



UNIWERSYTET
IM. ADAMA MICKIEWICZA
W POZNANIU

Wydział Matematyki i Informatyki

Robert Tarnas
Numer albumu: 402178

Sztuczne Sieci Neuronowe w kompozycji algorytmicznej
My thesis

Praca magisterska na kierunku **Analiza i Przetwarzanie Danych**
(specjalność: none)
napisana pod opieką
prof. UAM dr hab. Maciej Grześkowiak

Poznań, grudzień 2020

Poznań, 31 stycznia 2022 r.

Oświadczenie

Ja, niżej podpisany **Robert Tarnas**, student Wydziału Matematyki i Informatyki Uniwersytetu im. Adama Mickiewicza w Poznaniu oświadczam, że przedkładaną pracę dyplomową pt. *Sztuczne Sieci Neuronowe w kompozycji algorytmicznej* napisałem samodzielnie. Oznacza to, że przy pisaniu pracy, poza niezbędnymi konsultacjami, nie korzystałem z pomocy innych osób, a w szczególności nie zlecałem opracowania rozprawy lub jej części innym osobom, ani nie odpisywałem tej rozprawy lub jej części od innych osób. Oświadczam również, że egzemplarz pracy dyplomowej w wersji drukowanej jest całkowicie zgodny z egzemplarzem pracy dyplomowej w wersji elektronicznej. Jednocześnie przyjmuję do wiadomości, że przypisanie sobie, w pracy dyplomowej, autorstwa istotnego fragmentu lub innych elementów cudzego utworu lub ustalenia naukowego stanowi podstawę stwierdzenia nieważności postępowania w sprawie nadania tytułu zawodowego.

[TAK] wyrażam zgodę na udostępnianie mojej pracy w czytelni Archiwum UAM

[TAK] wyrażam zgodę na udostępnianie mojej pracy w zakresie koniecznym do ochrony mojego prawa do autorstwa lub praw osób trzecich

Streszczenie

Streszczenie po polsku (dla niektórych szablonów).

Słowa kluczowe: klasa

Abstract

Sample abstract. Sample abstract.

Keywords: klasa

Tu możesz umieścić swoją dedykację.

Spis treści

Rozdział 1. Elementarne wiadomości o sieciach neuronowych	9
1.1. Poznanie a rozpoznanie	13
1.2. Model sztucznego neuronu	14
1.3. Cel pracy	15
Rozdział 2. Podstawy aparatu matematycznego algebry liniowej	17
2.1. Grupa abelowa	17
2.2. Przestrzeń wektorowa	18
2.3. Iloczyn skalarny i przestrzeń unitarna	19
2.4. Proste zadanie klasyfikacji	20
Rozdział 3. Podstawy teorii muzyki i formatu MIDI	23
3.1. Podstawy teorii muzyki	23
3.2. Format MIDI	28
Rozdział 4. Reprezentacja i klasyfikacja utworu muzycznego	32
4.1. Zadanie klasyfikacji dźwięków	33
4.1.1. Klasyfikacja na podstawie podobieństwa dźwięków	34
4.1.2. Modelowanie binarne typu cisza - dźwięk	36
4.2. Procedura testowa	37
4.2.1. (A) Porównywanie utworu wzorcowego z innym utworem	38
4.2.2. (B) Porównywanie utworu wzorcowego z utworem losowym	39
4.3. Wnioski	41
Rozdział 5. Przekształcenia geometryczne w przestrzeni	42
5.1. Definicja macierzy	42
5.2. Iloczyn macierzy	43
5.2.1. Macierz obrotu	44
5.3. Obroty w \mathbb{R}^2	45
5.4. Obroty w \mathbb{R}^3	47
Rozdział 6. Obroty wektorów nut w przestrzeni trójwymiarowej	49
6.1. Obroty akordów w \mathbb{R}^3	49
6.2. Implementacja	50
6.3. Wyniki	52

6.4. Wnioski	52
6.5. Obrót utworu muzycznego w \mathbb{R}^3	52
6.6. Wnioski	54

WSTĘP

Sztuczne sieci neuronowe, to temat szeroko dyskutowany w literaturze naukowej. Podejmowali go różni badacze ludzkiej inteligencji, chcąc na różne sposoby zamodelować w maszynie jakiś aspekt ogólnej inteligencji i/lub działalności twórczej człowieka. Praca ta ma za zadanie poszerzyć granice naszego poznania w dziedzinie sztucznej inteligencji, podejmując temat wykorzystania metod uczenia uczenia maszynowego do realizacji zadania rozpoznawania i tworzenia muzyki.

Praca podzielona jest na sześć rozdziałów. pierwszy opisuje krótko historie i stan badań w dziedzinie sieci neuronowych. Rozdział drugi i trzeci wprowadzają niezbędny aparat pojęciowy wykorzystywany w dalszej części rozważań. Rozdziały: czwarty, piąty oraz szósty prezentują autorski system rozpoznawania i generowania muzyki, który poddany analizie skłania do pewnych przemyśleń na temat możliwości jakie oferuje sztuczna inteligencja, które sformułowane zostały w zakończeniu.

ROZDZIAŁ 1

Elementarne wiadomości o sieciach neuronowych

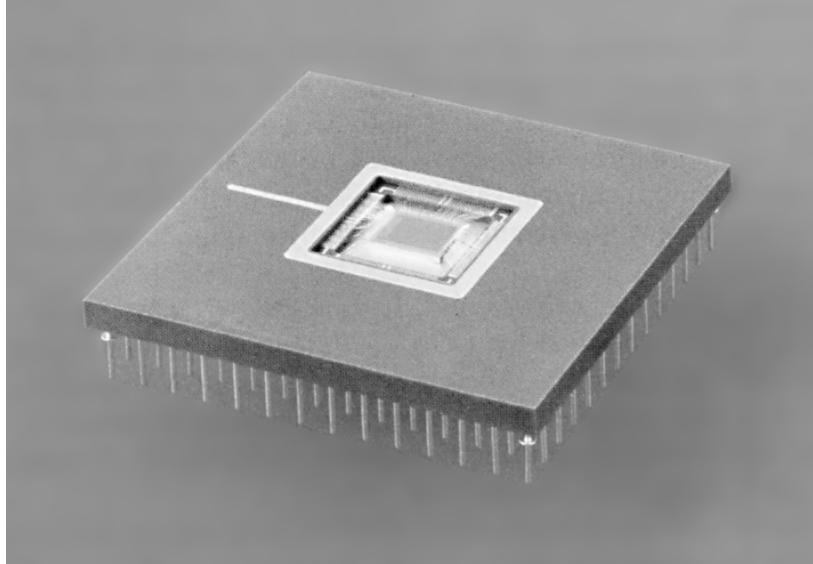
W ciągu ostatnich dekad obserwuje się rozwój badań nad sztucznymi sieciami neuronowymi. Badania te są prowadzone przez naukowców z różnych dziedzin, a ich rozwój jest motywowany osiągnięciami w zakresie badania mózgu człowieka. Obecnie odkryto wiele tajemnic działania mózgu. Wiadomo na przykład jaka jest budowa komórek nerwowych (neuronów) albo w jaki sposób wymieniają one informacje między sobą [1, s. 13].

Pomimo długich badań fundamentalne pytania o samą istotę działania mózgu nadal pozostają w mocy. Ciągłe nie potrafimy jednoznacznie ustalić czym jest myśl ludzka albo rozwiązać zagadkę istnienia świadomości. Fascynujące są dla nas także możliwości mózgu – jego niezwykła zdolność do nauki i adaptacji do otoczenia oraz umiejętność myślenia kreatywnego i tworzenia nowych rzeczy [1, s. 14].

Z inżynierskiego punktu widzenia stworzenie inteligentnych układów powinno się opłacić, ponieważ maszyna byłaby zdolna wyręczyć człowieka w wykonywaniu różnych skomplikowanych prac. Powstanie *sztucznych sieci neuronowych* było krokiem naprzód w realizacji tego przedsięwzięcia. Sieci takie mają niektóre właściwości mózgu człowieka. Posiadają zarówno zdolność do generalizacji (czyli do uogólniania nabytej wiedzy na nowe, nieznane przypadki), oraz zdolność do rozwiązywania zadań nieprecyzyjnie zdefiniowanych formalnie. Sieci takie mogą działać nawet przy pewnym poziomie uszkodzeń. Mają także stosunkowo dużą prędkość przetwarzania informacji [1, s. 15-16].

Sztuczne sieci neuronowe mogą być realizowane na dwa sposoby. W postaci sprzętowej sieć jest specjalnie zaprojektowanym układem scalonym. Zaprojektowanie układu scalonego jest kosztowne, zatem stosuje się taką implementację tylko dla modeli, których działanie zostało wcześniej potwierdzone. (zob. rys. 1) Innym rozwiązaniem jest realizacja sieci jako programu komputerowego. Jest to implementacja posiadająca dużą elastyczność, bardzo łatwo jest dokonać zmian w architekturze samej sieci. Dlatego też programy komputerowe wykorzystuje się

głównie na etapie badań samego modelu sieci, kiedy skuteczność końcowa modelu nie została dokładnie zbadana i udowodniona [1, s. 15].



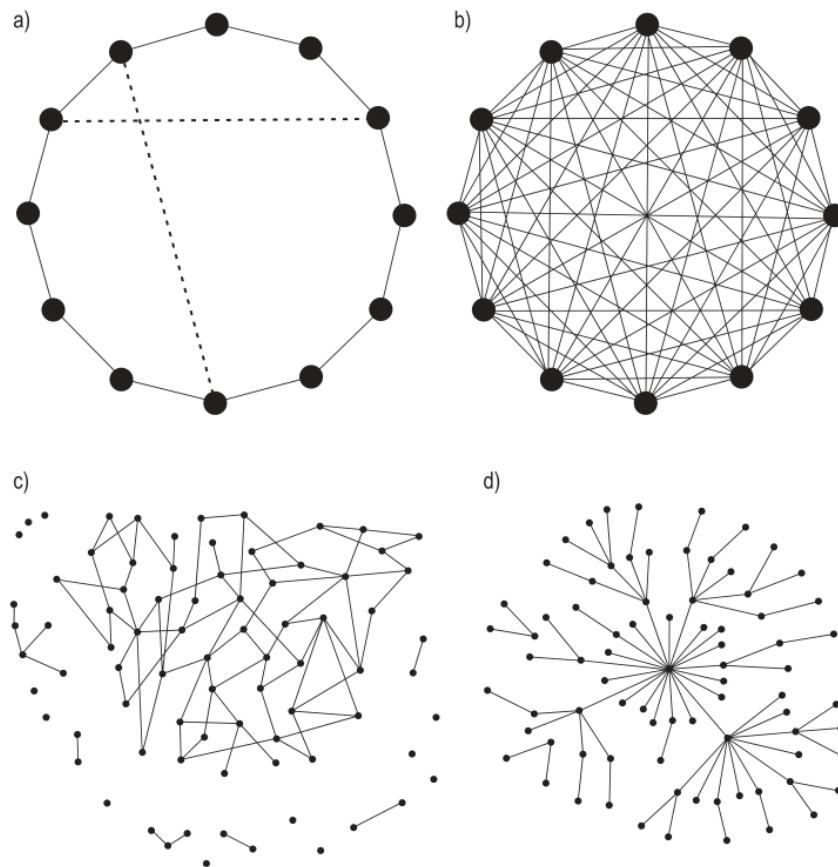
Rysunek 1.1. Sieć neuronowa zrealizowana w formie układu scalonego. Liczba neuronów samej sieci wynosi $16 \times 16 = 256$. Układ zawiera około 60 000 tranzystorów [1, s. 14] i [2].

Jak do tej pory sztuczne sieci neuronowe wykorzystywano do realizacji wielu skomplikowanych zadań. Jednym z najważniejszych zastosowań jest wykorzystanie sztucznych sieci neuronowych do rozpoznawania obiektów występujących na obrazie. Mogą być to obrazy różnego rodzaju - na przykład tę przedstawiające pismo odręczne (zob. rys 2). Sieci neuronowe są również szeroko wykorzystywane do rozwiązywania różnego rodzaju problemów optymalizacyjnych czy do przewidywania ruchów inwestorów na giełdzie [1, s. 15].



Rysunek 1.2. Baza danych *MNIST* zawiera skany odręcznych cyfr, które są powszechnie wykorzystywane w procesie uczenia różnych systemów przetwarzania obrazu. Źródło: Wikipedia

Sieci neuronowe – zarówno te biologiczne jak ich sztuczne odpowiedniki – są systemem złożonym. Mają one właściwości charakterystyczne dla sieci typu *małego świata*. To znaczy, że najkrótsza droga L między dwoma neuronami sieci jest stosunkowo krótka, a prawdopodobieństwo P tego, że neuron ma k sąsiednich neuronów, z którymi jest połączony jest wprost proporcjonalne do wartości k^{-Y} , gdzie stała wartość $2 < Y < 3$ – zależy od architektury sieci neuronowej. Na rysunku 3 przedstawiono kilka przykładów takich architektur (Za: [1, s. 15-16]).



Rysunek 1.3. (a) sieć jednowymiarowa z połączeniami bliskozasięgowymi (linie ciągłe) oraz z połączeniami typu *małego świata* (linie przerywane) (b) sieć całkowicie połączona o 12 węzłach (c) sieć przypadkowo połączona (graf przypadkowy) (d) sieć bezskalowa¹

¹ Sieć, w której rozkład liczby połączeń między węzłami jest zgodny z rozkładem *Yule-Simona*: $P(k) \sim k^{-Y}$, gdzie Y jest parametrem właściwym dla danej sieci i przyjmuje wartości z zakresu (2,3) Źródło: Wikipedia

1.1. POZNANIE A ROZPOZNANIE

Problem automatycznego rozpoznawania obiektów będę tutaj rozpatrywać bazując na idei podziału przestrzeni wyników na rozłączne zbiory (klasy). Wyróżnione zbiory powinny określać wartości wyników obserwacji właściwe poszczególnym obiektom badanego zbioru. *Wynik obserwacji* przedstawia zaobserwowane lub zmierzone wartości cech danego obiektu [3, s. 7].

Przez *przestrzeń wyników obserwacji* rozumiemy to samo co pod przestrzenią cech obiektu. Wynik obserwacji stanowi formalnie punkt przestrzeni cech.² *Rozpoznanie obiektu* polega zatem na obserwacji (pomiarze wartości) cech obiektu i sprawdzeniu do której klasy w przestrzeni cech zaobserwowany wynik należy. Taki sposób wnioskowania jest charakterystyczny dla zadania *klasyfikacji* [3, s. 7].

Podstawowym problemem jest sposób, w jaki dokonamy tego podziału przestrzeni na podstawie dostępnych danych. Kluczem do uzyskania satysfakcjonujących wyników może być sformułowanie we właściwy sposób zadania aproksymacji. Dzięki temu możliwe staje się wykorzystanie metody najmniejszych kwadratów do rekurencyjnego uzyskiwania rozwiązań. Taki sposób postępowania leży u podstaw działania sztucznych sieci neuronowych. Na bazie tego typu sieci zadania poznawania (uczenia się) i rozpoznawania uzyskują naturalną interpretację [3, s. 7-8].

Ujmując problem bardziej ściśle, poprzez rozpoznanie rozumiem wyróżnienie przedmiotu spośród innych występujących w analizowanym zbiorze obiektów. Poznanie zwykle oznacza zdobywanie wiedzy o czymś, nauczanie się czegoś na podstawie doświadczenia. Proces poznawania będę tutaj utożsamiać z uczeniem się. Wyróżniamy dwa sposoby uczenia się:

1. Uczenie się z nauczycielem (nadzorowane),
2. Uczenie się bez nauczyciela (nienadzorowane)

Uczenie z nauczycielem polega na wstępnym określeniu cech charakterystycznych dla każdego badanego obiektu: nauczyciel przedstawia wynik obserwacji cech wraz z określeniem właściwego rodzaju obiektu (właściwego wyniku klasyfikacji obiektu). Nauczanie polega na określeniu w przestrzeni cech zbioru wartości właściwych danego obiektu. Rezultatem uczenia jest określenie podzbiorów przestrzeni cech charakteryzujących poszczególne obiekty [3, s. 8].

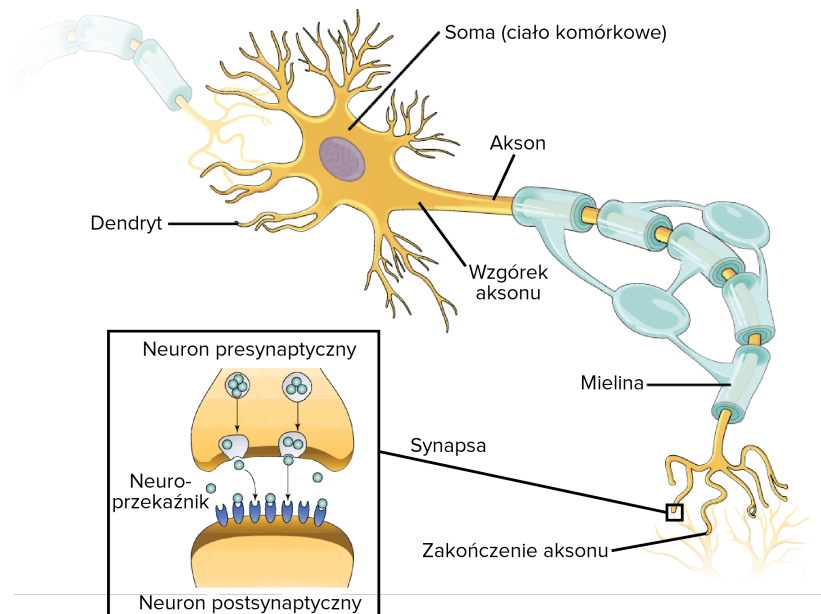
Sposób drugi polega na grupowaniu tych obserwacji, które powodują taką samą reakcję programu dokonującego klasyfikacji. Należy zaznaczyć, że nie zawsze

² Zakładamy tutaj, że badany przez nas wzór obiekt jest reprezentowany poprzez wektor liczb zwany wartościami cech. Przestrzeń na płaszczyźnie która zawiera wektory cech badanego obiektu nazywamy właśnie przestrzenią cech.

dochodzi tutaj do znalezienia relacji przyczynowo-skutkowej – często obserwacje są kojarzone w jedną grupę na podstawie innych, często nieintuicyjnych cech obiektów. Jeśli wyróżnione podzbiory tworzą podział przestrzeni cech (tzn. są rozłączne i wyczerpują całą badaną przestrzeń), to dzięki uzyskaniu odpowiednich klas abstrakcji (grup) możemy zdefiniować nowe obiekty w taki sposób, że każda z dostępnych klas definiuje obiekt [3, s. 8-9].

1.2. MODEL SZTUCZNEGO NEURONU

Jednym z najprostszych modeli sieci neuronowych jest **sztuczny neuron**. Pojedyncze sztuczne neurony to modele obliczeniowe, które zapoczątkowały rozwój uczenia maszynowego [4, s. 39]. Model sztucznego neuronu został zainspirowany – jak nazwa wskazuje – neuronem biologicznym, który został przedstawiony na rysunku 4.

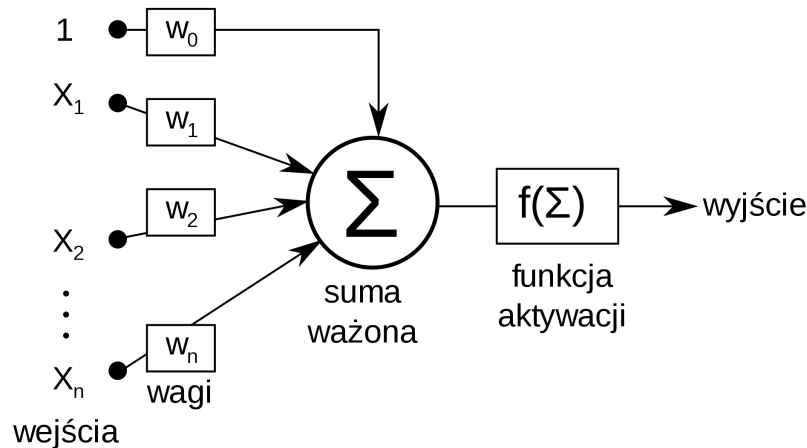


Rysunek 1.4. Model neuronu biologicznego. Źródło: Khan Academy

Jedną z pierwszych prób formalnego zdefiniowania zasady działania neuronu biologicznego podjęli w roku 1943 *McCulloch* i *Pitts*. Zaproponowali oni pierwszy model sztucznego neuronu tzw. **Neuron McCullocha-Pittsa** (zob. Rys. 5).

McCulloch i Pitts opisali neuron jako bramkę logiczną zawierającą binarne wyjścia. Zasada działania tego modelu jest prosta: do neuronu dociera wiele

sygnałów, które po integracji w ciele komórki generują sygnał na wyjściu, jeżeli zostanie przekroczona odgórnie ustalona wartość graniczna. Jeśli coś takiego nie będzie miało miejsca – neuron nie wygeneruje odpowiedzi na wyjściu [4, s. 40].



Rysunek 1.5. Model neuronu McCullocha-Pittsa. Jeśli wartość sumy ważonej przekroczy wartość progową funkcji aktywacji, to zostanie wygenerowana odpowiedź neuronu na wyjściu komórki. Źródło: Wikipedia

Innym przykładem jest **Perceptron Rosenblatta**. Frank Rosenblatt zaproponował algorytm zdolny do automatycznego uczenia się za pomocą optymalnych współczynników wag, które są przemnażane przez wartości obecne na wejściu neuronu, co pozwala w jednoznaczny sposób określić czy neuron ma przekazać dalej otrzymany sygnał. W kontekście uczenia nadzorowanego można wykorzystać ten mechanizm do realizacji zadania klasyfikacji obiektu do różnych klas [4, s.40].

1.3. CEL PRACY

Celem pracy nie jest opis konkretnej implementacji sieci neuronowej ani zaprojektowanie takiej sieci od podstaw. Celem, który zamierzamy zrealizować w niniejszej pracy jest zastosowanie pewnych cech i funkcji typowych dla sieci neuronowej do realizacji zadania kompozycji dzieła muzycznego. Sieć neuronowa jest mechanizmem opracowanym do rozpoznawania pewnych wzorców w danych i podejmowaniu na ich podstawie konkretnych decyzji lub tylko sugerowaniu pewnych rozwiązań właściwym decydentom. Te właściwości sieci zamierzamy właśnie wykorzystać. Planujemy zbudować system, który właściwie rozpozna cechy dzieła muzycznego i będzie w stanie je uogólnić do rozpoznawania ich w szerszej klasie przykładów

(zadanie klasyfikacji). Dodatkowo chcemy aby system dysponował zdolnością do tworzenia nowego utworu muzycznego na podstawie gotowego przykładu (zadanie generowania muzyki). Do realizacji tego zadania skorzystamy z pewnych cech, które zwyczajowo w literaturze przypisuje się właśnie sztucznym sieciom neuronowym.

ROZDZIAŁ 2

Podstawy aparatu matematycznego algebry liniowej

Nasze rozważania na temat modelowania dzieła muzycznego należy rozpocząć od zdefiniowania problemu w sposób formalny i wprowadzeniu podstawowych pojęć z zakresu niezbędnego aparatu matematycznego używanego do opisu i projektowania sieci neuronowych oraz do wprowadzenia pewnych podstawowych pojęć z zakresu teorii muzyki, których zdefiniowanie jest konieczne w celu prowadzenia dalszych rozważań. Realizacji tych dwóch zadań są poświęcone następne dwa rozdziały niniejszej pracy.

2.1. GRUPA ABELOWA

Definicja 2.1. Niech V będzie niepustym zbiorem wraz z zdefiniowanym działaniem wewnętrznym $+$. Zbiór $(V, +)$ jest **grupą abelową** jeśli spełnia on poniższe warunki¹:

$$\forall_{u,w,v \in V} \quad (u + w) + v = u + (w + v) \quad (\text{łączność})$$

$$\exists_{e \in V}, \quad \forall_{a \in V} \quad a + e = e + a = a \quad (\text{element neutralny})$$

$$\exists_{a \in V}, \quad \forall_{a^{-1} \in V} \quad a + (-a) = e + (-a) + a = e \quad (\text{elementy odwrotne})$$

a ponadto jeśli poniższe działanie jest przemienne to grupa jest grupą abelową.

$$\forall_{u,w \in V} \quad u + w = w + u$$

¹ Za: <http://en.math.uni.lodz.pl/~wiertelak/Temat20.pdf>

2.2. PRZESTRZENIE WEKTOROWE

Czwórkę postaci $(V; \mathbb{R}; +; \cdot)$, gdzie V jest zbiorem nad ciałem \mathbb{R} , $+$ jest działaniem wewnętrznym w zbiorze V oraz \cdot jest mnożeniem elementów zbioru V przez elementy ciała \mathbb{R} nazywamy *przestrzenią wektorową*, jeśli spełnione są następujące warunki:

- A1.** $(V; +)$ jest grupą abelową,
- A2.** $\bigwedge_{a \in \mathbb{R}^n, \bigwedge_{v, w \in V} a(v + w) = av + aw,$
- A3.** $\bigwedge_{a, b \in \mathbb{R}^n, \bigwedge_{v \in V} (a + b)v = av + bv,$
- A4.** $\bigwedge_{a, b \in \mathbb{R}^n, \bigwedge_{v \in V} a(bv) = (ab)v,$
- A5.** $\bigwedge_{v \in V} 1v = v.$

Poszczególne elementy zbioru V nazywamy *wektorami*, a elementy przestrzeni nad ciałem \mathbb{R} nazywamy *skalarami*. Zbiór wektorów V nazywamy *rzeczywistą przestrzenią wektorową* [5, s. 27].

Zdefiniujmy zbiór wektorów V nad ciałem \mathbb{R} gdzie, $v, w \in V$ i są one następującej postaci:

$$V = \{v : v = [v_1, \dots, v_n], \quad v_i \in \mathbb{R}, \quad i = 1, \dots, n\}.$$

Dla $v, w \in V$ w zbiorze v możemy następnie wprowadzić operację dodawania:

$$v + w = [v_1, \dots, v_n] + [w_1, \dots, w_n] = [v_1 + w_1, \dots, v_n + w_n].$$

Oznaczmy

$$-v = -[v_1, \dots, v_n] = [-v_1, \dots, -v_n]$$

jako element przeciwny do wektora v . Mamy zatem

$$v + (-w) = [v_1, \dots, v_n] + [-w_1, \dots, -w_n] = [v_1 - w_1, \dots, v_n - w_n]$$

Definiujemy element neutralny dodawania Θ postaci:

$$\Theta = [0, \dots, 0]$$

Wprowadzimy operację mnożenia wektora przez liczbę rzeczywistą $a \in \mathbb{R}$ (skalar) następującej postaci:

$$a \cdot v = a[v_1, \dots, v_n] = [av_1, \dots, av_n].$$

Zdefiniowany powyżej zbiór V wraz z operacją dodawania wektorów tworzy grupę abelową, a wraz z mnożeniem przez skalar jest przestrzenią liniową nad \mathbb{R} .

2.3. ILOCZYN SKALARNY I PRZESTRZENIE UNITARNE

Definicja 2.2.² Niech V będzie przestrzenią wektorową nad ciałem \mathbb{R} . Iloczynem skalarnym w przestrzeni V nazywamy funkcję $\langle \cdot, \cdot \rangle: V \times V \rightarrow \mathbb{R}$ spełniającą warunki:

- A1.** $\bigwedge_{v,w \in V} \langle v, w \rangle = \overline{\langle w, v \rangle},$
- A2.** $\bigwedge_{a,b \in \mathbb{R}} \bigwedge_{v_1, v_2 \in V} \langle av_1 + bv_2, w \rangle = a\langle v_1, w \rangle + b\langle v_2, w \rangle,$
- A3.** $\bigwedge_{v \in V \setminus \{\theta\}} \langle v, v \rangle > 0.$

Przestrzenią unitarną nazywamy parę $(V, \langle \cdot, \cdot \rangle)$, gdzie V jest przestrzenią wektorową, a $\langle \cdot, \cdot \rangle$ jest iloczynem skalarnym w przestrzeni V . Jeśli wiadomo jaki iloczyn skalarny został określony w przestrzeni V , to samą przestrzeń V nazywamy *przestrzenią unitarną* [5, s. 198-199].

Iloczyn skalarny ma następujące własności:

- A1.** $\bigwedge_{v \in V} \langle v, \theta \rangle = \langle \theta, v \rangle = 0,$
- A2.** $\bigwedge_{v \in V} \langle v, v \rangle \geq 0,$
- A3.** $\bigwedge_{v, w_1, w_2 \in V} \langle v, w_1 + w_2 \rangle = \langle v, w_1 \rangle + \langle v, w_2 \rangle,$
- A4.** $\bigwedge_{a \in \mathbb{R}} \bigwedge_{v, w \in V} \langle v, aw \rangle = a\langle v, w \rangle.$

Definicja 2.3. Niech V będzie przestrzenią unitarną. Normą (długością) wektora $v \in V$ nazywamy liczbę rzeczywistą nieujemną $\sqrt{\langle v, v \rangle}$. Normę wektora v oznaczamy przez $\|v\|$. Wektor $v \in V$ nazywamy wektorem *unormowanym*, jeżeli $\|v\| = 1$.

Jeśli V jest przestrzenią unitarną nad ciałem \mathbb{R}^n , to dla dowolnie wybranych $a \in \mathbb{R}$ i $v \in V$ zachodzi dodatkowo następująca równość:

$$\|av\| = |a| \|v\|$$

Dodatkowo, jeśli V jest przestrzenią unitarną, to dla dowolnych wektorów $v, w \in V$ zachodzą ponadto nierówności:

² Definicje opracowano na podstawie: [5, s. 198].

$$|\langle v, w \rangle| \leq \|v\| \cdot \|w\| \quad (\text{nierówność Schwarza})$$

$$\|v + w\| \leq \|v\| + \|w\| \quad (\text{nierówność Minkowskiego})$$

$$\left| \|v\| - \|w\| \right| \leq \|v - w\|.$$

2.4. PROSTE ZADANIE KLASYFIKACJI

W tym podrozdziale przedstawimy rozumowanie, którego celem jest pokazanie czytelnikowi w jaki sposób można, na poziomie ogólnym, zbudować prosty układ klasyfikujący obiekty.³ Układ ten będzie prostym klasyfikatorem binarnym, przydzielającym obiekty do dwóch niezależnych klas, a zasada jego działania będzie zbliżona, ale prostsza od zasady działania wspomnianego wyżej perceptronu.

Ustalmy, że $u, v \in \mathbb{R}^n$:

$$u = [u_1, \dots, u_n] \quad v = [v_1, \dots, v_n]$$

Niech u i v będą wyróżnionymi punktami przestrzeni \mathbb{R}^n . Zdefiniujmy na niej klasę $C^{(1)}$ punktów w przestrzeni \mathbb{R}^n jako zbiór punktów położonych bliżej punktu u niż v :

$$C^{(1)} = \left\{ x \in \mathbb{R}^n : \|x - u\| < \|x - v\| \right\} \quad (2.1)$$

Analogicznie określimy klasę $C^{(2)}$ jako zbiór punktów bliższych punktowi v :

$$C^{(2)} = \left\{ x \in \mathbb{R}^n : \|x - v\| < \|x - u\| \right\} \quad (2.2)$$

Punkty u i v rozumiemy jako wzorce wartości cech odpowiednich obiektów. Przyporządkowanie badanego punktu x przestrzeni cech \mathbb{R}^n do jednej z dwóch klas stanowi dość prosty przykład realizacji zadania rozpoznania obiektu.

Kolejny krok polega na wyznaczeniu granic dla klas $C^{(1)}$ i $C^{(2)}$.

Niech:

$$\|x - v\| = \|x - u\| \quad (2.3)$$

³ Opracowane na podstawie: [3, s. 11]

więc

$$\langle x - v | x - v \rangle = \langle x - u | x - u \rangle. \quad (2.4)$$

Zatem,

$$\langle x - v | x - v \rangle = \langle x - v | x \rangle - \langle x - v | v \rangle = \langle x | x \rangle - \langle x | v \rangle - \langle x | v \rangle + \langle v | v \rangle = \|x\|^2 + \|v\|^2 - 2\langle x | v \rangle$$

oraz

$$\langle x - u | x - u \rangle = \|x\|^2 + \|u\|^2 - 2\langle x | u \rangle.$$

Stąd,

$$\frac{1}{2}(\|v\|^2 - \|u\|^2) = \langle x | v \rangle - \langle x | u \rangle = \langle x | v - u \rangle = \langle v - u | x \rangle. \quad (2.5)$$

Niech $y = v - u$ oraz $w_0 = \frac{1}{2}(\|v\|^2 - \|u\|^2)$.

Wtedy:

$$\langle y | x \rangle = w_0.$$

Na tej podstawie możemy zdefiniować funkcję klasyfikującą:

$$f(x) = \begin{cases} 1, & \text{dla } \langle y | x \rangle > w_0 \\ -1, & \text{dla } \langle y | x \rangle < w_0 \end{cases} \quad (2.6)$$

Stąd,

$$\begin{cases} x \in C_1, & \text{dla } f(x) = 1 \\ x \in C_2, & \text{dla } f(x) = -1 \end{cases} \quad (2.7)$$

Przykład 2.1. Chcemy zaklasyfikować wektor x do którejś z klas C_1 lub C_2 .

$$v = [1, 2] \quad u = [3, -5] \quad x = [2, 4]$$

Otrzymujemy kolejno:

$$y = v - u = [1 - 3, 2 - (-5)] = [-2, 7]. \quad (2.8)$$

$$w_0 = \frac{1}{2}(\|v\|^2 - \|u\|^2) \quad (2.9)$$

$$= \frac{1}{2}((1 \cdot 1 + 2 \cdot 2) - (3 \cdot 3 + (-5) \cdot (-5))) \quad (2.10)$$

$$= \frac{1}{2}((1 + 4) - (9 + 25)) \quad (2.11)$$

$$= \frac{1}{2}(5 - 34) = \frac{1}{2} \cdot -29 = -14,5. \quad (2.12)$$

Wtedy

$$\langle y|x \rangle = [-2, 7] \cdot [2, 4] = [-2 \cdot 2 + 7 \cdot 4] = -4 + 28 = \mathbf{24}. \quad (2.13)$$

Jeśli $\langle y|x \rangle > w_0$ (**24** > **-14,5**), więc otrzymujemy 1.

Zatem z definicji funkcji $f(x) = 1$ wynika, że powinniśmy zaklasyfikować wektor x do klasy C_1 .

ROZDZIAŁ 3

Podstawy teorii muzyki i formatu MIDI

Zanim przejdziemy do omawiania poszczególnych części z jakich składa się utwór muzyczny. Należy wyjaśnić jedną zasadniczą kwestię - Muzyka istniała na tysiące lat zanim pojawiła się teoria ją opisująca. Koncepcje i reguły składające się na teorii muzyki są w zasadzie podobne do reguł jakie stosuje się w języku naturalnym. Podobnie jak opanowanie gramatyki, pozwala na swobodne prowadzenie konwersacji, tak opanowanie reguł tworzenia muzyki pozwala lepiej tworzyć własne kompozycje i czytać cudze [6, s. 22].

Wiele osób uważa, że teoria muzyki powstała w starożytnej Grecji, ale pierwsze instrumenty muzyczne, zdaniem historyków, powstały już około 7000 lat p.n.e. W okresie formowania się pierwszych osad ludzkich, na długo przed Starożytną Grecją. Na niektórych replikach fletów z kości słoniowej, które powstały w tamtym okresie nadal da się tworzyć utwory muzyczne. Fakt ten nie umniejsza oczywiście osiągnięciom Greków - Pitagoras z Samos stworzył dwunastodźwiękową przypominającą tą, której muzycy używają współcześnie. Stworzył także pierwsze *koło kwintowe*.¹ W istocie wkład starożytnych Greków w muzykę był na tyle znaczny, że przez dłuższy czas nie wprowadzano do niej żadnych znacznych modyfikacji [6, s. 23].

3.1. PODSTAWY TEORII MUZYKI

Wyjaśnienie podstaw teorii muzyki zaczniemy od jej części składowych: *nut*, *pauz* i *tempa*. Tempo (bit) to pulsacja dzieląca czas na równe odcinki. To czym jest bit dobrze uosabia mechanizm działania zegara. Na każdą minutę składa się 60 tyknień wskazówki sekund, a każde z tych tyknień to oddzielny bit. Jeśli przyspieszymy lub spowolnimy wskazówkę zegara, to zmienimy *tempo* tyknień. Rolą nut w muzyce jest przekazywanie informacji o tym co powinno być zagrane w każdej z oddzielnych

¹ Jest to rodzaj schematów pozwalający opisać pokrewne tonacje poszczególnych dźwięków.

tyknieć. Innymi słowy mówią one o tym jak długo i jak często powinno się zagrać określoną *wysokość dźwięku* [6, s. 31].

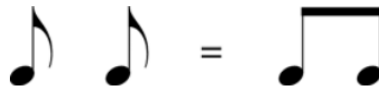
Każda nuta ma budowę trzyczęściową. Składa się z *główki*, *ogonka* i *chorągiewki*.

- Główka to okrągła część nuty, każda nuta ją ma. Cała nuta składa się tylko z główki,
- Ogonek to pionowa linia wychodząca od główki. Mają ją *ósemki*, *ćwierćnuty* i *półnuty*,
- Chorągiewka to mała linia wychodząca z lub od ogonka. Chorągiewkę ma *ósemka* i każda nuta od niej krótsza [6, s. 32-33] (zob. rys. 7).



Rysunek 3.1. Od lewej: cała nuta, ćwierćnuta i ósemka. Źródło: [6, s. 33].

Zamiast rysować chorągiewkę przy każdej nucie, można je połączyć *belką*, która trochę upraszcza zapis. (zob. rys 8).



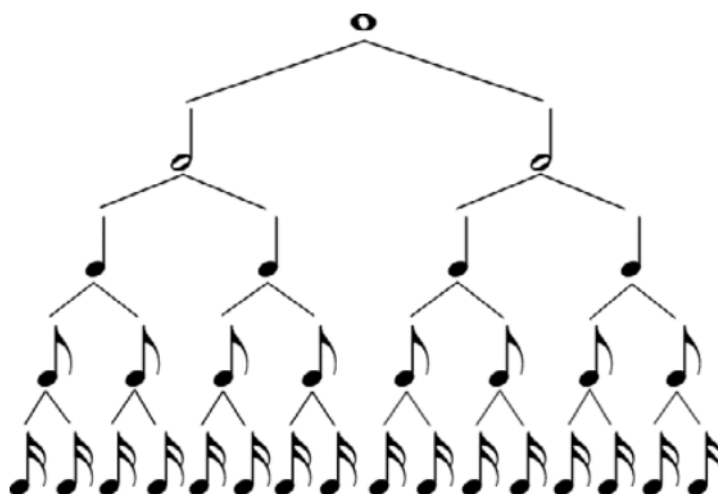
Rysunek 3.2. Ósemki można zapisywać oddzielnie lub razem łącząc je za pomocą belki. Źródło: [6, s. 33].

Kolejny rysunek przedstawia szesnastki z chorągiewkami. Sposób zapisu nie ma znaczenia, gdyż wszystkie gra się tak samo (zob. rys. 9).



Rysunek 3.3. Szesnastki pogrupowane jako: pojedyncze nuty, szesnastki połączone podwójnymi belkami oraz jako grupę nut połączonych jedną podwójną belką. Źródło: [6, s. 33].

Dobrym sposobem na prześledzenie w jaki sposób rozkładają się wartości poszczególnych nut jest skorzystanie z *drzewa nut* pokazanego na rysunku 10.



Rysunek 3.4. Drzewo nut składa się z całej nuty rozkładającej się na dwie półnuty, które następnie rozkładają się na cztery ćwierćnuty. Ćwierć nuty rozkładają się na ósemki, a ósemki na szesnastki Źródło: [6, s. 34].

Pauza to cisza w muzyce. Pauzy są szczególnie ważne gdy piszę się utwory, które mają być czytane przez innych ludzi. Pozwalają one na precyzyjniejsze wskazanie rytmu niż za pomocą samych nut. Rysunek 11 przedstawia względne wartości pauz. Podobnie jak w drzewie nut na samej górze jest pauza całonutowa, potem półnutowa, ćwierćnutowa, ósemkowa i szesnastkowa [6, s. 40-41]

tych podstawowych elementów [7, s. 22]. Poza nutami, tempem i pauzą. Utwór muzyczny składa się także z *interwałów*, *akordów* oraz *skal*.

Interwał to inaczej odległość pionowa między dwoma dźwiękami na pięciolinii (zob. rys. 12)



Rysunek 3.7. Interwał. Źródło: [7, s. 22].

Akord to współbrzmienie co najmniej trzech dźwięków ułożonych na pięciolinii jeden nad drugim. Akordy są ważnym elementem budującym *harmonie* w utworze. Harmonia zajmuje się łączeniem akordów w większe całości (zob. rys. 13) [7, s. 23].



Rysunek 3.8. Akord. Źródło: Opracowanie własne w programie MuseScore.

Zestaw następujących po sobie dźwięków, ułożonych w określonej kolejności, to *skala muzyczna*. Najczęściej spotyka się skale *minorową* (*molową*) i *majorową* (*durową*). Utwory oparte na skali molowej często uważa się za mające smutne brzmienie. Natomiast utwory oparte o skale durową często uważa się za radosne (zob. rys. 14) [6, s. 81 - 82].



Rysunek 3.9. Pentatonika z dźwiękiem centralnym D. Pentatonika to jedna z tradycyjnych skal muzycznych, która wywodzi się ze starożytności i jest popularna w muzyce ludowej Źródło: <https://pl.wikipedia.org/wiki/Pentatonika>

3.2. FORMAT MIDI

MIDI to z języka angielskiego *cyfrowy interfejs instrumentów muzycznych*. Służy on do przekazywania informacji pomiędzy cyfrowymi instrumentami muzycznymi. MIDI umożliwia komputerom, syntezatorom, keyboardom i kartom dźwiękowym komunikowanie się w wspólnym standardzie wymiany informacji oraz pozwala synchronizować się nawzajem. System ten pozwolił także na stworzenie łatwych w obsłudze *sekwencerów* i *syntezatorów perkusyjnych*.

Standard MIDI został stworzony w roku 1983 i znacznie ułatwił pracę z syntezatorami dźwięku. W czasach przed MIDI każdy syntezatorów należało indywidualnie zaprogramować [8].

W przeciwieństwie do innych formatów audio (jak .flac lub .wav) format MIDI sam w sobie nie przechowuje dźwięku, zawiera jedynie instrukcje jak dany dźwięk zrekonstruować. Jest swoistym rodzajem *cyfrowej pięciolinii* [9, s. 1 - 2]. Protokół MIDI pozwala na wysłanie informacji przez maksymalnie 16 różnych kanałów, co pozwala na jednoczesną grę do 16 różnych instrumentów. Można powiedzieć, że standard MIDI specyfikuje zarówno wymagania wstępne względem typu urządzenia mającego odtwarzać dźwięk jak i sam protokół transmisji, podług którego mają być wysyłane wiadomości w standardzie MIDI. Struktura pliku MIDI została pokazana na rysunku 12 [9, s. 1 - 2].

Time mark	Event	Note identity
Tick 1	Start	Pitch name 1
Tick 2	Start	Pitch name 2
Tick 3	End	Pitch name 1
Tick 4	End	Pitch name 2
Tick 5	Start	Pitch name 3
Tick 6	End	Pitch name 3
...




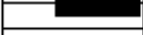


















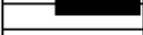















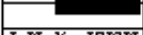
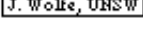

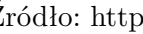










Rysunek 3.10. Struktura pliku MIDI. Plik MIDI składa się z tzw. *eventów*, które zawierają informacje o treści wiadomości MIDI oraz o czasie jej nadania.

Źródło: [9, s. 2 - 3] i [10].

Aby skorzystać z możliwości jakie daje nam MIDI należy zaopatrzyć się w instrumenty. Instrumenty mogą być dystrybuowane niezależnie jako pakiety *VSTi* dołączane oddzielnie do *syntezatora programowego* takiego jak *Helm* (zob. rys. 16) Do tworzenia muzyki w formacie MIDI można wykorzystać także *syntezator sprzętowy* taki jak *klawiatura MIDI* Klawiatura jest rodzajem kontrolera wysyłającego sygnały elektryczne w formacie MIDI, które sam protokół interpretuje i odtwarza jako dźwięk. Przykład takiej klawiatury to rysunek 17. warto zwrócić uwagę na to, że każdej nucie na klawiaturze przyporządkowany jest odpowiadający mu numer kodu formatu MIDI.



Rysunek 3.11. Interfejs programu Helm. Źródło: <https://tytel.org/helm/>

MIDI number		Note name	Keyboard	Frequency	
21	22	A0		27.500	
23		B0		30.868	29.135
24	25	C1		32.703	
26	27	D1		36.708	34.648
28		E1		41.203	38.891
29	30	F1		43.654	
31	32	G1		48.999	46.249
33	34	A1		55.000	51.913
35		B1		61.735	58.270
36	37	C2		65.406	
38	39	D2		73.416	69.296
40		E2		82.407	77.782
41	42	F2		87.307	
43	44	G2		97.999	92.499
45	46	A2		110.00	103.83
47		B2		123.47	116.54
48	49	C3		130.81	
50	51	D3		146.83	138.59
52		E3		164.81	155.56
53	54	F3		174.61	
55	56	G3		196.00	185.00
57	58	A3		220.00	207.65
59		B3		246.94	233.08
60	61	C4		261.63	
62	63	D4		293.67	277.18
64		E4		329.63	311.13
65	66	F4		349.23	
67	68	G4		392.00	369.99
69	70	A4		440.00	415.30
71		B4		493.88	466.16
72	73	C5		523.25	
74	75	D5		587.33	554.37
76		E5		659.26	622.25
77	78	F5		698.46	
79	80	G5		783.99	739.99
81	82	A5		880.00	830.61
83		B5		987.77	932.33
84	85	C6		1046.5	
86	87	D6		1174.7	1108.7
88		E6		1318.5	1244.5
89	90	F6		1396.9	
91	92	G6		1568.0	1480.0
93	94	A6		1760.0	1661.2
95		B6		1975.5	1864.7
96	97	C7		2093.0	
98	99	D7		2349.3	2217.5
100		E7		2637.0	2489.0
101	102	F7		2793.0	
103	104	G7		3136.0	2960.0
105	106	A7		3520.0	3322.4
107		B7		3951.1	3729.3
108		C8		4186.0	

Rysunek 3.12. Klawiatura MIDI, Źródło: <https://pl.wikipedia.org/wiki/MIDI>

ROZDZIAŁ 4

Reprezentacja i klasyfikacja utworu muzycznego

Na rysunku 18 została przedstawiona gama chromatyczna rozpoczynająca się od dźwięku C:



Rysunek 4.1. Skala dwunastodźwiękowa zwana także *chromatyczną*. Skala chromatyczna to skala, w której poszczególne stopnie oddalone są od siebie o pół tonu. Zawiera wszystkie dźwięki w obrębie danej oktawy, czyli łącznie 12 dźwięków. Krzyżyki oznaczają ruch w górę, bemole ruch w dół. Źródło: https://www.researchgate.net/figure/Chromatic-scale-starting-with-C-22_fig1_334131988

Skali chromatycznej odpowiada dowolny dwunastowymiarowy wektor wartości formatu *MIDI*. Dla przykładu skala chromatyczna rozpoczynająca się od dźwięku (C, 4), a kończąca się na dźwięku (C, 5), to wektor:

$$v = [60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71] \text{ gdzie } v \in \mathbb{R}^{12}.$$

W analogiczny sposób można rozisać dowolną gamę jako wektor wartości z których każda należy do przestrzeni liczb rzeczywistych. Należy jedynie pamiętać o tym, że wartości liczbowe poszczególnych dźwięków w formacie MIDI różnią się w zależności od wybranej oktawy i należy je rozisać zgodnie z powyższą klawiaturą MIDI.

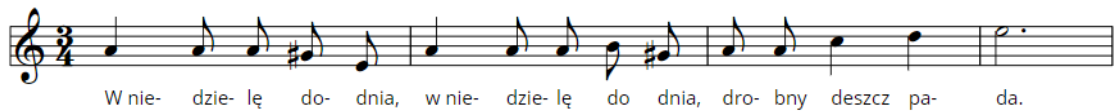
4.1. ZADANIE KLASYFIKACJI DŹWIĘKÓW

Stworzona w języku *Haskell* biblioteka *Euterpea*¹ pozwala w prosty sposób generować utwory muzyczne. Wykorzystuję się ją szeroko w muzyce algorytmicznej, analizie muzyki i syntezie dźwięku.

Euterpea tworzy dźwięki przesyłając komunikaty MIDI do urządzenia, który jest w stanie je odczytać. W najprostszym przypadku takim urządzeniem może być programowy syntezytor dźwięku [11]

Prezentowany przeze mnie prosty program w języku *Haskell* dokonuje klasyfikacji dźwięków, wykorzystując przy tym fragmenty dwóch gotowych utworów muzycznych:

1. Pierwszy utwór *Od Pilicy i Wolbroma* jest przedstawicielem muzyki ludowej i został udostępniony przez Instytut im. Oskara Kolberga (zob. Rysunek 18),
2. Utwór *Simple Jazz* pobrany z serwisu MuseScore (zob. Rysunek 19).



Rysunek 4.2. Utwór "Od Pilicy i Wolbroma", Źródło: Instytut im. Oskara Kolberga

Pierwszemu utworowi odpowiada wektor:

$$v = [69, 69, 69, 68, 64, 69, 69, 69, 71, 68, 69, 69]$$



Rysunek 4.3. Fragment utworu "Simple Jazz", Źródło: Musescore.com

Drugiemu utworowi odpowiada wektor u :

$$u = [69, 65, 67, 65, 69, 65, 69, 65, 67, 71, 65, 67]$$

Funkcje dokonujące obliczeń zostały zadeklarowane w następujący sposób:

¹ <https://www.euterpea.com/>

```

1  -- odejmowanie dwóch wektorów
2  odejmij :: [Float] -> [Float] -> [Float]
3  odejmij [] _ = []
4  odejmij _ [] = []
5  odejmij (x:xs) (y:ys) = x - y : (odejmij xs ys)
6
7
8  -- Iloczyn skalarny
9  dot :: [Float] -> [Float] -> Float
10 dot x y = sum $ zipWith (*) x y
11
12 klasyfikacja :: Float -> Int
13 klasyfikacja y_x
14     | y_x > w_0 = 1 -- klasa C1
15     | y_x < w_0 = -1 -- klasa C2
16
17 toAbsPitches :: [Pitch] -> [AbsPitch]
18 toAbsPitches ps = map absPitch ps
19
20
21 toPitches :: [AbsPitch] -> [Pitch]
22 toPitches as = map pitch as

```

Pomocnicze funkcje *ToPitches* i *toAbsPitches* umożliwiają konwersję z zapisu nutowego do liczb całkowitych i odwrotnie [12, s. 60].

Klasyfikacje możemy przeprowadzić na dwa sposoby:

- Na podstawie podobieństwa poszczególnych dźwięków do siebie,
- Na podstawie mierzalnych cech, które opisują utwór muzyczny. W tym wypadku skupimy się na *rytmie*.

4.1.1. Klasyfikacja na podstawie podobieństwa dźwięków

Mając dane dwa wektory v i u , sprawdźmy, czy wektor x jest bardziej podobny do muzyki ludowej (C_1), reprezentowanej przez wektor v , czy do muzyki jazzowej (C_2), reprezentowanej przez u .

$$v = [69, 69, 69, 68, 64, 69, 69, 69, 71, 68, 69, 69]$$

$$u = [69, 65, 67, 65, 69, 65, 69, 65, 67, 71, 65, 67]$$

Jako pierwszy przykład testowy wykorzystamy fragment utworu *The Entertainer* (zob.rys. 21) który zakodowaliśmy jako wektor x :

$$x = [60, 60, 60, 60, 60, 60, 60, 62, 63, 60, 62, 64, 64]$$



Rysunek 4.4. Fragment utworu "The Entertainer". Źródło:
<https://www.8notes.com/scores/13178.asp>

Przebieg klasyfikacji przebiega następująco. Najpierw obliczamy y :

$$y = [(69 - 69), (69 - 65), (69 - 67), (68 - 65), \\ (64 - 69), (69 - 65), (69 - 69), (69 - 65), \\ (71 - 67), (68 - 71), (69 - 65), (69 - 67)].$$

Zatem:

$$y = [0, 4, 2, 3, -5, 4, 0, 4, 4, -3, 4, 2].$$

Następnie obliczamy w_0 :

$$\begin{aligned} w_0 &= \frac{1}{2} (\|v\|^2 - \|u\|^2) \\ &= \frac{1}{2} [(69 \cdot 69) + (69 \cdot 69) + (69 \cdot 69) + (68 \cdot 68) \\ &\quad + (64 \cdot 64) + (69 \cdot 69) + (69 \cdot 69) + (69 \cdot 69) \\ &\quad + (71 \cdot 71) + (68 \cdot 68) + (69 \cdot 69) + (69 \cdot 69) \\ &\quad - (69 \cdot 69) + (65 \cdot 65) + (67 \cdot 67) + (65 \cdot 65) \\ &\quad + (69 \cdot 69) + (65 \cdot 65) + (69 \cdot 69) + (65 \cdot 65) \\ &\quad + (67 \cdot 67) + (71 \cdot 71) + (65 \cdot 65) + (67 \cdot 67)] \\ &= \frac{1}{2} (56473 - 53916) \\ &= \frac{1}{2} (2557) = 1278,5. \end{aligned}$$

Wtedy:

$$\begin{aligned} \langle y|x \rangle &= [0, 4, 2, 3, -5, 4, 0, 4, 4, -3, 4, 2] \\ &\quad \cdot [60, 60, 60, 60, 60, 60, 62, 63, 60, 62, 64, 64] = 1170. \end{aligned}$$

Jeśli, $\langle y|x \rangle < w_0$, to $f(x) = -1$, a to oznacza przydział do klasy C_2 .

$$x = [0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]$$

Na tej podstawie dokonujemy klasyfikacji. Zatem:

$$y = v - u = [0, 0, 0, 0, 1, 1, 0, -1, 0, 0, 0, 0, 1, 1, 0, 1, 0, -1, 1, 1, 0]$$

Następnie:

$$w_0 = \frac{1}{2}(\|v\|^2 - \|u\|^2) = 2.5.$$

Wtedy:

$$\langle y|x \rangle = 2.0.$$

Stąd:

$$\langle y|x \rangle < w_0, \quad \text{wtedy} \quad f(x) = -1$$

A to również oznacza przydział do klasy C_2 . Zatem klasyfikując na podstawie rytmu program ponownie uznał, że bluesowy utwór *The Entertainer* brzmi bardziej jazzowo niż ludowo.

4.2. PROCEDURA TESTOWA

Na podstawie powyżej opisanych dwóch metod przeprowadzono testy dla większej ilości utworów. Przebieg procedury testowania był następujący:

1. Wygenerowanie losowego wektora x ,
2. Obliczenie $y = v - u$,
3. Obliczenie wartości $w_0 = \frac{1}{2}(\|v\|^2 - \|u\|^2)$,
4. Obliczenie $\langle y|x \rangle$,
5. Przydział wektora x do klasy C_1 lub C_2 .

Jako, że wzorcowe wektory v i u zostały wprowadzone jako wartości całkowite, a do obliczenia iloczynu skalarnego potrzebujemy listy wartości zmiennoprzecinkowych, to ich konwersja na typ *Float* została przeprowadzona przy pomocy funkcji *floatList* zaimplementowanej następująco

```

1 |
2 | -- Konwersja listy int na float
3 | floatList :: [AbsPitch] -> [Float]
4 | floatList [] = []
5 | floatList (n:ns) = (x) : floatList(ns)
6 |     where x = fromIntegral n :: Float

```

Lp.	X_{input}	X_{output}
1.	C_1	C_2
2.	C_1	C_1
3.	C_1	C_2
4.	C_1	C_1
5.	C_1	C_1
6.	C_2	C_2
7.	C_2	C_2
8.	C_2	C_2
9.	C_2	C_1
10.	C_2	C_2

Tabela 4.1. Wyniki klasyfikacji na podstawie podobieństwa dźwięków. Źródło: Opracowanie własne.

Testy przeprowadzono w dwóch wariantach. W wariancie pierwszym (A) porównywano wektor wzorcowy danego typu muzyki (ludowej bądź jazzowej), z innym utworem z tego samego gatunku muzyki. Przykłady zostały pobrane z serwisu *MuseScore* lub z *Instytutu im. Oskara Kolberg*a, odpowiednio 5 utworów muzyki ludowej lub jazzowej. W wariancie (B) wektory wzorcowe były porównywane z losowo wygenerowanym innym wektorem, żeby sprawdzić jakie decyzje podejmuje algorytm dla losowo wygenerowanych sekwencji dźwięków.

4.2.1. (A) Porównywanie utworu wzorcowego z innym utworem

Podsumowanie: Wyniki klasyfikacji na podstawie podobieństwa dźwięków kształtują się następująco:

1. Dla muzyki ludowej klasyfikator osiągnął 60% dokładności (2 z 5 utworów zaklasyfikowano źle)
2. Dla muzyki jazzowej klasyfikator osiągnął 80% dokładności (1 z 5 utworów zaklasyfikowano źle)

Łącznie dla 10 przypadków, 7 razy dokonano poprawnej klasyfikacji (70% dokładności).

Podsumowanie: Wyniki klasyfikacji na podstawie podobieństwa rytmu kształtują się następująco:

1. Dla muzyki ludowej klasyfikator osiągnął 100% dokładności (0 z 5 utworów zaklasyfikowano źle)

Lp.	X_{input}	X_{output}
1.	C_1	C_1
2.	C_1	C_1
3.	C_1	C_1
4.	C_1	C_1
5.	C_1	C_1
6.	C_2	C_2
7.	C_2	C_2
8.	C_2	C_1
9.	C_2	C_2
10.	C_2	C_2

Tabela 4.2. Wyniki klasyfikacji na podstawie podobieństwa rytmu. Źródło:
Opracowanie własne.

2. Dla muzyki jazzowej klasyfikator osiągnął 90% dokładności (1 z 5 utworów zaklasyfikowano źle)

Łącznie dla 10 przypadków, 9 razy dokonano poprawnej klasyfikacji (90% dokładności).

4.2.2. (B) Porównywanie utworu wzorcowego z utworem losowym

Generowanie losowego wektora nut przeprowadzono za pomocą monady zaimplementowanej w następujący sposób:

```

1
2 import System.Random (randomRIO)
3 -- funkcje generujace listy
4 randomFloat :: Float -> IO([Float])
5 randomFloat 0 = return []
6 randomFloat n = do
7   r <- randomRIO (20, 108) -- zakres
8   rs <- randomFloat (n-1) -- generuje liste
9   return (r:rs)
10
11 randomBin :: Int -> IO([Int])
12 randomBin 0 = return []
13 randomBin n = do
14   r <- randomRIO (1,0) -- zakres
15   rs <- randomBin (n-1)

```

Lp.	X_{input}	X_{output}
1.	C_1	C_1
2.	C_1	C_2
3.	C_1	C_2
4.	C_1	C_2
5.	C_1	C_2
6.	C_2	C_2
7.	C_2	C_2
8.	C_2	C_1
9.	C_2	C_1
10.	C_2	C_2

Tabela 4.3. Wyniki klasyfikacji na podstawie podobieństwa rytmu dla wektorów losowych. Źródło: Opracowanie własne.

16 | `return (r:rs)`

Funkcja *randomFloat* generuje wektor wartości dźwiękowych o dowolnej długości, zakres losowania zmieniał się co iteracje i został ustalony na przedział 20 - 108, co odpowiada zakresowi wartości formatu *MIDI*. Funkcja *randomBin* generuje wektor złożony z zer i jedynek, również o dowolnej długości.

Podsumowanie: Wyniki klasyfikacji na podstawie podobieństwa rytmu kształtują się następująco:

1. Dla muzyki ludowej klasyfikator osiągnął 80% dokładności (1 z 5 utworów zaklasyfikowano źle)
2. Dla muzyki jazzowej klasyfikator osiągnął 60% dokładności (2 z 5 utworów zaklasyfikowano źle)

Łącznie dla 10 przypadków, 7 razy dokonano poprawnej klasyfikacji (70% dokładności).

Podsumowanie: Wyniki klasyfikacji na podstawie podobieństwa rytmu kształtują się następująco:

1. Dla muzyki ludowej klasyfikator osiągnął 10% dokładności (4 z 5 utworów zaklasyfikowano źle)
2. Dla muzyki jazzowej klasyfikator osiągnął 40% dokładności (3 z 5 utworów zaklasyfikowano źle)

Łącznie dla 10 przypadków, 3 razy dokonano poprawnej klasyfikacji (30% dokładności).

4.3. WNIOSKI

Najlepsze wyniki klasyfikacji osiągnięto podczas testowania na utworach rzeczywistych. Algorytm osiągał porównywalnie gorsze wyniki w wariancie z utworami generowanymi losowo. Klasyfikacja na podstawie podobieństwa dźwięków osiągała gorsze wyniki niż klasyfikacja na podstawie rytmu dla utworów rzeczywistych, ale jej skuteczność znacznie się pogorszyła dla utworów generowanych losowo, co może oznaczać, że algorytm klasyfikacji faktycznie dobrze wykrywa rytm melodii. Głównym wnioskiem płynącym z badań jest teza, że klasyfikacja na podstawie rytmu jest dokładniejsza, ale żeby ją potwierdzić, bądź odrzucić należało by wykonać testy dla większej ilości przypadków testowych.

ROZDZIAŁ 5

Przekształcenia geometryczne w przestrzeni

Przekształceniem geometrycznym lub *odwzorowaniem geometrycznym* nazywamy funkcję, której dziedziną i przeciwdziedziną jest zbiór wszystkich punktów przestrzeni \mathbb{R}^n . W węższym znaczeniu jest to funkcja wzajemnie jednoznaczna przekształcająca jeden zbiór punktów, zwany *figurą geometryczną* lub *wektorem* w inny obiekt tego samego typu.¹

5.1. DEFINICJA MACIERZY

Układ $m \cdot n$ elementów $a_{ij} \in \mathbb{R}$ ($i = 1, \dots, m, j = 1, \dots, n$) zapisany w następującej postaci:

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \cdots & \cdots & \cdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$

nazywamy macierzą o m wierszach i n kolumnach zdefiniowaną nad ciałem \mathbb{R} (lub macierzą wymiaru $m \times n$). [13, s. 85]. Macierz można też zapisywać krócej:

$$\mathbb{A} = [a_{i,j}]_{m \times n}$$

Liczby zawarte w macierzy nazywamy jej *elementami* (zob. rys. 23) [5, s. 18].

$$\mathbb{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 1 & 0 \end{bmatrix}$$

Rysunek 5.1. Macierz prostokątna \mathbb{A} wymiaru 3×2 .

Macierz o wymiarach $n \times n$ nazywamy *macierzą kwadratową stopnia n* (zob. rys. 24).

¹ Źródło: <https://www.math.edu.pl/przekształcenia-w-przestrzeni> [Dostęp: 26.01.22]

$$\mathbf{B} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

Rysunek 5.2. Macierz kwadratowa \mathbf{B} wymiaru 4×4 . Macierz kwadratowa ma taką samą liczbę wierszy i kolumn.

Odwołanie się do elementu macierzy następuje poprzez nazwę macierzy wraz z towarzyszącym jej dolnym indeksem wierszowym i kolumnowym. Dla przykładowej macierzy $\mathbf{A}_{3 \times 3}$ indeksy elementów wyglądają następująco:

$$\mathbf{A}_{3 \times 3} = \begin{bmatrix} \mathbf{a_{11}} & a_{12} & a_{13} \\ a_{21} & \mathbf{a_{22}} & a_{23} \\ a_{31} & a_{32} & \mathbf{a_{33}} \end{bmatrix}$$

Pogrubieniem oznaczono tak zwaną *przekątną główną macierzy*, którą tworzą elementy o równych wartościach indeksów wierszowych i kolumnowych.

Wektorem wierszowym nazywamy macierz złożoną tylko z jednego wiersza. Na przykład:

$$\mathbf{A}_{1 \times 3} = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$

Analogicznie, *wektorem kolumnowym* nazywamy macierz złożoną tylko z pojedynczej kolumny:

$$\mathbf{A}_{3 \times 1} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

5.2. ILOCZYN MACIERZY

Definicja 5.1. Iloczynem macierzy²

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \in \mathbb{M}_{m \times n} \quad , \quad \begin{bmatrix} b_{11} & \cdots & b_{1p} \\ \vdots & & \\ b_{n1} & \cdots & b_{np} \end{bmatrix} \in \mathbb{M}_{n \times p}$$

² Opracowano na podstawie [13, s. 88]

nazywamy macierz

$$\begin{bmatrix} c_{11} & \cdots & c_{1p} \\ \vdots & & \\ c_{m1} & \cdots & c_{mp} \end{bmatrix} \in \mathbb{M}_{m \times p},$$

gdzie

$$c_{ij} = \sum_{k=1}^n a_{ik} b_{kj}$$

i piszemy

$$[c_{ij}] = [a_{ij}] \cdot [b_{ij}] \text{ lub } [a_{ij}][b_{ij}].$$

Przykład 5.1. Załóżmy, że mamy dwie macierze, macierz $\mathbb{A}_{2 \times 3}$ i macierz $\mathbb{B}_{3 \times 2}$. Po ich pomnożeniu otrzymamy nową macierz \mathbb{C} , która posiada tyle wierszy ile miała macierz \mathbb{A} i tyle kolumn ile miała macierz \mathbb{B} . Zatem w tym wypadku macierz \mathbb{C} ma 2 wiersze i 2 kolumny. Zatem mnożenie:

$$\mathbb{A}_{2 \times 3} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \quad \mathbb{B}_{3 \times 2} = \begin{bmatrix} 1 & 3 \\ 2 & 4 \\ 1 & 2 \end{bmatrix}$$

daje nam:

$$\mathbb{C}_{2 \times 2} = \begin{bmatrix} 1 \cdot 1 + 2 \cdot 2 + 3 \cdot 1 & 1 \cdot 3 + 2 \cdot 4 + 3 \cdot 2 \\ 4 \cdot 1 + 5 \cdot 2 + 6 \cdot 1 & 4 \cdot 3 + 5 \cdot 4 + 6 \cdot 2 \end{bmatrix} = \begin{bmatrix} 8 & 17 \\ 20 & 44 \end{bmatrix}$$

.

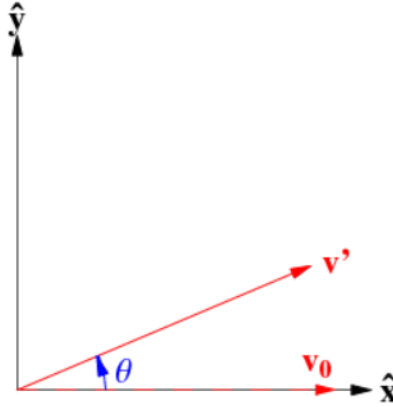
Jak można zauważyć macierz \mathbb{C} jest sumą iloczynów kolejnych elementów z każdego wiersza macierzy \mathbb{A} przez każdy kolejny element z każdej kolumny macierzy \mathbb{B} . Operacja mnożenia macierzy jest przydatna do wyznaczenia tzw. *Macierzy obrotu*.

5.2.1. Macierz obrotu

Macierz obrotu to szczególny rodzaj macierzy, umożliwiający poprzez mnożenie z danym wektorem v wymiaru n uzyskanie nowego wektora v' obróconego o zadany kąt ϕ względem początku układu współrzędnych umieszczonego nad ciałem \mathbb{R}^n .³

³ Źródło: <https://www.obliczeniowo.com.pl/258>

5.3. OBROTY W \mathbb{R}^2



Rysunek 5.3. Obrót wektora v_0 o kąt ϕ względem osi x . Za: [14].

Macierz obrotu w \mathbb{R}^2 oznaczamy zwykle symbolem $R(\phi)$ ⁴:

$$R(\phi) = \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix}$$

Zatem:

$$v' = R_\phi \cdot v_0.$$

Obrót wektora w \mathbb{R}^2 można wtedy wykonać poprzez zwykłe mnożenie macierzowe w następujący sposób:

$$v = R(\phi) \cdot v = \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \cdot \cos \phi & - & y \cdot \sin \phi \\ x \cdot \sin \phi & + & y \cdot \cos \phi \end{bmatrix}$$

Macierz ta obraca wektory kolumnowe w następujący sposób:

$$v' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

Wtedy współrzędne wektora:

$$v' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

mają wartości:

$$x' = x \cos \phi - y \sin \phi$$

⁴ Za: [14].

$$y' = x \sin \phi + y \cos \phi$$

Obrót jest przeciwny do obrotu zgodnego z ruchem wskazówek zegara jeżeli kąt ϕ jest dodatni. Macierz obrotu zgodnego z ruchem wskazówek zegara ma postać:

$$R(-\phi) = \begin{bmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{bmatrix}$$

Obroty w \mathbb{R}^2 spełniają własność przemienności. To znaczy, że nie ma znaczenia kolejność wykonywania obrotów. Niezależnie od kolejności otrzymuje się takie same wartości w wektorze wynikowym. Załóżmy, że mamy wektor:

$$v = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Przykład 5.2. Obróćmy wektor v o kąt 45° w kierunku zgodnym i przeciwnym do ruchu wskazówek zegara:

$$R(45^\circ) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} \cos \phi & -\sin \phi \\ \sin \phi & \cos \phi \end{bmatrix}$$

Z tego wynika, że współrzędne wynoszą:

$$x' = 1 \cdot \frac{\sqrt{2}}{2} - 0 \cdot \frac{\sqrt{2}}{2} = \frac{1}{\sqrt{2}}$$

oraz

$$y' = 1 \cdot \frac{\sqrt{2}}{2} + 0 \cdot \frac{\sqrt{2}}{2} = \frac{1}{\sqrt{2}}$$

Zatem otrzymane współrzędne wektora v' przy obrocie w kierunku przeciwnym do ruchu wskazówek zegara wynoszą:

$$v' = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

$$R(-45^\circ) = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{bmatrix}$$

Zatem współrzędne x' i y' wynoszą:

$$x' = 1 \cdot -\left(\frac{\sqrt{2}}{2}\right) - 0 \cdot -\left(\frac{\sqrt{2}}{2}\right) = -\left(\frac{1}{\sqrt{2}}\right)$$

$$y' = 1 \cdot -\left(\frac{\sqrt{2}}{2}\right) + 0 \cdot -\left(\frac{\sqrt{2}}{2}\right) = -\left(\frac{1}{\sqrt{2}}\right)$$

Otrzymujemy zatem, że współrzędne wektora v' dla obrotu w kierunku zgodnym z kierunkiem ruchu wskazówek zegara wynoszą:

$$v' = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix}$$

5.4. OBROTY W \mathbb{R}^3

Poniżej definiujemy macierze elementarne, które obracają wektor v względem, kolejno: osi Ox , Oy oraz Oz . Macierze obrotu mają postać:

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi & \cos \phi \end{bmatrix}$$

$$R_y(\phi) = \begin{bmatrix} \cos \phi & 0 & \sin \phi \\ 0 & 1 & 0 \\ -\sin \phi & 0 & \cos \phi \end{bmatrix}$$

$$R_z(\phi) = \begin{bmatrix} \cos \phi & -\sin \phi & 0 \\ \sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Przykład 5.3. Macierz $R_z(\phi)$ dla kąta $\phi = 90^\circ$ obraca wektor v względem osi OZ . Współrzędne wektora v' można wyliczyć dokonując pomnożenia macierzy R_z przez wektor v postaci:

$$v = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

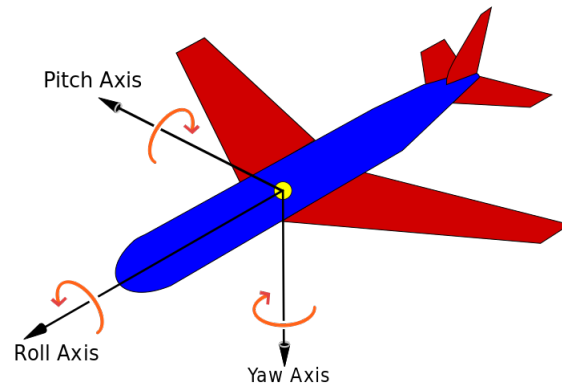
$$R_z(90^\circ) = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} \cos 90^\circ & -\sin 90^\circ & 0 \\ \sin 90^\circ & \cos 90^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

Analogiczne obliczenia możemy wykonać także dla pozostałych osi. Oś OX :

$$R_x(90^\circ) = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos 90^\circ & -\sin 90^\circ \\ 0 & \sin 90^\circ & \cos 90^\circ \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Oś OY :

$$R_y(90)^\circ = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} \cos 90^\circ & 0 & \sin 90^\circ \\ 0 & 1 & 0 \\ -\sin 90^\circ & 0 & \cos 90^\circ \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}$$



Rysunek 5.4. Na rysunku widać położenie trzech osi obrotu: Ox , Oy i Oz wraz z towarzyszącymi im kątami. Źródło:

https://pl.wikipedia.org/wiki/Macierz_obrotu [Dostęp: 11.01.22]

ROZDZIAŁ 6

Obroty wektorów nut w przestrzeni trójwymiarowej

Wspomniane w poprzednim rozdziale macierze rotacji można wykorzystać do tworzenia muzyki. W zdefiniowanej w rozdziale pierwszym przestrzeni wektorowej nad ciałem \mathbb{R} możemy rozważyć istnienie wektorów wartości dźwiękowych (nut) oraz operacje na nich. Obrót każdego z takich wektorów v gdzie $v \in V$ nad ciałem \mathbb{R} przeprowadzimy za pomocą mnożenia przez macierze obrotu dla osi: OX , OY oraz OZ . Obroty wektorów zostały zaimplementowane na dwa sposoby:

1. Obroty akordów o kąt 5° w przestrzeni \mathbb{R}^3 ,
2. Obroty wektora nut w przestrzeni \mathbb{R}^3 , reprezentującego własnoręcznie skomponowany utwór w tonacji C-dur, kolejno o kąty 5° , 10° , 15° oraz 20° stopni.

6.1. OBROTY AKORDÓW W \mathbb{R}^3

Na potrzeby eksperymentu przygotowano pogrupowane parami wektory od v_1 do v_4 następującej postaci:

$$v_1 = \begin{bmatrix} 60 \\ 64 \\ 67 \end{bmatrix} \quad v_2 = \begin{bmatrix} 69 \\ 61 \\ 64 \end{bmatrix} \quad v_3 = \begin{bmatrix} 67 \\ 71 \\ 62 \end{bmatrix} \quad v_4 = \begin{bmatrix} 65 \\ 69 \\ 60 \end{bmatrix}$$

Następnie każdy z tych wektorów przemnożono przez macierze obrotu odpowiadające poszczególnym osiom OX , OY oraz OZ przestrzeni \mathbb{R}^3 :

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{bmatrix}$$

$$R_y(\beta) = \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix}$$

$$R_z(\gamma) = \begin{bmatrix} \cos \gamma & -\sin \gamma & 0 \\ \sin \gamma & \cos \gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Macierz całego obrotu w \mathbb{R}^3 wyznacza się poprzez pomnożenie wektora v kolejno przez macierze poszczególnych osi, tj.

$$R = R_x(\alpha) \cdot R_y(\beta) \cdot R_z(\gamma).$$

gdzie kąt α odpowiada rotacji wokół osi OX , kąt β odpowiada rotacji wokół osi OY , a kąt γ odpowiada rotacji wokół osi OZ . W niniejszym eksperymencie wszystkie wektory obrócono o tę samą wartość, czyli kąt 5° .

6.2. IMPLEMENTACJA

```

1  type Vector = [Double]
2  type Matrix = [[Double]]
3
4  -- macierz obrotu
5  mx p = [[1,0,0], [0, cos p, sin p], [0, -sin p, cos p]]
6  my p = [[cos p, 0, -sin p], [0, 1, 0], [sin p, 0, cos p]]
7  mz p = [[cos p, sin p, 0], [-sin p, cos p, 0], [0, 0, 1]]
8  -- Dane
9  -- v1 do v4 to akordy
10 v1 :: Vector
11 v1 = [60,64,67]
12 v2 :: Vector
13 v2 = [69,61,64]
14 v3 :: Vector
15 v3 = [67,71,62]
16 v4 :: Vector
17 v4 = [65,69,60]
```

Wyznaczenie macierzy całego obrotu w \mathbb{R}^3 :

```

1  -- macierze obrotu 3D
2  mXY = mmult (mx 5) (my 5)
3  mXYZ = mmult (mXY) (mz 5) -- cala macierz obrotu o kat 5
```

Funkcja *mmult* służy do mnożenia macierzowego i została zaimplementowana następująco:

```

1 | mmult :: Num a => [[a]] -> [[a]] -> [[a]]
2 | mmult a b = [[sum $ zipWith (*) ar bc | bc <- (L.transpose b)] | ar <- a]

```

procedurę obracania wektorów zaimplementowano w poniższy sposób:

```

1 | wektor1 = do
2 |     toInt(concat(mmult (w1) mXYZ))
3 |     let nw1 = [80,37,66]
4 |     map pitch nw1

```

funkcja *pitch* służy konwersji wartości liczbowych na wartości dźwiękowe. Następnie do każdego z wektorów dodano ćwierćnuty (qn) za pomocą następującego fragmentu kodu:

```

1 | --obrocony wektor 1
2 | rw1 :: [(PitchClass, Octave)]
3 | rw1 = [(Gs,5),(Cs,2),(Fs,4),(G,5),(G,1), (Cs, 5)]
4 | rw1_play :: Music Pitch
5 | rw1_play = chord $ toMusicPitch qn (rw1)

```

Na koniec zaimplementowano monadę umożliwiającą odgrywanie parami oryginalnego akordu wraz z akordem obróconym:

```

1 | pierwsza_para :: Music Pitch
2 | pierwsza_para = akord1_music :+: rw1_play
3 |
4 | druga_para :: Music Pitch
5 | druga_para = akord2_music :+: rw2_play
6 |
7 | trzecia_para :: Music Pitch
8 | trzecia_para = akord3_music :+: rw3_play
9 |
10 | czwarta_para :: Music Pitch
11 | czwarta_para = akord4_music :+: rw4_play
12 |
13 | -- granie wszystkich par
14 | test :: IO ()
15 | test = do
16 |     putStrLn "pierwsza para"
17 |     play pierwsza_para
18 |     putStrLn "Druga para"
19 |     play druga_para
20 |     putStrLn "Trzecia Para"
21 |     play trzecia_para
22 |     putStrLn "Czwarta para"
23 |     play czwarta_para

```

6.3. WYNIKI

Przez v oznaczamy oryginalny wektor, a przez v' wektor obrócony o kąt $\phi = 5^\circ$ w \mathbb{R}^3 :

$$\begin{aligned} v_1 &= \begin{bmatrix} 60 \\ 64 \\ 67 \end{bmatrix} & v'_1 &= \begin{bmatrix} 80 \\ 37 \\ 66 \end{bmatrix} \\ v_2 &= \begin{bmatrix} 69 \\ 61 \\ 64 \end{bmatrix} & v'_2 &= \begin{bmatrix} 92 \\ 34 \\ 55 \end{bmatrix} \\ v_3 &= \begin{bmatrix} 67 \\ 71 \\ 62 \end{bmatrix} & v'_3 &= \begin{bmatrix} 95 \\ 42 \\ 50 \end{bmatrix} \\ v_4 &= \begin{bmatrix} 65 \\ 69 \\ 60 \end{bmatrix} & v'_4 &= \begin{bmatrix} 93 \\ 41 \\ 48 \end{bmatrix} \end{aligned}$$

6.4. WNIOSKI

Wartości każdego z obróconych wektorów od v_1 do v_4 zawierają podobne wartości liczbowe co wektory oryginalnych akordów. Wynika z tego lepsza harmonicznosc uzyskanych dźwięków i ich większe podobieństwo do oryginalnych akordów. Taki efekt uzyskaliśmy dzięki obracaniu wektorów o małe wartości kąta ϕ w przypadku większych kątów np. kąta 90° uzyskane dźwięki znacznie różniłyby się od oryginalnych, a przejście pomiędzy oryginałem, a nowo uzyskanym wektorem obróconym byłoby bardziej zauważalne, co odbiłoby się negatywnie na odczuciach estetycznych odbiorcy muzyki i harmonii samego dźwięku.

6.5. OBRÓT UTWORU MUZYCZNEGO W \mathbb{R}^3

Jako podstawę utworu bazowego zdefiniowanego w Haskellu następujące tablice dźwięków. Każda z poniższych tablic zawiera 12 dźwięków:

```
1 | linia1 = [C, E, G, B, D, F, A, C, E, G, B, C]
```

```

2 | linia2 = [C, F, A, D, G, C, F, B, E, A, D, G]
3 | linia3 = [C, A, F, D, B, G, E, C, A, F, D, B]
4 | linia4 = [C, G, E, B, G, D, B, F, D, A, F, C]

```

Następnie do każdej z tablic dołączono nuty. Funkcja *pcToQN* dołącza ćwierćnuty, natomiast funkcja *pcToHN* dołącza półnuty:

```

1 | part1 = map pcToHN (linia1)
2 | part2 = map pcToQN (linia2)
3 | part3 = map pcToHN (linia3)
4 | part4 = map pcToQN (linia4)

```

Ostatni krok to skonwertowanie takiego zapisu w formacie *dźwięk – długość* na typ muzyczny *Music Pitch*, co umożliwi odgrywanie utworu. Umożliwia to konstruktor *line \$*:

```

1 | part1_music = line $ part1
2 | part2_music = line $ part2
3 | part3_music = line $ part3
4 | part4_music = line $ part4

```

Mając skomponowany utwór, przystępujemy do jego obrotu w przestrzeni \mathbb{R}^3 . W tym celu każda z linii melodycznych skonwertowaliśmy na wektory:

```

1 | v1 :: Vector
2 | v1 = [0,4,7,11,2,5,9,0,4,7,11,0]
3 | v2 :: Vector
4 | v2 = [0,5,9,2,7,0,5,11,4,9,2,7]
5 | v3 :: Vector
6 | v3 = [0,9,5,2,11,7,4,0,9,5,2,11]
7 | v4 :: Vector
8 | v4 = [0,7,4,11,7,2,11,5,2,9,5,0]

```

Dzięki temu można wykonać operacje przemnożenia poprzez macierze obrotu osi *OX*, *OY* i *OZ*, analogiczne do przedstawionych wcześniej. Obrót zaimplementowano następująco:

```

1 | --obracanie wektorow
2 | wektor1 = do
3 |     toInt(concat(mmult (w1) mXYZ))
4 |
5 |
6 | wektor2 = do
7 |     toInt(concat(mmult (w2) mXYZ))
8 |
9 | wektor3 = do

```

```

10      toInt(concat(mmult (w3) mXYZ))
11
12
13  wektor4 = do
14      toInt(concat(mmult (w4) mXYZ))

```

Każdy pojedynczy obrót umożliwiał uzyskanie nowego wektora trójwymiarowego. Dla wektorów v_1 , v_2 , v_3 i v_4 przy przykładowym obrocie o kąt $\phi = 20^\circ$ uzyskaliśmy kolejno:

$$v'_1 = \begin{bmatrix} 7 \\ 3 \\ 3 \end{bmatrix} \quad v'_2 = \begin{bmatrix} 9 \\ 4 \\ 3 \end{bmatrix} \quad v'_3 = \begin{bmatrix} 5 \\ 8 \\ 4 \end{bmatrix} \quad v'_4 = \begin{bmatrix} 4 \\ 6 \\ 3 \end{bmatrix}$$

W tej procedurze testowej każdy kolejny obrót wykonywaliśmy o inny kąt, chcąc sprawdzić, czy także tym razem uzyskany utwór wynikowy będzie podobny do oryginalnego. Składając wszystkie próby razem uzyskaliśmy utwór o długości 12 dźwięków, dokładnie tak jak w utworze oryginalnym:

```

1  -- 1. Obrót o 5 stopni
2  rotate1 = [8,1,1,10,1,1,9,5,2,7,3,2] -- linia1
3  -- 2. o 10 stopni
4  rotate2 = [3,3,7,4,4,9,7,1,8,5,1,6] -- linia2
5  -- 3. o 15 stopni
6  rotate3 = [6,5,2,8,6,3,6,9,2,4,7,1] -- linia3
7  -- 4. o 20 stopni
8  rotate4 = [7,3,3,9,4,3,5,8,4,4,6,3] -- linia4

```

Następująca monada umożliwia odgrywanie utworu oryginalnego razem z obroconym:

```

1  test :: IO ()
2  test = do
3      putStrLn "pierwszy utwor"
4      play utwor
5      putStrLn "Ten sam utwor, ale po obrocie"
6      play rotate_utwor

```

6.6. WNIOSKI

Nowo uzyskany utwór zachował harmonie oryginalnego utworu, a uzyskane dźwięki wydają się podobne do oryginału. Wynika z tego następujący wniosek: Jeżeli obracamy utwór w przestrzeni nad ciałem \mathbb{R}^3 , to podobnie jak w przypadku akordów

należy to uczynić o małe wartości kątów, gdyż zwiększa to prawdopodobieństwo tego, że uzyskane dźwięki będą brzmieniowo zbliżone do oryginału. Nie musimy jednak obracać utworu cały czas o ten sam kąt – mogą to być inne wartości kątów o ile ich wartości nie są zbyt duże.

Zakończenie

Celem pracy było pokazanie jak duży potencjał tkwi w metodach uczeniach maszynowego takich jak sieci neuronowe. Okazuje się, że tak klasyczny problem algorytmiczny jak klasyfikacja obiektu do jednej z predefiniowanej klasy może umożliwić rozwój dziedzin zupełnie niezwiązanych z informatyką, takich jak tworzenie muzyki – będącej jak dotychczas niepodzielnym królestwem człowieka. Maszyna, jeżeli dobrze się ją zaprogramuje, potrafi rozpoznawać utwory muzyczne równie dobrze jak człowiek. Jednak potencjał metod numerycznych nie ogranicza się jedynie do zadań rozpoznawania. Maszyna jest zdolna także do komponowania muzyki. Metoda, którą tutaj pokazaliśmy ma wiele ograniczeń, ale na pewno istnieje w niej pewien potencjał, który jest wart poznania i dalszych rozważań, do których niniejsza praca miała zachęcić.

Bibliografia

- [1] Robert A. Kosiński. *Sztuczne sieci neuronowe Dynamika Nieliniowa i chaos*. Wydawnictwo WNT, Warszawa, 2017.
- [2] Leon O Chua. *CNN: A Paradigm for Complexity*. WORLD SCIENTIFIC, 1998.
- [3] Włodzimierz Kwiatkowski. *Metody Automatycznego Rozpoznawania Wzorców*. Bel-Studio, Warszawa, 2007.
- [4] Sebastian Raschka and Vahid Mirjalili. *Python Uczenie Maszynowe*. Helion S.A, Gliwice, 2019.
- [5] Jerzy Rutkowski. *Algebra Liniowa w Zadaniach*. Wydawnictwo Naukowe PWN, Warszawa, 2008.
- [6] Michael Pilhofer and Holly Day. *Teoria muzyki dla bystrzaków*. Septem, Warszawa, 2014.
- [7] Wojciech Usarzewicz. *Teoria muzyki dla muzyków komputerowych*. SELF-PUBLISHER, Warszawa, 2018.
- [8] MIDI. MIDI [online]. wikipedia : wolna encyklopedia, 2021-07-24 14:48Z. [dostęp: 2021-12-03 12:23Z]. Dostępny w Internecie: [//pl.wikipedia.org/wiki/MIDI?oldid=64150834](https://pl.wikipedia.org/wiki/MIDI?oldid=64150834).
- [9] Hélio de Oliveira and Raimundo Oliveira. Understanding midi: A painless tutorial on midi format. *cs.SD*, 2017.
- [10] Chi Tse, Xiao Fan Liu, and Michael Small. Analyzing and composing music with complex networks: Finding structures in bach's, chopin's and mozart's. *Music from a complex network perspective*, 09 2008.
- [11] Tomasz Obrebski. Euterpea. <https://obrebski.faculty.wmi.amu.edu.pl>, Marzec 2020. Dostęp: 17.08.21.
- [12] Paul Hudak and Donya Quick. *The Haskell School of Music*. Cambridge University Press, Cambridge, 2018.
- [13] H. Guściora and M. Sadowski. *Repetitorium z algebry liniowej*. Państwowe Wydawnictwo Naukowe, Warszawa, 1977.
- [14] Eric W. Weisstein. Three-dimensional rotation matrices. *MathWorld*, 2000.