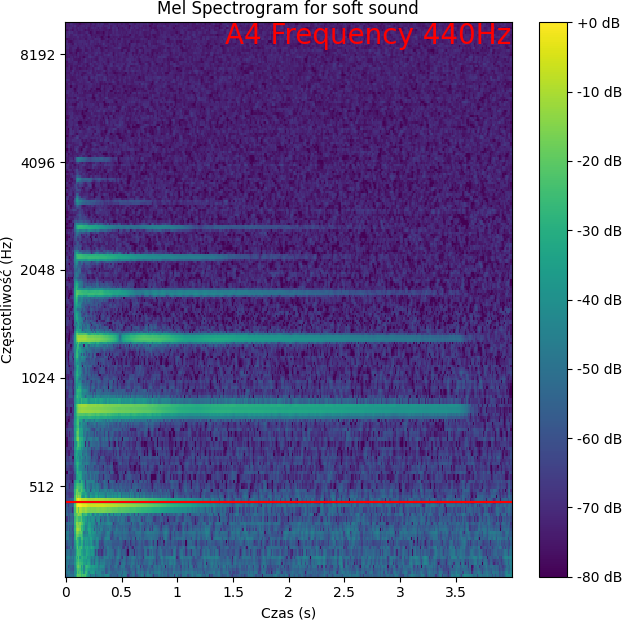
|  |  |  |
| --- | --- | --- |
| *a)* | *b)* | *c)* |

Rys 2.9 – zestawione spektrogramy melowe dla a) brzmienia jasnego, b) brzmienia miękkiego oraz c) niesklasyfikowanego z zaznaczoną częstotliwością badanego dźwięku

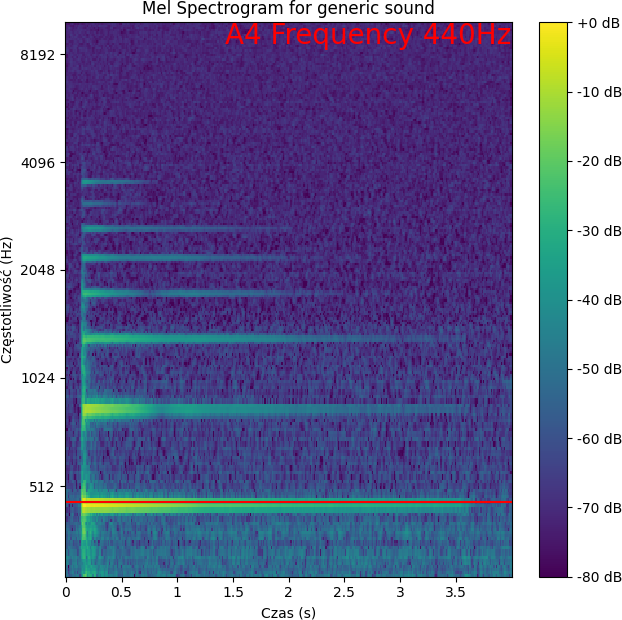
Widać nieznaczne różnice pomiędzy ilością występujących składowych harmonicznych. Co prawda, intensywności są porównywalne, ale długość trwania tych składowych również wykazuje zróżnicowanie. W przypadku jasnego brzmienia na spektrogramie poniżej (rysunek o numerze *2.10*) można zauważyć, że częstotliwość bazowa (440 Hz) jest obecna podczas trwania całej próbki dźwięku. Bardzo ważnym elementem jest też obecność wyższych harmonicznych, szczególnie tych przekraczających częstotliwość 4096 Hz. Mają one porównywalnie mniejsze intensywności, ale porównując je z rys *2.11* oraz *2.12* są one widoczne. Zliczono jedenaście występujących składowych harmonicznych. Liczba ta prawdopodobnie zwiększyłaby się, posiadając bardziej precyzyjny sprzęt pomiarowy oraz odizolowane środowisko pomiarowe, w którym nie występowałyby szumy. Porównując z kolei spektrogram melowy brzmienia miękkiego (rysunek numer *2.11*) widzimy znaczą różnicę. Cechującą własnością jest bardzo krótkie brzmienie bazowej częstotliwości. Porównując ją z brzmieniem jasnym (*Rys. 2.10*) oraz niesklasyfikowanym (*Rys. 2.12*), widzimy dłuższe oraz intensywniejsze harmoniczne wyższego rzędu, kończące się na częstotliwości 4400 Hz. Zliczono dziewięć występujących harmonicznych. Z kolei brzmienie niesklasyfikowane (*Rys. 2.12*) charakteryzuje się, tak jak w przypadku jasnego, pełną długością brzmienia tonu podstawowego, oraz występowaniem porównywalnych z brzmieniem miękkim harmonicznych, lecz brakująca jest występująca powyżej 4090 Hz Ich widoczna ilość to osiem.



Rys 2.10 – spektrogram melowy dla brzmienia jasnego



Rys 2.11 – spektrogram melowy dla brzmienia miękkiego

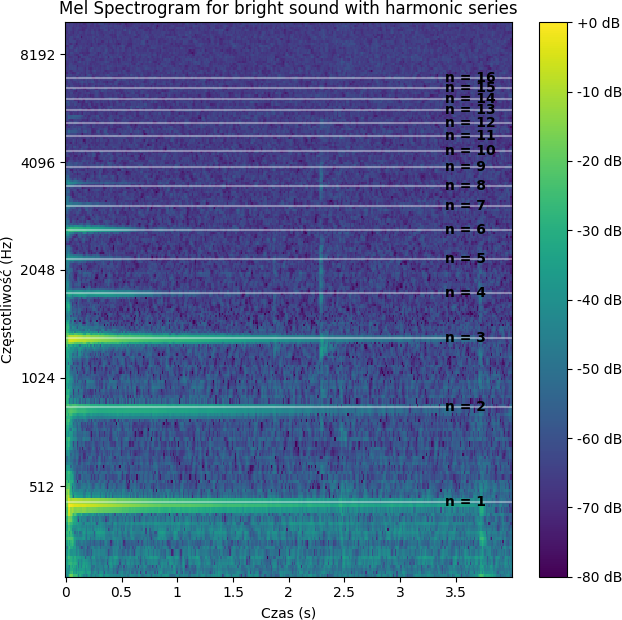


Rys 2.12 – spektrogram melowy dla brzmienia niesklasyfikowanego (pośredniego)

Przeanalizowano również rzędy występujących harmonicznych. Na spektrogramach uwydatniono częstotliwości dla harmonicznych występujących w przedziale częstotliwości (440Hz – 7040Hz). Wartości tych częstotliwości wyliczono korzystając z analogii wynikającej z zależności występującej we wzorze (2.2). Dzięki zależności pomiędzy sinusoidą a rzędem harmonicznej można częstotliwości te wyliczyć z prostego wzoru:

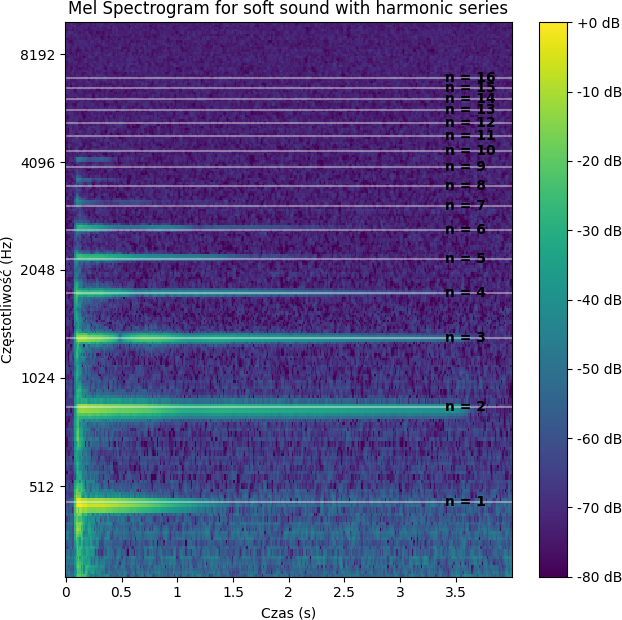
|  |  |
| --- | --- |
|  | (2.6) |

Gdzie to rząd harmonicznej, to częstotliwość harmonicznej dla rzędu n, a to częstotliwość bazowa. Wynik tej analizy zaprezentowania poniżej, wraz z podsumowywującym opisem. Białymi liniami zaznaczono częstotliwości, a etykiety opisujące rząd – czarnym kolorem (*Rys. 2.13 – 2.15*).



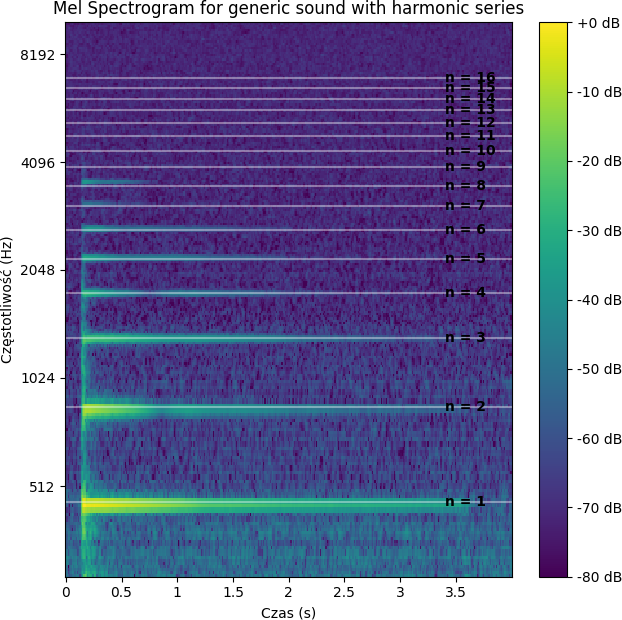
Rys 2.13 – spektrogram melowy dla brzmienia jasnego z zaznaczonymi częstotliwościami harmonicznymi

Na powyższym wykresie można zaobserwować to, że najwyższą harmoniczną jest ta rzędu . Można zaobserwować też ledwo widoczną harmoniczną rzędu .



Rys 2.14 – spektrogram melowy dla brzmienia miękkiego z zaznaczonymi częstotliwościami harmonicznymi

W przypadku tego brzmienia, najwyżej występującą harmoniczną opisuje rząd .



Rys 2.15 – spektrogram melowy dla brzmienia nieokreślonego z zaznaczonymi częstotliwościami harmonicznymi

Najwyższą harmoniczną jest harmoniczną dla .

Dzięki danym które zostały pozyskane poprzez graficzną analizę spektrogramów oraz dokładniejsze określenie występujących na nich elementach, stwierdzono że różnice pomiędzy tymi brzmieniami są skorelowane z ilością oraz intensywnościami składowych szeregu harmonicznego w następujący sposób:

– Gdy ton podstawowy występuje porównywalnie długo z harmonicznymi rzędu oraz , a pozostałe harmoniczne są liczne (występują powyżej rzędu ) i znacznie krócej trwają od tych drugiego oraz trzeciego rzędu, to uzyskana barwa dźwięku może być odebrana oraz skategoryzowana jako jasna.

– Gdy ton podstawowy występuje krótko w porównaniu z harmonicznymi rzędu oraz , a pozostałe harmoniczne są stopniowo coraz krótsze w odniesieniu do tych rzędu drugiego oraz trzeciego, to uzyskana barwa dźwięku może zostać odebrana oraz skategoryzowana jako miękka

– Gdy ton podstawowy występuje porównywalnie długo z harmonicznymi rzędu oraz , a pozostałe harmoniczne stopniowo zmniejszają swoją długość, to odebrana barwa dźwięku może zostać skategoryzowana jako nieokreślona, znajdująca się pomiędzy dwoma wcześniejszymi.

Na spektrogramach zauważyć można przesunięcie fazowe harmonicznych składowych i ich wartości teoretycznych. Jest to prawdopodobnie wynikiem przybliżenia częstotliwości do liczb całkowitych, jak i również czynnikami zewnętrznymi. Przykładem może być minimalne tłumienie fal poprzez ich propagację, bądź charakterystyka aparatury służącej do odczytywania częstotliwości (w tym przypadku – mikrofonu) co oznaczałoby przesunięcia ich wartości od początkowych.

Dzięki występującej korelacji pomiędzy różnicami w specyfikacji harmonicznych możliwe jest stworzenie modelu który będzie w stanie rozpoznawać brzmienia dzięki analizie tych cech. W celu zwiększenia precyzji i wszechstronności użytkowej modelu, powiększono dane treningowe o zbiory próbek pochodzących od instrumentów akustycznych. Po względnym porównaniu spektrogramów zaobserwowano podobna korelację z bardziej uwydatnionymi harmonicznymi wyższego rzędu. Widoczne są inne zależności, więc należy przygotować osobne modele do rozpoznawania brzmień instrumentów cyfrowych oraz akustycznych. Porównanie zaprezentowano na obrazkach poniżej:

### 

### 3.1.3. Przygotowanie zbiorów treningowych

## 3.2. Stworzenie modelu sieci neuronowej

## 3.3. Aplikacja do klasyfikacji dźwięku

### 3.3.1. Struktura i działanie aplikacji

### 3.3.2. Elementy graficzne

### 3.3.3. Implementacja i funkcjonalność modelu sieci

# 4. PODSUMOWANIE

W projekcie nie skupiono się na badaniu cech innych brzmień w kategorii „generic”, ponieważ badania takie wymagałyby pozyskanie bardzo dużego zbioru danych, a sama analiza musiałaby dotyczyć wszystkich szczegółów. Stworzenie modelu oraz urządzenia byłoby wtedy narażone na niepowodzenie, ponieważ w celu dokładnego wytrenowania modelu oraz jego predykcji, która znajdowałaby się w satysfakcjonującej pewności, należałoby używać mikrofonów bądź urządzeń odczytujących częstotliwości o bardzo małym błędzie pomiarowym i wysokiej precyzji, ponieważ różnice mogłyby dotyczyć bardzo małych różnic w długości składowych szeregu harmonicznego. Całe urządzenie byłoby bardzo narażone na szum, a model podatny na anomalie z niego wynikające. Przy założonej kategoryzacji pomyślnie udało się stworzyć urządzenie potrafiące te brzmienia rozróżnić za pomocą modelu konwolucyjnej sieci neuronowej. W celu rozwinięcia projektu adekwatnym byłoby zaopatrzenie urządzenia w wyższej jakości urządzenia do zbierania dźwięku oraz pozyskanie o wiele większej ilości danych, włączając także inne brzmienia.

# WYKAZ LITERATURY

[1] Casati R., Dokic J., Sounds, Stanford Encyclopedia of Philosophy, CSLI, Stanford University.

[2] Fundamentals of Telephone Communication Systems. Western Electrical Company.

[3] https://www.if.pw.edu.pl/~anadam/WykLadyFO/FoWWW\_13a.html [04.03.2025]

[4] https://home.agh.edu.pl/~kakol/efizyka/w13/main13e.html [04.03.2025]

[5] https://zpe.gov.pl/a/przeczytaj/DdtkgmBMy [04.03.2025]

[6] Min Xu, Ling - Yu Duan, Jianfei Cai, Liang-Tien Chia, Changsheng Xu2 and Qi Tian, „HMM-Based Audio Keyword Generation”

[7] Fang Zheng, Guoliang Zhang and Zhanjiang Song (2001), „Comparison of Different Implementations of MFCC”

[8] Reflecting on Echoes and the Cepstrum: A Look at Quefrency Alanysis and Hearing [11.03.2025]

[9] APD Politechniki Warszawskiej - MFCC: [11.03.2025] https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://p ages.mini.pw.edu.pl/~rafalkoj/www/%3Fdownload%3DAPD%2520- %2520MFCC.pdf&ved=2ahUKEwiXxIekt4KMAxXqPRAIHRCoH2wQFnoECBgQAQ& usg=AOvVaw1FpNVb78NoHxvCEKNCJDCG

[10] Grzegorz Graczyk, Andrzej Stasiak, „Analiza częstotliwości podstawowej dźwięku”, Serwis Wayback machine[11.03.2025]

[11] Artur Zacniewski, „Analiza porównawcza wybranych transformat

w kontekście zobrazowania zaszumionego sygnału harmonicznego”

[12] Centrum pomocy MatLab [17.03.2025]

[13] Dokumentacja biblioteki librosa [21.03.2025]

[14] Roheen Qamar, Baqar Ali Zardari, „Artificial Neural Networks: An Overview”

[15] Yang Ding, Hongzheng Zhang, Wanmacairang Huang, Xiaoxiong Zhou,

Zhihan Shi, „Efficient Music Genre Recognition Using ECAS-CNN: A Novel Channel-Aware Neural Network Architecture”

[16] MathWorks – What Is a Convolutionel Neural Network? [24.03.2025]

[17] Sakib Mostafa , Fang-Xiang Wu, Neural Engineering Techniques for Autism Spectrum Disorder Chapter 3 – „Diagnosis of autism spectrum disorder with convolutional autoencoder and structural MRI images”

[18] „About python” na oficjalnej stronie pythona [27.03.2025]

[19] Dokumentacja frameworku Kivy [27.03.2025]

[20] Dokumentacja biblioteki NumPy [27.03.2025]

[21] Dokumentacja biblioteki Librosa [27.03.2025]

[22] Dokumentacja biblioteki TensorFlow [27.03.2025]

[23] Dokumentacja biblioteki MatPlotLib [27.03.2025]

[24] Dokumentacja biblioteki Pillow [27.03.2025]

[25] https://www.techtarget.com/whatis/definition/embedded-device [27.03.2025]

[26] https://www.raspberrypi.com/ [27.03.2025]

# WYKAZ RYSUNKÓW

Rys 2.1 – http://www.strojenie-pianin.pl/index.php?strona=historia [04.03.2025]

Rys 2.2 – materiały własne

Rys 2.3 – materiały własne

Rys 2.4 – https://pl.yamaha.com/pl/products/musical\_instruments/pianos/arius/ydp-

143/index.html [04.03.2025]

Rys 2.5 do 2.9 – materiały własne

Rys 2.10 – Autorstwa Dake, Mysid - Vectorized by Mysid in CorelDraw on an image by Dake https://commons.wikimedia.org/w/index.php?curid=1412126

Rys 2.11 – Sakib Mostafa , Fang-Xiang Wu, Neural Engineering Techniques for Autism Spectrum Disorder Chapter 3 – „Diagnosis of autism spectrum disorder with convolutional autoencoder and structural MRI images”

# MATERIAŁY DODATKOWE

Repozytorium projektu: https://github.com/Kiryl24/Master-thesis-project