МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет кібербезпеки, комп'ютерної та програмної інженерії
Кафедра інженерії програмного забезпечення

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри

С.В. Зибін

"___" _____2020 р.

ДИПЛОМНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСНИКА ОСВІТНЬОГО СТУПЕНЯ
МАГІСТРА

Тема: «Програмний метод розпізнавання звуку високої частоти»

Виконавець: Чемерис Максим Максимович

Керівниця: к.т.н доцентка Серебрякова Світлана Вікторівна

Нормоконтролер: к.т.н доцент Радішевський Микола Федорович

Київ 2020

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет кібербезпеки, комп'ютерної та програмної інженерії
Кафедра інженерії програмного забезпечення
Освітній ступінь магістр
Спеціальність 121 Інженерія програмного забезпечення
Освітньо-професійна програма «Програмне забезпечення систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Зибін С.В.
"___" _____ 2020 р

ЗАВДАННЯ
на виконання дипломної роботи студента
Чемериса Максима Максимовича

1. Тема дипломної роботи: «Програмний метод розпізнавання звуку високої частоти"
затверджена наказом ректора від 21.09.2020р. № 1715/ст.

2. Термін виконання проекту: з 5.10.2020 р. до 13.12.2020 р.

3. Вихідні дані до проекту: програмний продукт розробити за допомогою інструментів Web-програмування (серверна частина + web-інтерфейс).

4. Зміст пояснювальної записки:
   1. Методи і підходи до розпізнавання аудіо подій.
   2. Етапи процесу розпізнавання звуку.
   3. Програмний метод розпізнавання аудіо подій.
   4. Імплементація програмного методу у вигляді програмного додатку.

5. Перелік обов'язкових слайдів презентації:
   1. Діаграма класів програми.
   2. Діаграма компонентів додатку.
   3. Use Case діаграма.
   4. Функціональні можливості програми.
   5. Демонстрація першого запуску програми.
   6. Демонстрація роботи модулів програми.

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1.	Розробка та затвердження графіка роботи	05.10.20-11.10.20	
2.	Підготовка та написання 1 розділу	05.10.20-25.10.20	
3.	Перший нормо-контроль 1-2 розділів	26.10.20-01.11.20	
4.	Підготовка та написання 3 розділу	02.11.20-15.11.20	
5.	Підготовка та написання 4 розділу	16.11.20-22.11.20	
6.	Редагування та друк пояснювальної записки, графічного матеріалу	23.11.20-29.11.20	
7.	Проходження нормоконтролю, перепліт пояснювальної записки. Отримання відгуку керівника. Підготовка презентації та тексту доповіді.	30.11.20-06.12.20	
8.	Попередній захист дипломної роботи. Підпис ПЗ завідувачем кафедри на допуск до захисту та для отримання рецензії.	07.12.20-13.12.20	
9.	Отримання рецензії. Для проходження контролю на плагіат здати секретарю ДЕК текст ПЗ одним файлом.	07.12.20-13.12.20	
10.	Здати секретарю ДЕК: ПЗ, ГМ, CD-R з електронними версіями ПЗ, ГМ, презентацію, відгук керівника, рецензію, довідку про успішність, 2 папки, 2 конверта)	07.12.20-13.12.20	
11.	Захист дипломної роботи перед ЕК	22.12.20	

Дата видачі завдання 05.10.2020

Керівниця дипломної роботи: доц., к.т.н, доц. Серебрякова С.В.

Завдання прийняв до виконання: Чемерис М.М.

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Програмний метод розпізнавання звуку високої частоти»: 92 сторінок, 45 рисунки, 4 таблиці, 18 використаних джерел, 1 додаток.

АПРОКСИМАЦІЯ, ДКП (ДИСКРЕТНЕ КОСИНУСНЕ ПЕРЕТВОРЕННЯ), ІНТЕРФЕРЕНЦІЯ, КЕПСТР, КОГЕРЕНТНІСТЬ, КІХ-ФІЛЬТР, СПЕКТР, LDA (LATENT DIRICHLET ALLOCATION), PCA (PRINCIPAL COMPONENT ANALYSIS), SVM (SUPPORT VECTOR MACHINES).

Об’єкт дослідження - система розпізнавання звуку високої частоти та відповідного реагування на розпізнаний звук.

Предмет дослідження - програмний метод розпізнавання звуку високої частоти.

Мета дипломної роботи – підвищення рівня безпеки громадян шляхом скорочення часу реагування на небезпечні події.

Методи та засоби дослідження – програмні алгоритми розпізнавання звуків високої частоти.

В процесі роботи був зроблений глибокий аналіз існуючих алгоритмів розпізнавання аудіо подій. В результаті чого виявилось, що існуючим системам не вистачає ряду особливостей, які можна імплементувати в розроблюваний алгоритм, щоб покращити якість продукту.

Результати роботи можуть бути використані при розробці систем безпеки у місцях, де є ризик виникнення злочину чи небезпечної ситуації, що супроводжується звуком високої частоти (конкретних об’єктів, приватних будинків, вулиць населених пунктів тощо).

Розробка та дослідження проводилися під управлінням ОС Windows 10. Розробка програми проводилася у середовищі VS Code, на мовах програмування Python та TypeScript.

ABSTRACT

Explanatory note to *Software Method for High Frequency Sound Recognition*: 92 pages, 45 figures, 4 tables, 18 sources, 1 additions.

APPROXIMATION, DCT (DISCRETE COSINE TRANSFORM), INTERFERENCE, CEPSTRUM, COHERENCE, FIR FILTER, SPECTRUM, LDA (LATENT DIRICHLET ALLOCATION), PCA (PRINCIPAL COMPONENT ANALYSIS), SVM (SUPPORT VECTOR MACHINES).

The object of research is a high frequency sound recognition system, and the corresponding response to the recognized sound.

The subject of research is a software method of high frequency sound recognition.

The purpose of the thesis is to increase the level of security of citizens by reducing the response time to dangerous events.

Research methods and tools - software algorithms for recognizing high frequency sounds.

In this work there is a deep analysis of existing algorithms of recognition of audio events. As a result, it turned out that the existing systems lack a number of features that could be implemented in the developed algorithm to improve product quality.

The results of this work can be used in the development of security systems in places where there is a risk of crime or a dangerous situation, accompanied by high-frequency sound (specific objects, private houses, streets, etc.).

Development and research were conducted under Windows 10. The program was developed in VS Code, in the programming languages Python and TypeScript.

ЗМІСТ

СПИСОК ТЕРМІНІВ ТА СКОРОЧЕНЬ

Апроксимація — науковий метод, що полягає в заміні одних об'єктів іншими, в якомусь сенсі близькими до вихідних, але більш простими.

ДКП (Дискретне косинусне перетворення) — одне з ортогональних перетворень. Варіант косинусного перетворення для вектора дійсних чисел.

Інтерференція — явище накладання двох або більше когерентних хвиль, в результаті чого в одних місцях спостерігається підсилення результуючої хвилі (інтерференційний максимум), а в інших місцях послаблення (інтерференційний мінімум).

Кепстр — один з видів обробки сигналів, функція зворотного перетворення Фур'є від логарифма спектра потужності сигналу.

Когерентність —властивість хвилі зберігати свої частотні, поляризаційні й фазові характеристики.

КІХ-фільтр — фільтр з кінцевою імпульсною характеристикою — один з видів лінійних цифрових фільтрів, характерною особливістю якого є обмеженість за часом його імпульсної характеристики.

Спектр — це розподіл значень фізичної величини (зазвичай енергії, частоти або маси).

LDA (Latent Dirichlet allocation) — прихований розподіл Діріхле — генеративна статистична модель, що дозволяє пояснювати набори спостережень ненаглядовими групами, які пояснюють, чому деякі частини даних подібні.

PCA (Principal component analysis) — метод головних компонент — метод факторного аналізу в статистиці, який використовує ортогональне перетворення множини спостережень з можливо пов'язаними змінними у множину змінних без лінійної кореляції, які називаються головними компонентами.

SVM (Support vector machines) — метод опорних векторів — метод аналізу даних для класифікації та регресійного аналізу за допомогою моделей з керованим навчанням з пов'язаними алгоритмами навчання, які називаються опорно-векторними машинами.

ВСТУП

Актуальність теми. У нинішній час за допомогою програмного та апаратного забезпечення можна досягти високого рівня людської безпеки. Протягом десятиліть правоохоронні організації все частіше використовують алгоритми виявлення та обробки звуку (розпізнавання звуків пострілів, вибухів, битого скла тощо), та системи локалізації для виявлення можливих інцидентів чи злочинів, що допомагає миттєво реагувати на них. Ці системи стали доступними завдяки простим конфігураціям мікрофонів, за допомогою яких можливо визначити місце джерела звуку з точністю до метра.

Область використання алгоритму розпізнавання звуку високої частоти не обмежується лише проблемою безпеки. За допомогою цього алгоритму можливо сповіщати людей із вадами слуху про можливу небезпеку, пришвидшити приїзд відповідальних служб у випадку небезпечних ситуацій. Із розвитком баз даних, датчиків, аудіо пристроїв та обчислювальних потужностей постала проблема в класифікації звуків. Із розвитком Інтернету та мікроконтролерів можливе впровадження елементів “розумного дому”, керованих за допомогою звуку високої частоти (звуковий сигнал, сигналізація, дзенькіт скла тощо).

Не дивлячись на те, що значну роль у цьому процесі може відігравати відеоспостереження, за допомогою якого можна отримати більш точну картину подій, аудіо аналітика має деякі переваги порівняно з відео аналітикою: вартість обладнання (мікрофони, фільтри, мережеве обладнання) та їх підтримка є значно дешевшими, ніж вартість та підтримка відеокамер; при роботі системи в режимі реального часу потік даних аудіо інформації значно менший за об’ємом, ніж потік даних з камер, що уможливлює більш лояльні вимоги до каналу передачі даних. Системи аудіо аналітики можуть бути особливо корисними для міського спостереження, де можна автоматично почати транслювати відео наживо на поліцейському дисплеї з місця пострілу чи вибуху. Особливо гостро ця потреба відчувається у об'єктах та територіях, значних для держави (заводи, станції тощо). Технології аудіо аналітики також можуть використовуватись при визначенні подій. Зі зростанням розуміння можливостей даних систем

застосування аудіо аналітики тільки поширюватиметься.

Більшість програмних продуктів, пов’язаних із розпізнаванням аудіо подій, є міжнародними масштабними проектами з історією, що застосовуються у галузі ведення оборони та внутрішньої безпеки цілих міст. Однак існує ряд особливостей, які можна імплементувати в розроблюваний алгоритм, щоб покращити як якість послуг, так і розуміння продукту в цілому.

Об’єкт дослідження – система розпізнавання звуку високої частоти та відповідного реагування на розпізнаний звук.

Предмет дослідження - програмний метод розпізнавання звуку високої частоти.

Мета дипломної роботи – розробка програмного методу розпізнавання аудіо подій високої частоти.

Задачі дипломної роботи:

1. Методи і підходи до розпізнавання аудіо подій.
2. Етапи процесу розпізнавання звуку.
3. Програмний метод розпізнавання аудіо подій.
4. Імплементація програмного методу у вигляді програмного додатку.

Методи та засоби дослідження – програмні алгоритми розпізнавання звуків високої частоти. Програмний продукт було розроблено з використанням мови програмування Python, інтерфейсна складова була написана за допомогою TypeScript.

Наукова новизна дипломної роботи полягає у застосуванні програмних методів до розробки нового програмного додатку розпізнавання звуків високої частоти.

Практична цінність. Розроблений програмний додаток можна застосовувати на практиці для розпізнавання високочастотних звуків.

РОЗДІЛ 1
МЕТОДИ І ПІДХОДИ ДО РОЗПІЗНАВАННЯ АУДІОПОДІЙ

1.1. Основні задачі при розпізнаванні звуку

Розпізнавання звуку — процес перетворення сигналу в цифрову інформацію, наприклад текстові дані. Нині, в епоху смартфонів та голосових помічників дуже популярні алгоритми розпізнавання мовних сигналів — звуків, синтезованих людським мовним апаратом [1]. Як відомо, звуковий сигнал в комп'ютерних технологіях може представлятися в вигляді деякого набору відліків його амплітуд, вироблених через певні проміжки часу (період дискретизації) і подаються деякою кількістю двійкових розрядів (розрядність вибірки). Такий формат зручний для зберігання звукового сигналу і його перетворення назад в безперервний сигнал. Однак, деякі операції обробки звукового сигналу в такому форматі буває не дуже зручно, або взагалі неможливо здійснювати. Це пов'язано з тим, що звуковий сигнал складається зі складових його частот з певною амплітудою і фазою.

Таким чином, застосування таких операцій обробки звукового сигналу як "фільтр нижніх частот", "фільтр верхніх частот", або процес розпізнавання голосових сигналів, вимагає перетворення формату звукового сигналу у вигляді відліків його частотного спектра. Після цього перетворення звуковий сигнал буде представлений у вигляді коефіцієнтів, відповідних амплітуд і фаз частот, з яких складається цей сигнал.

Основними задачами при розпізнаванні сигналів є:

1. Параметризація вхідного сигналу — виділення придатних для аналізу параметрів з вхідного звукового сигналу.

2. Аналіз параметризованого сигналу — пошук відповідності між вхідним звуковим сигналом та аналогом у базі звуків.

Ці задачі являються основними як для існуючих, старих систем, так і для нових систем, що засновуються на використанні нейронних мереж.

Після перетворення вхідного звукового сигналу, отримаємо набір

параметрів, тобто числове представлення окремого фрейму. Тепер постає задача безпосереднього розпізнавання вхідного сигналу. Залежно від системи розпізнавання та алгоритму, що використовується, задача перетворення цифрового сигналу може змінитися на задачу порівняння вхідного сигналу з деяким, вже існуючим, еталоном. [7] На стан вхідного сигналу впливають такі характеристики, як швидкість сигналу, гучність, наявність шумів. Алгоритми, що базуються на перетворенні є складними у реалізації, часто потребують великих обчислювальних можливостей та не гарантують ідеальну точність розпізнавання. [8] Підхід, що базується на порівнянні вхідного сигналу з еталонами був першим, який досить успішно вирішував задачу розпізнавання звуків.

Але з розвитком технологій розпізнавання з’являлися нові алгоритми, які використовували інші підходи до процесу розпізнавання і не потребували еталонів для свого функціонування. Наступним успішним алгоритмом, що використовувався для вирішення задач розпізнавання, став алгоритм, що використовує приховані марковські моделі, які базуються на описі стохастичних процесів та статистичних законах. В наш час найбільш популярні системи розпізнавання використовують нейронні мережі, адаптовані до задач розпізнавання звуків [9].

Звук як фізичне явище є подовжньою хвилею, що розповсюджується в пружному середовищі. У подовжній хвилі частинки коливаються вперед-назад біля положення стійкої рівноваги у напрямі розповсюдження хвилі. Подовжня хвиля є чергуванням згущувань (ущільнень) і розріджень у напрямі переміщення хвилі. Звук може розповсюджуватися тільки в пружному середовищі, тобто в середовищі, яке здатне відновлювати свою первинну форму, спотворену (деформовану) в результаті короткочасної дії на неї сили пружності. Джерелом виникнення хвилевого руху (джерелом звуку) може служити будь-яке тіло, здатне здійснювати пружні коливання — мембрана, дифузор, металева пластина, струна, стовп повітря (у трубах) і так далі.

Швидкість розповсюдження звукової хвилі залежить від щільності і

пружності середовища (вона пропорційна пружності і обернено пропорційна щільності). Швидкість також залежить від температури і тиску. У таблиці 1 приведені дані про швидкість звуку в деяких середовищах.

Таблиця 1.

Швидкість звуку в різних середовищах

Середовище	**Щільність, г/см^3 при t = 20^{o}C**	**Швидкість, м/с**
Алюміній	2,7	5100
Сталь	7,8	5000
Мідь	8,89	3600
Дерево	0,6-0,9	3000-4000
Пробка	0,22-0,26	500
Гума	0,95	35-70
Вода	1	1456
Повітря	1,29•10-3	344

Незважаючи на те що щільність металів на 3-4 порядки вища, ніж щільність газів, і в 4-5 разів вища, ніж щільність рідин, швидкість звуку в металах більша за рахунок більшої пружності.

Реальний звук, який ми чуємо — це складне явище. По-перше, звукова хвиля, що створює тиск на барабанну перетинку вуха, на практиці є результуючою звуковою хвилею від декількох джерел і по-друге, на її формування здійснюють вплив різні явища, такі як інтерференція, відбиття, заломлення, поглинання і розсіяння.

Інтерференція. Явище інтерференції базується на принципі суперпозиції хвиль, сенс якого зводиться до наступного: якщо в середовищі одночасно розповсюджується система декількох різних хвиль, то кожна з хвиль розповсюджується незалежно від інших. При цьому результуюча швидкість, зсув, прискорення кожної частинки середовища дорівнюють векторним сумам відповідних величин, обумовлених кожній з хвиль порізно. Таким чином, при накладенні, наприклад, двох хвиль, результуюча хвиля не схожа на початкові, більше того - вона може як збільшуватися по амплітуді, так і зменшуватися (рис.

1.1 і рис 1.2). Це і є інтерференція, тобто посилення коливань в одних точках простору і ослаблення коливань в інших крапках в результаті накладення двох або кількох звукових хвиль.

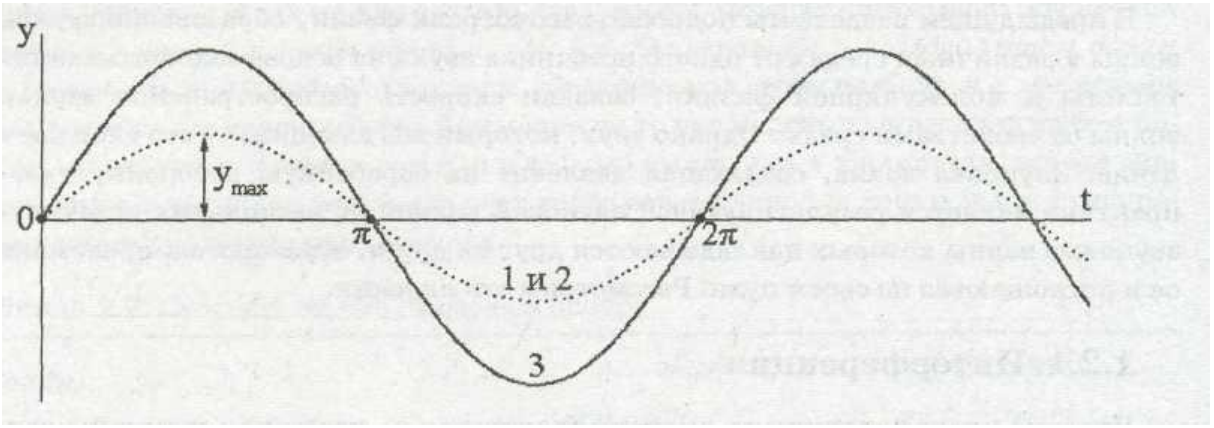


Рис. 1.1. Накладення двох хвиль з однаковою амплітудою, частотою і фазою

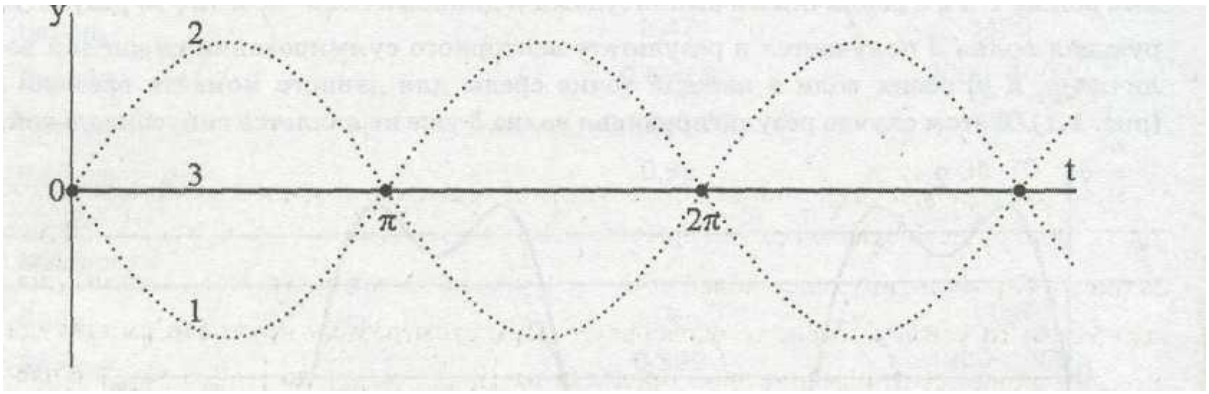


Рис. 1.2. Накладення двох хвиль з однаковою амплітудою, частотою і протилежними фазами

Коли ми чуємо звуки різних, але близьких за величиною частот одразу від двох джерел, до нас приходять то гребені обох звукових хвиль, то гребінь однієї хвилі і западина іншої. В результаті накладення двох таких хвиль звук то посилюється, то слабшає, поки різниця фаз невелика. Цей коливальний процес із наростанням і спаданням амплітуди результуючої хвилі називають биттям (рис. 1.3).

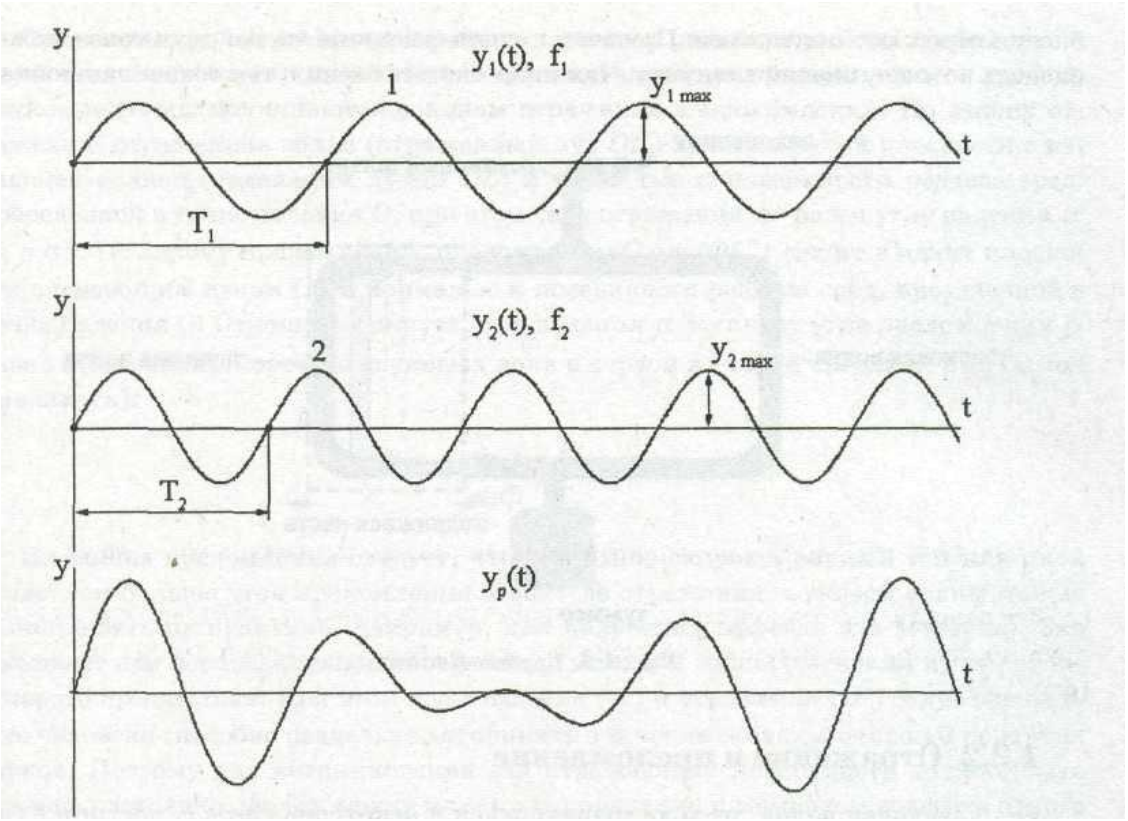


Рис. 1.3. Биття в результаті накладення двох хвиль

Відбиття і заломлення. Якщо звукова хвиля, що розповсюджується в деякому середовищі 1, досягає межі перетину цього середовища з іншим середовищем 2, то виникають відбита і заломлена хвилі (рис. 1.4).

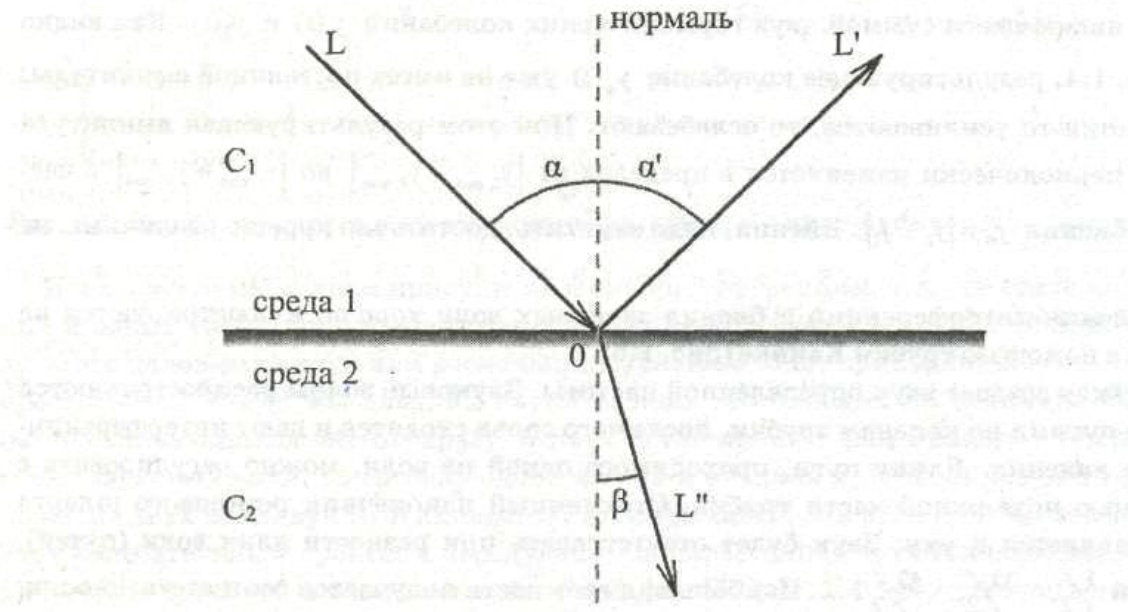


Рис. 1.4. Відбиття і заломлення хвиль на межі двох середовищ

Відбита хвиля розповсюджується від межі розділу в цьому ж середовищі 1, що і первинна (падаюча) хвиля. Заломлена хвиля розповсюджується в середовищі 2. Звукові хвилі підкоряються законам відбиття і заломлення. За законом відбиття, відбита хвиля (відбитий промінь OL) лежить в одній площині з падаючою хвилею (падаючим променем OL) і нормаллю до поверхні розділу середовищ, проведеної в точці падіння. При цьому кут відбиття дорівнює куту падіння. За законом заломлення, заломлений промінь (OL") лежить в одній площині з падаючим променем OL і нормаллю до поверхні розділу середовищ, проведеної в точці падіння O. Відношення синуса кута падіння до синуса кута заломлення дорівнює відношенню швидкостей звукових хвиль в першому і другому середовищах.

Із закону заломлення виходить, що чим вища швидкість звуку в тому або іншому середовищі, тим більший кут заломлення. Наслідком відбиття є луна, яка в деяких випадках може бути небажаним ефектом. Луна виникає при перпендикулярному відбитті звукової хвилі від деякої перешкоди. При цьому кути падіння і відбиття дорівнюватимуть 0. Вухо людини здатне роздільно сприйняти протягом секунди близько 10 коротких звуків. Тому для виникнення луни відбиваюча поверхня має бути віддалена настільки, щоб між моментом появи і моментом повернення одного звуку пройшло не менше 0,1 с. При швидкості розповсюдження звукової хвилі в повітрі ~ 340 м/с така мінімальна відстань складає близько 17 метрів.

Поглинання і розсіювання. Енергія звукової хвилі в процесі її розповсюдження поглинається середовищем. В результаті поглинання зменшується амплітуда коливань і відстань, на яку розповсюджується звук (відбувається загасання коливань). Ступінь поглинання звукової енергії в рідинах і газах залежить, з одного боку, від властивостей середовища, а з іншої — від частоти звукових коливань. Чим вища частота звукових коливань, тим більш хаотична молекулярна швидкість молекул в елементі стискуваного об'єму, тим більш молекулярного розсіяння зазнає на своєму шляху звукова хвиля і тим на меншу відстань передадуться звукові коливання.

Ще одна причина загасання коливань — розсіяння звуку, яке виникає в результаті взаємодії звукової хвилі з численними перешкодами, що зустрічаються на її шляху (зустрічні потоки повітря, завихрення, вітер). В результаті зіткнення з цими перешкодами звукова хвиля як би "розсипається" на безліч хвиль, які розповсюджуються у різних напрямах. Цей чинник більшою мірою має вплив на низькі частоти.

Дифракція. Дуже важлива властивість звукових хвиль — здатність оминати малі перешкоди. Суть цього явища полягає в тому, що плоска звукова хвиля порушує у країв перешкоди елементарні хвилі, що сходяться позаду перешкоди. Таким чином хвиля проникає в область геометричної тіні. Якщо розмір перешкоди набагато більший за довжину хвилі, то звукова хвиля відбивається від такої перешкоди. Якщо ж розміри перешкоди порівняні з довжиною хвилі або менші за неї, то звукова хвиля дифрагує.

Резонанс. У загальному випадку, резонанс — це ефект різкого зростання амплітуди вимушених коливань певної пружної системи при наближенні або повному збігу частоти вимушених коливань з власною частотою цієї системи.

Вимушені коливання системи викликаються дією на неї періодичних зовнішніх сил. Вимушені періодичні коливання в пружному звуковому середовищі можуть створювати будь-які тіла, що здійснюють періодичні механічні коливання (мембрана, динамік, струна тощо).

Власна частота деякої системи — це частота вільного коливання цієї системи. Вільними коливаннями називаються такі коливання, які виникають в пружній системі унаслідок якого-небудь одноразового початкового відхилення системи від стану стійкої рівноваги.

Резонанс звуку може бути бажаним і небажаним явищем. Наприклад, телефонні і мікрофонні мембрани можуть коливатися на різних частотах, але при цьому розробники прагнуть уникнути резонансу (резонанс повинен бути поза бажаною областю частот, інакше резонансні частоти відтворюватимуться надмірно голосно). А ось в акустиці, при прослуховуванні шумів, що створюються різними об'єктами, чи налаштуванні різної радіоапаратури, чи

виборі потрібної частоти в радіоприймачі явище резонансу створюється спеціально.

Реверберація. Розповсюдження звукових хвиль в закритих приміщеннях істотно відрізняється від їх розповсюдження на відкритому просторі. У приміщенні всі перераховані явища можуть робити вплив на результуючий звук. Проте, найбільший вплив роблять відбиття і поглинання, коли хвилі відбиваються то від однієї, то від іншої стінки приміщення (стелі, підлоги). Відбиття звукових коливань можуть сильно впливати на кінцеве сприйняття звуку: вони можуть змінювати забарвлення звуку, насиченість, глибину. Так, звук, що йде від джерела, розташованого в закритому приміщенні, багато разів ударяючись і відбиваючись від стін приміщення, сприймається слухачем як звук, що супроводжується специфічним гулом. Такий гул називається реверберацією.

Теоретично, якби стіни, підлога та стеля зовсім не поглинали звукові коливання і повністю відбивали б їх, то реверберація наростала б нескінченно. Проте на практиці, завдяки ефекту сильного поглинання при відбитті звукової хвилі від твердої стіни, а також з огляду на те, що кожне відбиття звукової хвилі зменшує енергію, час реверберації є кінцевим, а гучність реверберації не піднімається вище за деяке значення.

Часом реверберації називається час, протягом якого гучність звукового сигналу падає на 60 децибел щодо її первинного значення. При цьому об'ємна щільність енергії звукових хвиль зменшується в 106 разів в порівнянні з її первинним значенням. Час реверберації характеризує згасання звуку в закритих приміщеннях після припинення дії джерела звуку.

Відбиття звуку прийнято ділити на ранні відбиття і власне реверберацію. Ранніми відбиттями називають повторення прямого звуку, що прийшли до слухача протягом перших 50 мс. Решта відбиття приходить до слухача багато разів накладеними один на одного і складають той самий гул реверберації.

1.2. Вибір аудіо формату для аналізу звуку

При вирішенні задачі розпізнавання звуку сам процес має відбуватися в режимі реального часу. Це передбачає багаторазове повторення трьох операцій:

1. Запис порції звукових коливань.
2. Обробка отриманих значень.
3. Виведення результату.

Для зниження часу аналізу звукових даних необхідний аудіо формат, який дозволить максимально швидко обробляти дані. Формат представлення звукових даних в цифровому вигляді залежить від способу квантування аналогово-цифровим перетворювачем. Виділяються наступні групи аудіо форматів:

- аудіо формати без стиснення (wav, aiff);
- аудіо формати із стисненням без втрат (ape, flac);
- аудіо формати, із застосуванням стиснення з втратами (mp3, ogg).

Для швидкісного і точного аналізу формати із стисненням і втратами непридатні. Наявність втрат неприйнятно, оскільки знижується точність одержуваних результатів, а необхідність розпакування даних збільшує час їх обробки. Отже, необхідно використовувати формат без компресії.

Аналіз показав, що всю необхідну інформацію в зручному для обробки вигляді містить формат WAVE (wav). Аудіо файл wave складається з двох блоків. Перший блок — інформаційний заголовок файлу, в якому міститься наступна інформація:

- розмір файлу;
- кількість каналів;
- частота дискретизації;
- глибина звучання (швидкість).

Другий блок складається з даних, що зберігають цифровий сигнал — набір значень амплітуд. Звук складається з коливань, які при оцифруванні набувають ступінчастого вигляду. Це обумовлено тим, що комп'ютер може відтворювати в будь-який короткий проміжок часу звук певної амплітуди (гучності) і цей

короткий момент не нескінченно короткий. Тривалість цього проміжку і визначає частота дискретизації. Сукупність амплітуди і короткого проміжку часу носить назву семпл. Амплітуда виражається числом, яке може займати в файлі 8, 16, 24, 32 біт (або більше).

Таким чином, чим більше місця зарезервовано в пам'яті під числову характеристику амплітуди, тим більший діапазон значень можна записати. При розробці програмної реалізації важливо враховувати важливий момент, що в одноканальному аудіо файлі значення амплітуди розташовані послідовно. Як приклад, в стерео звуці спочатку йде значення всіх амплітуд для лівого каналу, потім для правого каналу.

Таблиця 2.

Структура wave файлу

Номер байту	**Поле**	**Опис**
0..3 (4 байти)	chunkId	Містить символи “RIFF” в ASCII кодуванні
4..7 (4 байти)	chunkSize	Розмір файлу
8..11 (4 байти)	format	Містить символи “WAVE”
12..15 (4 байти)	subchunk1Id	Містить символи “fmt”
16..19 (4 байти)	subchunk1Size	16 для формату для імпульсно-кодової модуляції
20..21 (2 байти)	audioFormat	Аудіо формат
22..23 (2 байти)	numChannels	Кількість каналів
24..27 (4 байти)	sampleRate	Частота дискретизації
28..31 (4 байти)	byteRate	Кількість байт, переданих за секунду відтворення
32..33 (2 байти)	blockAlign	Кількість байт для одного семплу, включаючи всі канали
34..35 (2 байти)	bitsPerSample	Глибина звучання
36..39 (4 байти)	subchunk2Id	Містить символи “data”
40..43 (4 байти)	subchunk2Size	Кількість байт в області даних
44..	Data	WAV-дані

Як випливає з таблиці 2, аудіо формат wav дозволяє записувати сигнал

одразу по декількох каналах і з різною частотою дискретизації. Людський слух сприймає звуки з частотою дискретизації не більше 22 кГц, тому робоча частота повинна знаходитися в тих же межах. Крім того недоцільно орієнтувати підсистему на обробку сигналів декількох каналів, оскільки в більшості обчислювальних пристроїв з вбудованими можливостями звукозапису застосовуються одноканальні мікрофони із середньою якістю запису, а використання декількох каналів збільшить час обробки.

Перші чотири байти в файлі — назва основної частини “RIFF”. Всі сигнатури записані в кодуванні ANSII. Наступні 4 байти (іменовані найчастіше як “chunkSize”) вказують на розмір ланцюжка, починаючи з цієї позиції. Чотири наступних байти задають формат RIFF-файлу. Так як ми розглядаємо WAVE файл, то відповідно в цьому полі записано “WAVE” в кодуванні ANSII. Чотири байти “subchunk1Id” — назва нової частини “fmt”, розмір якої задають наступні за нею чотири байти (“subchunk1Size”). Наступні два байти (“audioFormat”) задають ступінь стиснення. Якщо там число, відмінне від одиниці, стиснення файлу було проведено. Наступні два байти (“numChannels”) - кількість каналів (1 — моно звук, 2 — стерео звук і так далі). Наступні два байти (“sampleRate”) — частота дискретизації. Наступні два байти (“byteRate”) — кількість байт, переданих за секунду відтворення. Наступні два байтаи (“blockAlign”) — кількість байт для одного семплу, включаючи всі канали. І два байти (“bitsPerSample”) — кількість біт в семплі, “глибина” звучання. Чотири байти — назва наступного блоку “data”, розмір якого задають наступні чотири байти (“subchunk2Size”). Наступні байти — звукові дані.

Якщо порахувати кількість байт до звукових даних і скласти їх з кількістю байтів, відведених під звукові дані, то можна помітити, що в ряді випадків може виявитися недолік в 250-300 байтів до розміру файлу. Це відбувається через те, що в файлі також зберігаються додаткові дані, які не важливі при відтворенні. Наприклад, автора, дату створення файлу, хто створив файл, жанр і так далі. Важливий момент полягає в тому, що різні програми форматування розташовують ці дані в різних місцях. Таким чином, визначені наступні

параметри аудіо файлу:

- одноканальний режим запису (моно);
- 16 бітове квантування;
- частота дискретизації 22050 Гц;

1.3. Ключові особливості та недоліки алгоритмів роботи зі звуком минулих років

Існує певна кількість алгоритмів і методів для розпізнавання параметризованого звукового сигналу. [2] Залежно від основного призначення системи розпізнавання доцільно використовувати той, чи інший алгоритм.

Найпопулярнішими методами для розпізнавання є:

- використання алгоритму динамічної трансформації часової шкали;
- прихована марковська модель;
- векторне квантування;
- метод опорних векторів;
- використання нейронних мереж;
- модель гаусових сумішей.

Dynamic Time Warping (DTW) — метод динамічної трансформації часової шкали, що дає змогу знайти близькість між двома послідовностями вимірювань за деякий проміжок часу. У загальному випадку ці послідовності можуть бути різної довжини, і вимірювання можуть проводитися з різною швидкістю [3]. Алгоритм DTW обчислює матрицю розмірності (M × N), де M — кількість векторів-характеристик образу сигналу, що розпізнається, і N — кількість векторів еталону. Матричний елемент D (i, j) — є оцінкою глобального шляху вирівнювання до точки (i, j). Цей метод був основною технікою порівняння часових рядів та застосовується для розпізнавання мови та слів з 1970-х років. Основною перевагою алгоритму DTW є простота реалізації. Завдання методу динамічної трансформації часової шкали полягає у створенні метрики відстані між двома вхідними часовими рядами. Подібність або неподібність двох часових рядів, як правило, обчислюється шляхом перетворення даних у вектори та

обчислення евклідової відстані між цими точками у векторному просторі.

Рішення розподілу звукових сигналів (тобто часових рядів) на рівні частини фактично полягає у розділенні проблеми на підзадачі. Коли ми знайдемо оптимальний шлях (найкоротший шлях) для кожної підзадачі, ми розробимо оптимальне рішення для нашої основної проблеми. DTW створює зсув у часі і відображає кожен елемент серії до найближчого елемента іншої серії. Іншими словами, DTW знаходить оптимальну відстань (тут найкраща відстань — це найкоротша відстань) між елементами, виконуючи це відображення. Тож буде створено часовий зсув (рис. 1.5).

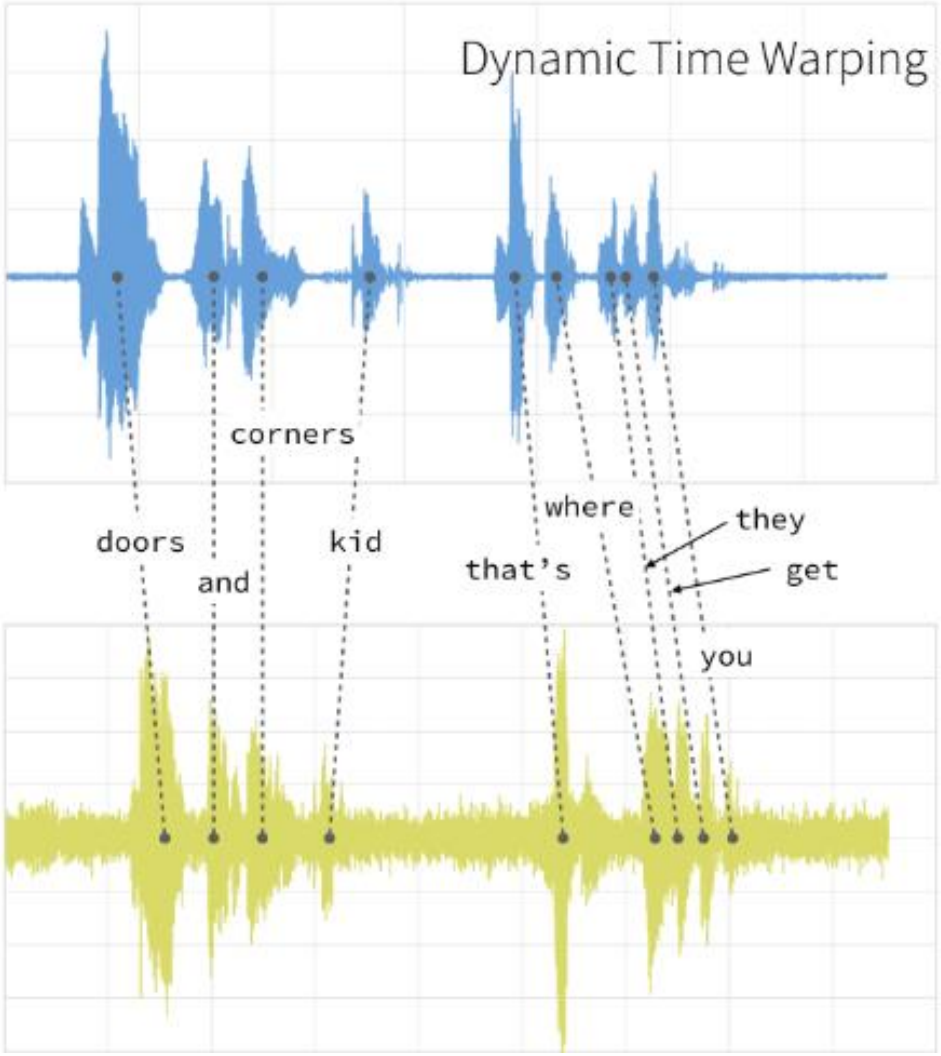


Рис. 1.5. Часовий зсув із DTW

Ці відстані між точками зберігаються в таблиці. Це називається меморизацією. Потім додаються найкоротші шляхи, і отримаємо показник

подібності між двома часовими рядами (рис. 1.6).

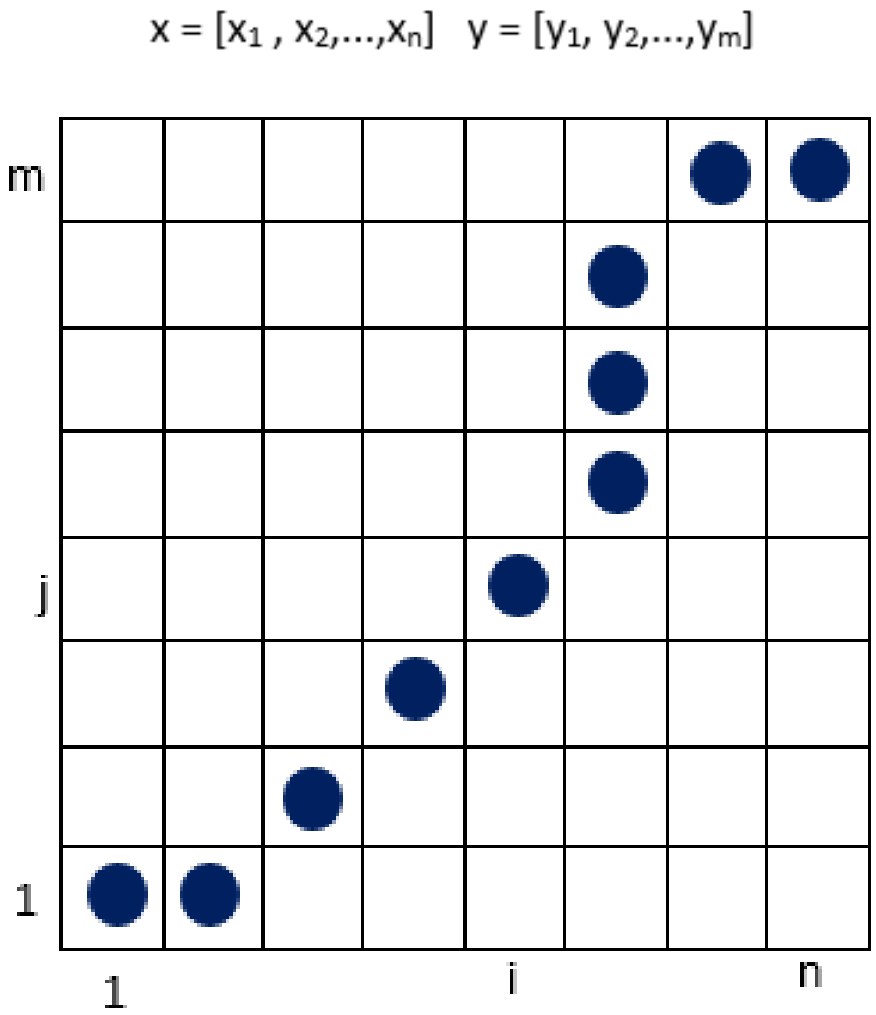


Рис. 1.6. Шлях викривлення

Далі утворюється шлях, який називається деформаційним шляхом. Час переміщується відповідно до цього шляху, тому два ряди досягнуть однакових рівнів часу. У міру зменшення шляху викривлення подібність між двома часовими рядами зростає. Шлях викривлення існує за деякими правилами. Функція викривлення представляє ці правила, і застосовується до обох серій. Ця функція містить деякі обмеження: монотонність, безперервність, граничні умови та вікно деформації. Завдяки цим обмеженням шляхи, які слід випробувати, обмежені.

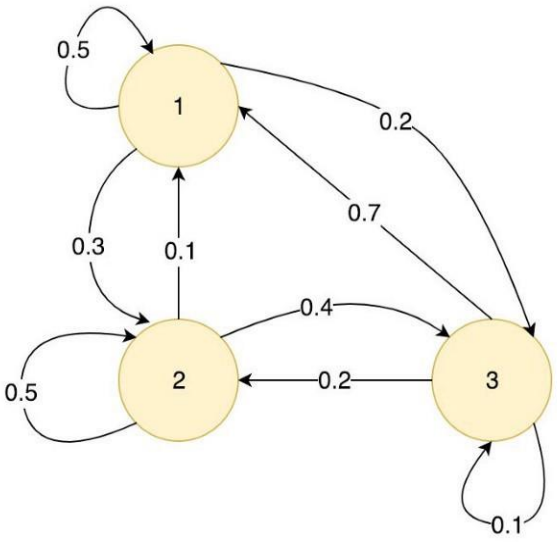
Скажімо, є голосові записи двох різних людей, які говорять одне й те саме речення. Вони говорили однакові слова в різний час. Було б доречним застосувати алгоритм DTW для вирішення часового зсуву. Звуковий сигнал

генерує часовий ряд, тож, щоб виміряти подібність між двома часовими рядами, потрібно зробити наступне:

1. Поділити обидва часові ряди на рівні частини.
2. Порівняти одну точку часового ряду з кожною точкою інших часових рядів і зберегти відстані в таблиці.
3. Виконати крок 2 для кожної точки часового ряду.
4. Потім виконати кроки 2 та 3 для другого часового ряду.
5. Створити деформаційний шлях.
6. Додати всі мінімальні відстані. Це міра схожості між двома часовими рядами.

Hidden Markov Model (HMM) — прихована марковська модель — статистична модель, яка може використовуватися для вирішення задачі класифікації прихованих параметрів на основі спостережуваних даних. HMM — це кінцевий автомат, в якому переходи між станами здійснюються з певною ймовірністю, і задано стартовий стан, з якого починається процес. Через дискретні моменти часу може здійснюватися перехід в нові стани. При цьому кожному прихованому стану з заданою ймовірністю відповідає стан, що спостерігається. Крім того, поточний стан автомата залежить тільки від кінцевого числа попередніх, а закон зміни станів не змінюється в часі [4]. HMM мають досить високу точність розпізнавання.

Ланцюг Маркова (рис. 1.7) містить усі можливі стани системи та ймовірність переходу з одного стану в інший. Ланцюг Маркова першого порядку передбачає, що наступний стан залежить лише від поточного стану.



$$\begin{array}{c} \\ S_1 \\ S_2 \\ S_3 \end{array} \begin{array}{c} \begin{array}{ccc} S_1 & S_2 & S_3 \end{array} \\ \begin{bmatrix} 0.5 & 0.1 & 0.7 \\ 0.3 & 0.5 & 0.2 \\ 0.2 & 0.4 & 0.1 \end{bmatrix} \end{array}$$

Рис. 1.7. Ланцюг Маркова

$$\mathrm{P}(X_{n+1} = x \mid \underbrace{X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n}_{\text{can be ignored}}) = \mathrm{P}(X_{n+1} = x \mid X_n = x_n)$$

З цією моделлю буде набагато легше впоратися. Однак у багатьох системах машинного навчання не всі стани можна спостерігати, і їх називають прихованими або внутрішніми станами. Ймовірність спостереження спостережуваного з урахуванням внутрішнього стану називається ймовірністю викидів. Ймовірність переходу з одного внутрішнього стану в інший називається ймовірністю переходу.

Vector Quantization (векторне квантування) – розбиття простору можливих значень векторної величини на кінцеве число областей. Цей метод обробки сигналу дає змогу моделювати ймовірність функції розподілу векторів. Спочатку цей метод використовувався для стиснення даних. Він працює шляхом поділу великого набору векторів на групи, що мають приблизно однакові значення.

Основна мета стиснення даних — зменшити швидкість передачі даних або зберігання даних при збереженні необхідної точності даних. Вектор ознак може представляти безліч різних можливих кодувань мовних параметрів, включаючи коефіцієнти лінійного прогнозного кодування. У методі векторного квантування вибірка з навчальних векторів перетворюється у фіксовану множину кодових векторів. Одним з поширених методів формування подібної множини, званого

також кодовою книгою, є алгоритм К-середніх. Алгоритм К-середніх розбиває вихідну множину на К кластерів, де К — попередньо задане число. Для цього, спочатку значення середніх ініціюються деякими векторами з вихідної множини. Потім на кожній ітерації алгоритму відбувається розподіл векторів в найближчі до них кластери (для цього обчислюється відстань між вектором і поточними значеннями середніх) і перерахунок середнього в кожному кластері. Алгоритм завершується після того, як на черговій ітерації стани кластерів не змінилися або після досягнення заданої максимальної кількості ітерацій. Метод векторного квантування простий в реалізації, проте не завжди дає високу точність розпізнавання.

Support vector machine (метод опорних векторів) – метод полягає в побудові оптимальної поділяючої гіперплощини. Під оптимальною розуміється гіперплощина, яка перпендикулярна найкоротшому відрізку, що з'єднує опуклі оболонки різних класів, і проходить через середину цього відрізка. Іншими словами, оптимальна гіперплощина повинна максимізувати ширину поділяючої смуги між класами. Метод опорних векторів став популярним інструментом у багатьох видах машинного навчання. SVM базується на статистичній теорії навчання та мінімізації структурних ризиків та є відносно новим перспективним методом вивчення розділових функцій при розпізнаванні образів (класифікаційні) завдання та представляє нові методи навчання.

У найпростіших завданнях розпізнавання використовується лінійне розділення для створення класифікатора з максимальним запасом. Для цього поставлена задача розглядається як обмежена нелінійна задача оптимізації. У випадках, коли дані класи неможливо лінійно розділити у вихідному вхідному просторі, SVM спочатку нелінійно перетворює вхідний простір у більш вимірний простір об'єктів. Це перетворення може досягатись за допомогою різних нелінійних відображень: поліноміальних, сигмоподібних, як у багатошарових персептронах, відображення RBF, що мають в якості основних функцій радіально-симетричні функції, такі як сплайн-функції.

Нейронні мережі. У наш час стали дуже поширеними додатки, що використовують алгоритми штучних нейронних мереж. Це пов'язано з величезним технічним розвитком комп'ютерів і обчислювальної техніки загалом, особливо в останні десятиліття. Загалом класифікація аудіо даних за допомогою методів машинного навчання є добре вивченою задачею. Запропоновано ряд методів класифікації різних типів звуків, музики та мови. Було опубліковано багато підходів до класифікації, заснованих на тих алгоритмах, які мають ряд переваг порівняно зі статистичними класифікаторами. Цей тип класифікаторів сильно залежить від розподілу статистичних даних, тоді як нейромережеві класифікатори можуть оцінювати нелінійний зв'язок між вхідними та бажаними вихідними даними. Більше того, вхідні дані можуть бути пошкодженими або неповними; часто рішення проблеми може бути настільки складним, що її опис неможливий. У цьому випадку використання штучної нейронної мережі є практично єдиною можливістю.

Нейронні мережі поділяються на синхронні і асинхронні. У синхронних нейронних мережах в кожен момент часу свій стан змінює лише один нейрон. В асинхронних — стан змінюється одразу у цілої групи нейронів, як правило, у всього шару. Можна виділити дві базові архітектури — шаруваті і зв'язні мережі. Ключовим в шаруватих мережах є поняття шару.

Шар — один або кілька нейронів, на входи яких подається один і той же загальний сигнал.

Шаруваті нейронні мережі — нейронні мережі, в яких нейрони розбиті на окремі групи так, що обробка інформації здійснюється пошарово.

У шаруватих мережах нейрони і-го шару отримують вхідні сигнали, перетворюють їх і через точки розгалуження передають нейронам (i + 1) шару. І так до k-го шару, який видає вихідні сигнали для інтерпретатора і користувача. Число нейронів в кожному шарі не пов'язане з кількістю нейронів в інших шарах, може бути довільним.

В рамках одного шару дані обробляються паралельно, а в масштабах всієї мережі обробка ведеться послідовно — від шару до шару. Однак сигнал не

завжди подається на всі нейрони прошарку. У когнітроні, наприклад, кожен нейрон поточного шару отримує сигнали тільки від близьких йому нейронів попереднього шару. Шаруваті мережі в свою чергу, можуть бути одношаровими і багатошаровими. У багатошаровій мережі перший шар називається вхідним, наступні — внутрішніми або прихованими, останній шар — вихідним. Таким чином, проміжні шари — це всі верстви в багатошарової нейронної мережі, крім вхідного і вихідного. Вхідний шар мережі реалізує зв'язок з вхідними даними, вихідний — з вихідними. Таким чином, нейрони можуть бути вхідними, і прихованими.

Вхідний шар організований з вхідних нейронів, які отримують дані і поширюють їх на входи нейронів прихованого шару мережі.

Прихований нейрон — це нейрон, що знаходиться в прихованому шарі нейронної мережі.

Вихідні нейрони, з яких організовано вихідний шар мережі, видає результати роботи нейронної мережі.

У зв'язних мережах кожен нейрон передає свій вихідний сигнал іншим нейронам, включаючи самого себе. Вихідними сигналами мережі можуть бути всі або деякі вихідні сигнали нейронів після кількох тактів функціонування мережі. Всі вхідні сигнали подаються всім нейронам.

Найбільш часто використовуваним методом класифікації за допомогою нейронної мережі є багатошаровий персептрон. Існує ряд досліджень, де штучна нейронна мережа була застосована для класифікації різних аудіо класів, особливо з охоронних та військових районів, таких як різні постріли та вибухи.

Gaussian Mixture Model (модель гаусових сумішей) представляє собою параметричну функцію щільності ймовірності. Модель зручна для моделювання характеристик каналу звукозапису, навколишнього середовища. Існує певний ряд методів для оцінки параметрів моделі. Одним з найбільш популярних і добре себе зарекомендованих є метод оцінки максимальної правдоподібності [5]. Мета оцінки за даним методом полягає у визначенні параметрів моделі, які максимально підвищать ймовірність правдоподібності моделі при заданих даних

для навчання. Модель являє собою ефективний алгоритм, який дає змогу проводити ідентифікацію з високою точністю розпізнавання [6]. Однак виникає ряд проблем, які пов’язані з вибором числа компонентів моделі та ініціалізацією її початкових параметрів.

Коли сигнали обробляються в стиснуті величини перетворення Фур'є або відповідні кепстри, модель гаусових сумішей виявляється цілком доречною для таких особливостей звуку, її можна використовувати як модель для представлення звукових особливостей, моделювання даних та статистичної класифікації. Класифікатори, засновані на моделі гаусових сумішей, є високоефективними та широко поширеними в мовленнєвих дослідженнях, в першу чергу для розпізнавання мовців, амортизації мови та розпізнавання мови.

1.4. Алгоритми, що працюють сьогодні

Якщо аналізувати більш сучасні алгоритми розпізнавання, що мають більш прикладний характер і використовуються для покращення рівня безпеки громадян, варто відзначити систему ShotSpotter [11], лідер в США при розпізнаванні пострілів з вогнепальної зброї в міських умовах. У найбільш небезпечних районах міст встановлені спрямовані мікрофони (на стовпах, будинках, високих спорудах), що вловлюють звуки міського фону. У разі ідентифікації пострілу, інформація з GPS-координатами конкретного мікрофона передається на центральний комп'ютер, де проводиться додатковий аналіз звуку, щоб відсіяти можливі помилкові спрацьовування, наприклад, звук вертольоту чи вибух петарди. Якщо постріл підтверджується, патруль виїжджає на місце розташування джерела звуку. Система, встановлена в Вашингтоні ще в 2006 році, за минулі роки локалізувала 39000 пострілів з вогнепальної зброї, і поліція змогла швидко відреагувати в кожному конкретному випадку.

На рисунку 1.8. зображена схема принципу роботи системи ShotSpotter за

участі таких компонентів:

1. Постріл з вогнепальної зброї.
2. Сенсори, що приймають звукові хвилі від пострілу.
3. Камери, що фіксують джерело звуку на відео.
4. Командний центр, куди передається інформація.
5. Патруль поліції, що виїжджає на місце пострілу.

Рис. 1.8. Ілюстрація роботи системи ShotSpotter

Проект “AudioAnalytics”, що базується у Великобританії, надає одразу кілька рішень для різних варіантів використання алгоритму. Архітектура запропонованих рішень має наступний вигляд: програма CoreLogger, що працює на пристрої кінцевого користувача, дозволяє приймати і відображати, або зберігати тривожні події. Працює в зв'язці з іншою частиною загальної системи — Sound Packs — яка представляє не що інше як набір різних модулів для аудіо аналітики.

Основні можливості даних модулів представляють собою розпізнавання наступних аудіо подій:

- агресія (розмова на підвищених тонах, крик);
- сигналізація автомобілів;
- розбите скло;
- пошук ключових слів ("поліція", "допоможіть" і т. д.);
- постріли;
- крик / плач дитини.

Додатково надається частина системи під назвою Core Trainer, яка на підставі поданого їй на вхід набору аудіо сигналів, виділить найбільш унікальні частини і сформує новий патерн для SoundPacks.

Висновки

Існує досить велика кількість методів та алгоритмів розпізнавання голосових сигналів. Всі вони мають свої переваги та недоліки. Використання того чи іншого алгоритму сильно залежить від напряму використання системи розпізнавання, наявних ресурсів, розміру словника, залежності від диктора.

Основними задачами при розпізнаванні голосових сигналів є:

1) Параметризація вхідного сигналу — виділення придатних для аналізу параметрів з вхідного звукового сигналу.

2) Аналіз параметризованого сигналу — пошук відповідності між вхідним голосовим сигналом та реальним вимовленим словом.

Звук – це хвиля, що розповсюджується в пружному середовищі. Джерелом виникнення хвилевого руху (джерелом звуку) може служити будь-яке тіло, здатне здійснювати пружні коливання. Якщо в середовищі одночасно розповсюджується система декількох різних хвиль, то кожна з хвиль розповсюджується незалежно від інших – явище інтерференції.

На даний час існує певна кількість методів, що дають змогу вирішувати завдання ідентифікації аудіо події, причому кожен із наведених методів має свої

переваги та недоліки. Проте, найбільш поширеним методом є модель гаусових сумішей, що добре себе зарекомендувала в якості стохастичної моделі для побудови систем розпізнавання [9]. Ця модель зручна не тільки для моделювання характеристик звуку, але і каналу звукозапису, навколишнього середовища. Окремі компоненти моделі можуть моделювати окрему множину акустичних ознак. Кожна з компонент моделі відображає як загальні, так і індивідуальні для кожного звукового сигналу особливості.

РОЗДІЛ 2
ЕТАПИ ПРОЦЕСУ РОЗПІЗНАВАННЯ ЗВУКУ

2.1. Детектування та розпізнавання аудіо подій різних типів

В якості прикладу вихідних даних представлені спектрограми трьох різних типів аудіо подій:

1. Масив пострілів (рис. 2.1).
2. Розбите скло (рис. 2.2).
3. Крик (рис. 2.3).

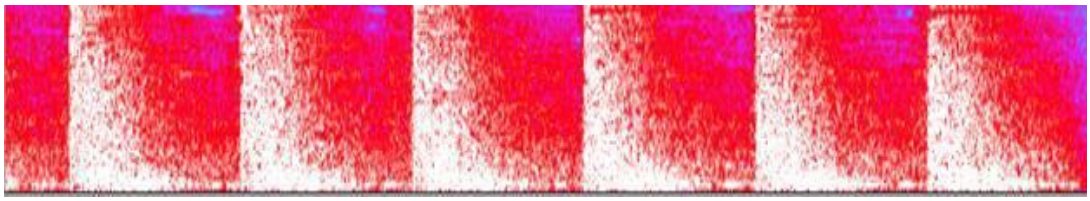


Рис. 2.1. Спектрограма аудіо події “Набір пострілів”

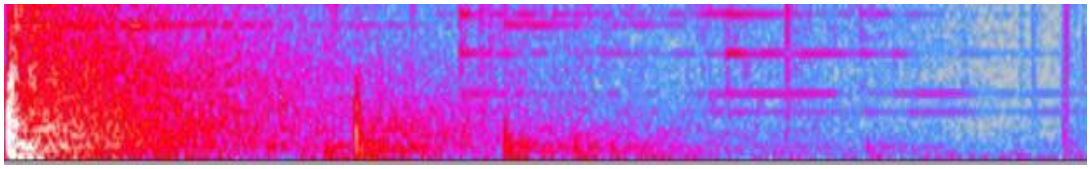


Рис. 2.2. Спектрограма аудіо події “Розбите скло”

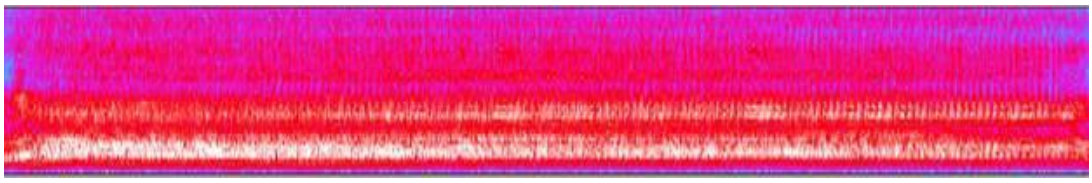


Рис. 2.3. Спектрограма аудіо події “Крик”

Із бази аудіо даних були завантажені приклади аудіо записів цих подій. Таким чином, задача розпізнавання тривожних подій поділяється на дві частини:

1. Виділення різких імпульсних сигналів з фонового шуму в потоці аудіо даних.
2. Класифікація (розпізнання) сигналу і віднесення до одного з заданих типів.

Більшість методів базуються на визначенні потужності для набору послідовних блоків аудіо сигналу. Різні методи відрізняються способом автоматичного виявлення блоку, що відповідає різкому імпульсному звуку:

- на основі стандартного відхилення нормованих значень потужностей блоків;
- на основі застосування медіанного фільтру для значень потужностей блоків;
- за динамічним порогом для значень потужностей блоків;

Основними етапами розпізнавання (рис. 2.4) аудіо подій є:

1. Буферизація з перекриттям.
2. Передобробка.
3. Витяг ознак.
4. Постобробка ознак.
5. Навчання /класифікація.

Рис. 2.4. Схема етапів розпізнавання аудіо подій

Буферизація з перекриттям

На першому етапі відбувається перетворення вихідного аудіо сигналу в набір фреймів з перекриттям (рис. 2.5).

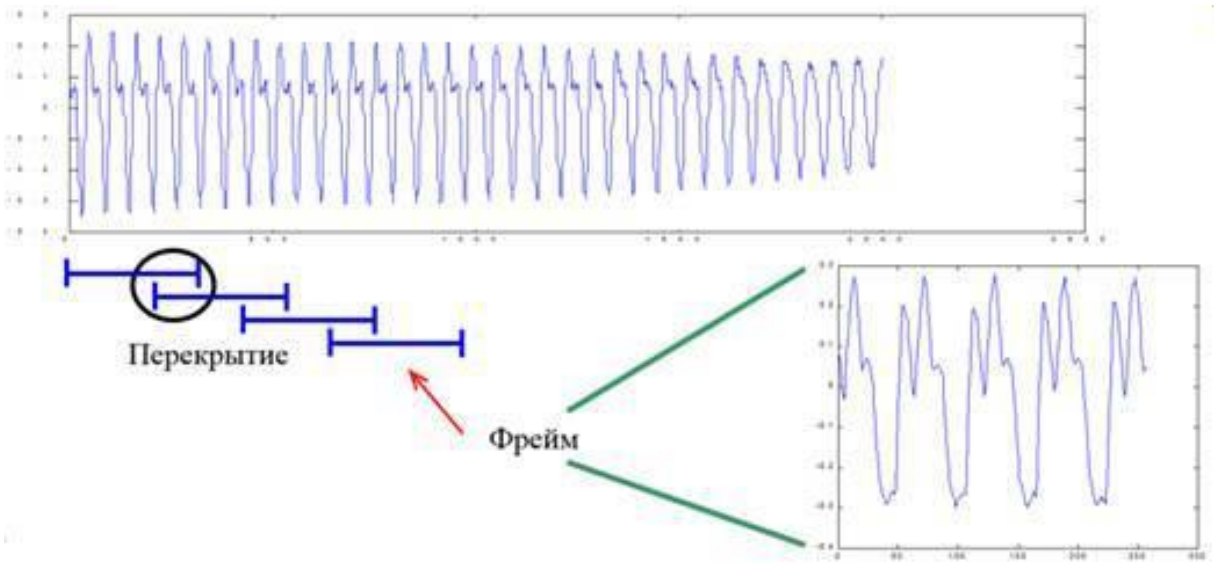


Рис. 2.5. Буферизація з перекриттям

Стадія передобробки

Стадія передобробки включає в себе, як правило, pre-emphasis фільтрацію і віконне зважування. Pre-emphasis обробка здійснюється за рахунок застосування КІХ-фільтра H (z) = 1 a / Z. Це необхідно для спектрального згладжування сигналу [13]. В такому випадку сигнал стає менш сприйнятливим до різних шумів, що виникають в процесі обробки.

Віконне зважування необхідно застосовувати в зв'язку з тим, що аудіо фрейм обмежений в часі, тому при переході в частотну область відбуватиметься ефект просочування спектра бічних пелюсток, пов'язане з формою спектра функції прямокутного вікна (він має вигляд sin (x) / x). Тому, щоб зменшити вплив цього ефекту, застосовується зважування вихідного сигналу різного виду вікнами, з формою, відмінною від прямокутної. Відлік вхідної послідовності множиться на відповідну функцію вікна, що тягне за собою обнулення значень сигналу на краях вибірки. В якості зважених функцій найчастіше виступають вікна Хеммінга, Блекмен, плоске, Кайзеля-Бесселя, Дольфа-Чебишева [14].

Витяг ознак

Існує кілька підходів до вилучення дескрипторів (ознак) з аудіо сигналу. Всі вони задаються спільною метою зменшити надмірність сигналу і виділити найбільш релевантну інформацію, і, в той же час, відкинути нерелевантну. Як правило, ознаки, що описують аудіо сигнал з різних точок зору, комбінуються в один вектор ознак, на основі якого відбувається процес навчання і потім класифікації з використанням обраної навченої моделі. Далі будуть представлені найбільш популярні ознаки, що виділяються з аудіо.

Статистика в тимчасовій області (Time-domain statistics)

Нульова швидкість перетину (рис. 2.6) — кількість перетинів осі часу аудіо сигналом [15].

$$zcr = \frac{1}{T-1}\sum_{t=1}^{T-1}\mathbb{I}\{s_t s_{t-1} < 0\}$$

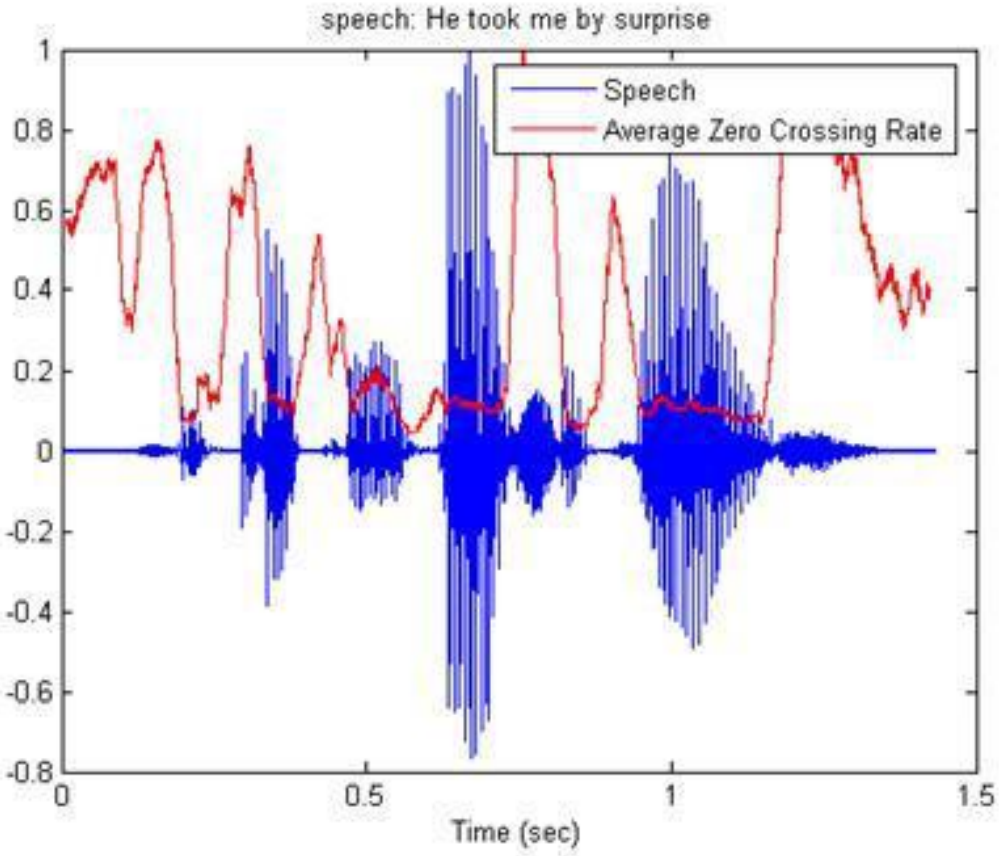


Рис. 2.6. Нульова швидкість перетину

Short-time energy — середнє значення енергії для аудіо фрейму.

$$E_n = \frac{1}{T}\sum_{i=1}^{T} x_i^2$$

Розділивши кожен фрейм на набір під-фреймів, обчислюється набір енергій для кожного з підфреймів.

Статистика в частотній області

Спектральний центроїд – являє собою інтерпретацію “центру мас” спектра. Обчислюється як сума частот, зважених відповідними амплітудами спектра, поділена на суму амплітуд:

$$Spectral\ \ Centroid\ \ = \frac{\sum_{k=1}^{N} kF[k]}{\sum_{k=1}^{N} F[k]}$$

Де F[k] — амплітуда спектра, відповідна k-му значенню частоти в спектрі ДПФ. Потім отримане значення зручно нормувати на максимальне значення частоти (Fs / 2), в результаті діапазон можливих “центр мас” спектра буде лежати в діапазоні [0-1].

Миттєва ширина спектра / пропускна здатність (рис. 2.7) — визначається як другий центральний момент.

$$SS = \frac{\sum_{k} (f_k - SC)^2 F[f_k]}{\sum_{k} F[f_k]}$$

Fk – значення частот в DFT, F[fk] – значення амплітуд, SC – значення спектрального центроїду.

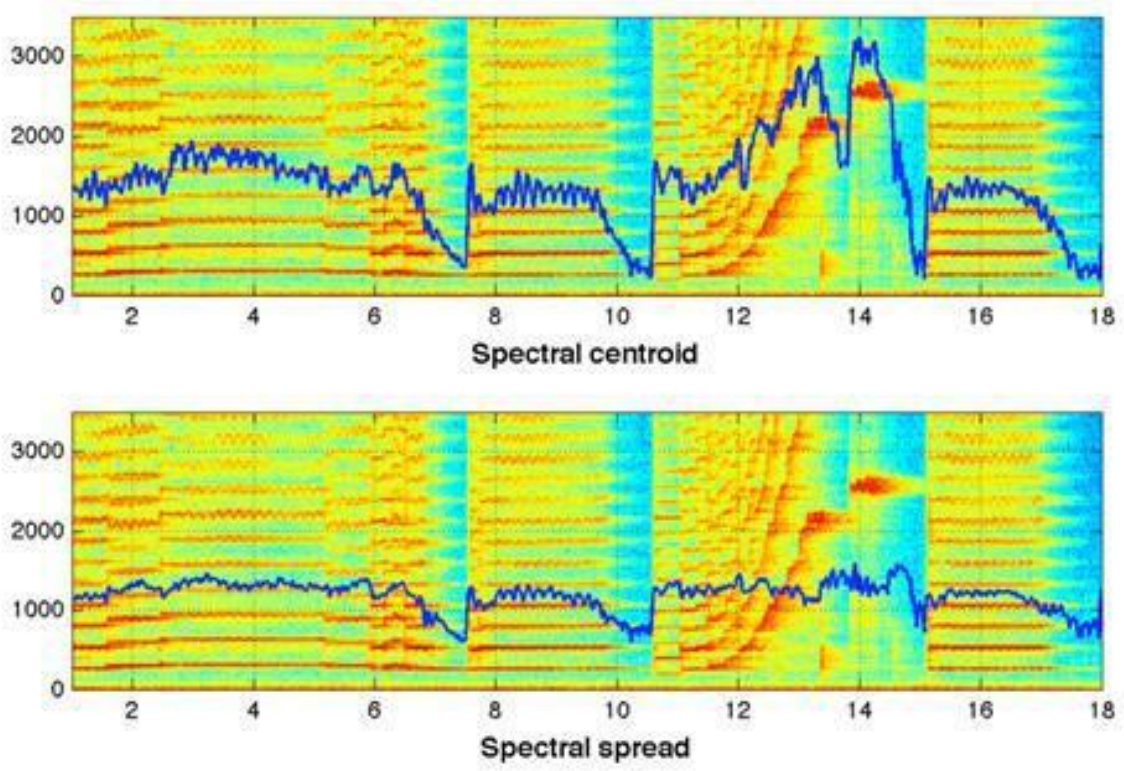


Рис. 2.7. Миттєва ширина спектра

Рівність звукового спектра — відображає відхилення потужності спектра сигналу від пологої форми. З точки зору людського сприйняття, характеризує ступінь тональності звукового сигналу.

Спектр енергетичної смуги

Простір частот розділяється на N смуг, після чого обчислюється енергія спектра в кожній смузі. Отримані значення беруться в якості ознак. По суті ознаки в такому випадку є значення енергії спектра в “низькій роздільній здатності”.

Кепстральні коефіцієнти (рис. 2.8)

Виходять на основі попередніх ознак шляхом переведення їх в кепстральний простір. Кепстр являє собою не що інше як “спектр логарифма спектра”, замість перетворення Фур'є застосовують дискретне косинусне перетворення (ДКП):

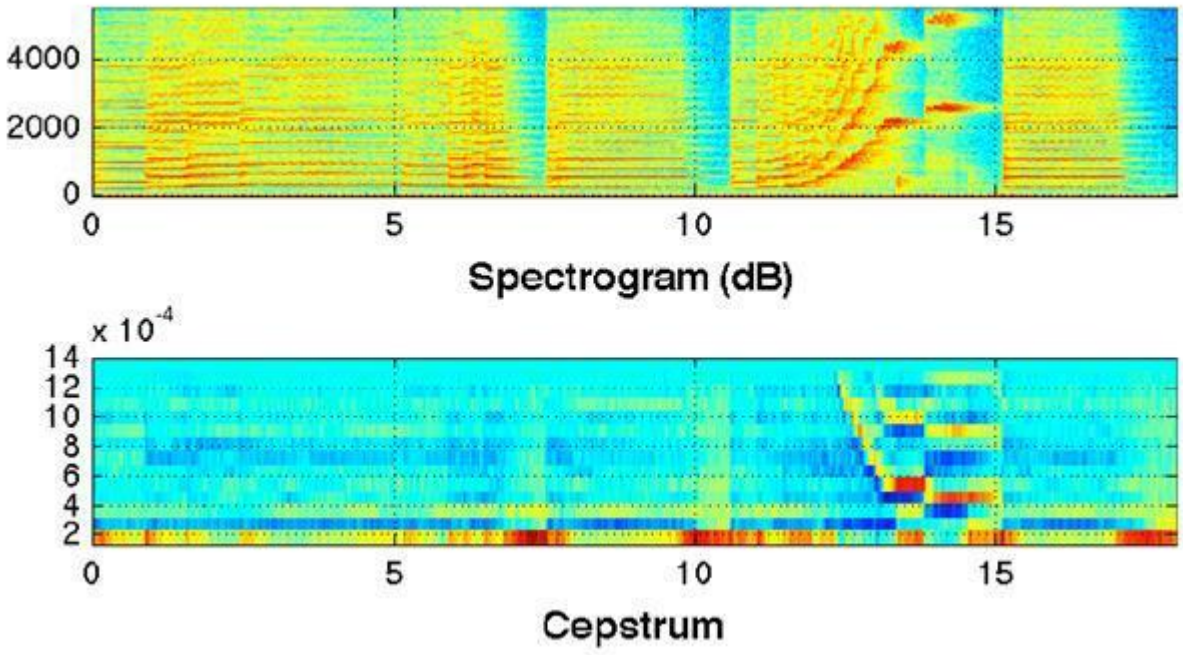


Рис. 2.8. Кепстральні коефіцієнти

MFCC — мел-частотні кепстральні коефіцієнти

Мел — одиниця висоти звуку, заснована на сприйнятті цього звуку людськими органами слуху. Як відомо, амплітудно-частотна характеристика сприйняття звуку людськими органами чуття навіть віддалено не нагадує пряму, а амплітуда — не точна міра гучності звуку. Тому і ввели емпірично вибрані одиниці гучності, наприклад, фон.

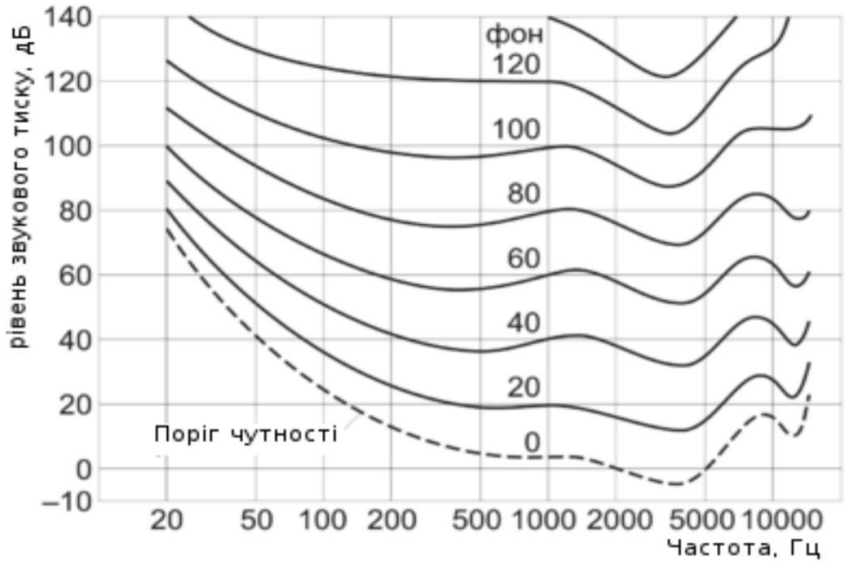


Рис. 2.9. Графік залежності рівня звукового тиску від частоти.

Аналогічно, висота звуку, що сприймається людським слухом залежить не лінійно від його частоти.

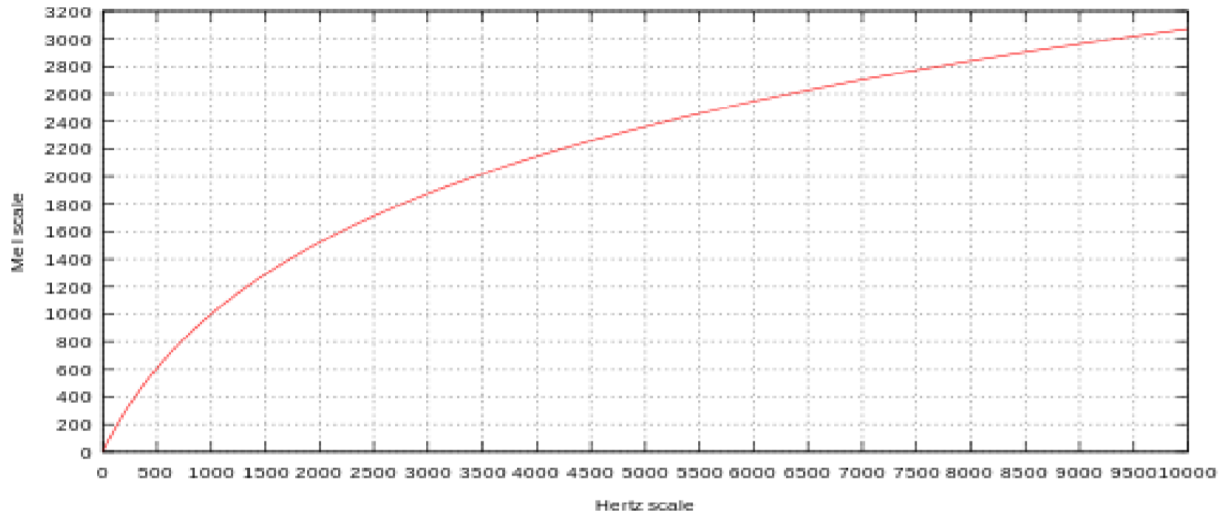


Рис. 2.10. Графік залежності висоти звуку в мелах залежно від частоти коливань

Подібні одиниці виміру часто використовують при вирішенні задач розпізнавання, оскільки вони дозволяють наблизитися до механізмів людського сприйняття, яке поки що лідирує серед відомих систем розпізнавання мови. [17] Відповідно до теорії утворення мови, вона являє собою акустичну хвилю, яка випромінюється системою органів: легкими, бронхами і трахеєю, а потім перетворюється в голосовому тракті. Якщо припустити, що джерела збудження і форма голосового тракту відносно незалежні, мовний апарат людини можна представити у вигляді сукупності генераторів тональні сигнали і шумів, а також фільтрів:

1. Генератор імпульсної послідовності (тонів).
2. Генератор випадкових чисел (шумів).
3. Коефіцієнти цифрового фільтра (параметри голосового тракту).
4. Нестаціонарний цифровий фільтр.

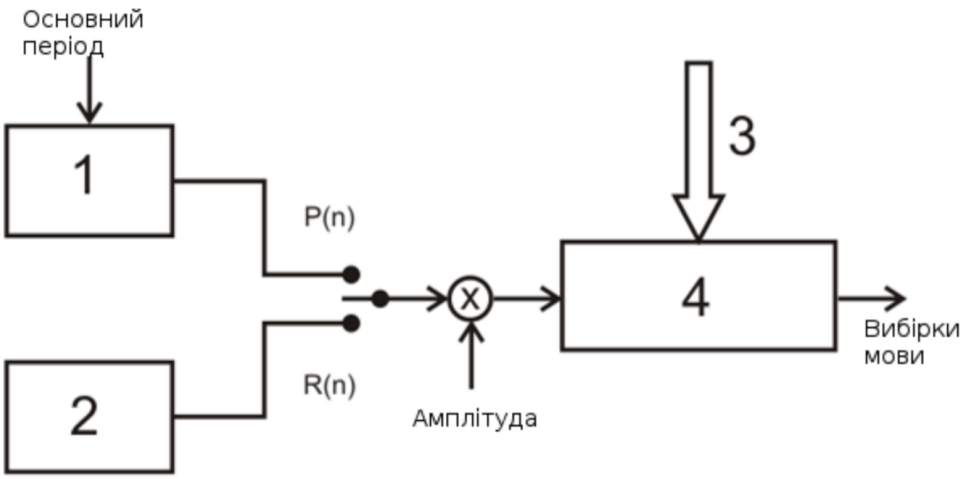


Рис. 2.11. Схема голосового апарату.

Сигнал на виході нестаціонарного цифрового фільтра можна представити у вигляді згортки: $f(t) = s(t) \otimes h(t)$, де s(t) — початковий вигляд акустичної хвилі, а h(t) — характеристика фільтра (залежить від параметрів голосового тракту); У частотній області: $F(w) = S(w) \times H(w)$. Добуток можна прологарифмувати, щоб отримати замість нього суму. Потім потрібно перетворити цю суму так, щоб отримати непересічні набори характеристик вихідного сигналу і фільтра.

В залежності від цілей можна використовувати пряме перетворення Фур'є або дискретне косинусне перетворення. Отже, кепстр — це енергетичний спектр (залежність енергії від імпульсу) функції. Після виділення мел-кепстральних коефіцієнтів можна переходити до використання алгоритмів розпізнавання такі як нейронні мережі, або алгоритм динамічної типізації часової шкали. [18]

Аудіо сигнал розподіляється на фрейми, проводиться pre-emphasis фільтрація та віконне зважування, виконується швидке перетворення Фур’є, далі спектр пропускається через набір трикутних фільтрів, розташованих рівномірно на мел-шкалі. Це призводить до більшої щільності фільтрів у області низьких частот і меншої густини у області високих частот, що відображає чутливість сприйняття звукових сигналів людським вухом. Таким чином, основна інформація “знімається” з аудіо сигналу в області низьких частот, що є найбільш релевантною ознакою звукових сигналів (особливо голосових). Далі відліки переводяться в кепстральний простір за допомогою дискретного косинусного перетворення (ДКП).

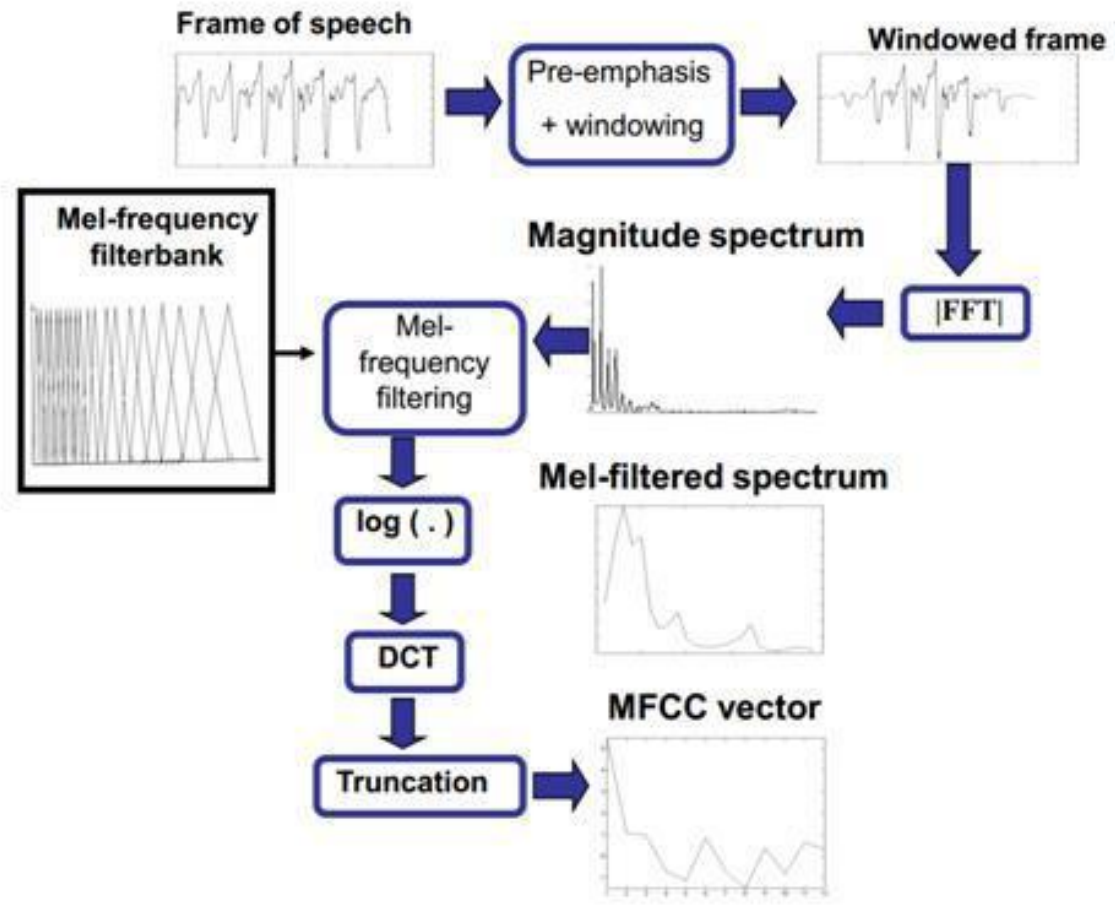


Рис. 2.12. Загальна схема отримання мел-частотних кепстральних коефіцієнтів

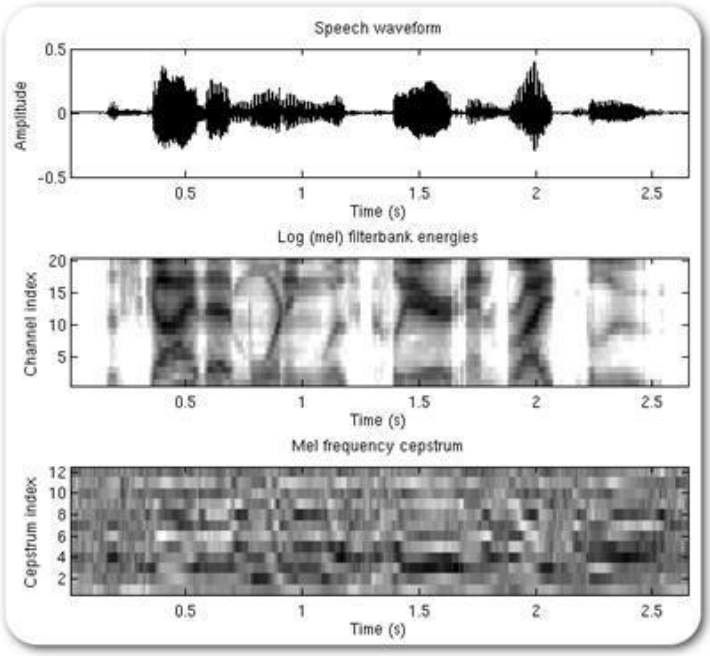


Рис. 2.13. Мел-частотниі кепстральні коефіцієнти

Математичний опис:

Застосовуємо до сигналу перетворення Фур'є

$$X_a[k] = \sum_{n=0}^{N-1} x[n] e^{\frac{-2\pi i}{N} kn}, \;\; 0 \leq k < N$$

Складаємо гребінку фільтрів, використовуючи віконну функцію

$$H_m = \begin{cases} 0 & k < f[m-1] \\ \frac{(k - f[m-1])}{(f[m] - f[m-1])} & f[m-1] \leq k < f[m] \\ \frac{(f[m+1] - k)}{(f[m+1] - f[m])} & f[m] \leq k \leq f[m+1] \\ 0 & k > f[m+1] \end{cases}$$

Для якої частоти f [m] отримуємо з рівності

$$f[m] = (\frac{N}{F_s}) B^{-1}(B(f_1) + m \frac{B(f_h) - B(f_1)}{M+1})$$

B (b) — перетворення значення частоти в мел-шкалу, відповідно,

$$B^{-1}(b) = 700(exp(b/1125) - 1)$$

Обчислюємо енергію для кожного вікна

$$S[m] = ln(\sum_{k=0}^{N-1} |X_a[k]|^2 H_m[k]), \;\; 0 \leq m < M$$

Застосовуємо дискретне косинусне перетворення

$$c[n] = \sum_{m=0}^{M-1} S[m] cos(\pi n(m + 1/2)/M), \;\; 0 \leq n < M$$

Коефіцієнти лінійного прогнозування

В якості ознак виступають коефіцієнти, що прогнозують сигнал на основі лінійної комбінації попередніх відліків. По суті є коефіцієнтами КІХ-фільтра відповідного порядку:

$$s(n) = -\sum_{i=1}^{N_{lp}} a_{N_{lp}}(i) * s(n-i) + e_n$$

2.2. Постобробка ознак

Після вилучення необхідних ознак сигналу для їх подальшого використання проводиться нормалізація ознак так, щоб кожен компонент вектора ознак мав середнє значення 0 і стандартне відхилення 1:

$$\vec{f}_k^{norm}(d) = \frac{\vec{f}_k(d) - \mu_d}{\sigma_d}$$

Часто застосовується техніка mid-term analysis, коли проводиться усереднення ознак за набором послідовних фреймів. Як правило, в якості інтервалу для усереднення обирається час від 1 до 10 секунд. Розмірність вектора ознак виходить досить великою, що істотно впливає в подальшому на продуктивність процесу навчання. Ось чому в цьому випадку застосовуються добре відомі і теоретично вивчені методи скорочення розмірності вектора ознак (LDA, PCA і ін).

Однією з переваг використання методів скорочення розмірності є здатність істотно збільшити швидкість процесу навчання за рахунок зменшення кількості ознак. Більш того, обравши ознаки із найкращими дискримінаційними здібностями в окремий набір, можливо позбутись нерелевантної інформації, що підвищить точність роботи алгоритмів машинного навчання (таких як SVM). Різниця методів PCA і LDA полягає в тому, що метод PCA дозволяє скоротити розмірність вектору ознак за рахунок виділення незалежних компонентів.

2.3. Вибір класифікатора

Нарешті, останнім етапом алгоритму є вибір класифікатора (навчальної моделі). У ряді досліджень в області аудіо аналітики порівнюються точність розпізнавання при використанні різних моделей класифікаторів при різних типах аудіо подій. Відзначається, що використання ієрархічних класифікаторів істотно збільшує точність розпізнавання в порівнянні з використанням мультикласових класифікаторів.

Найбільш простим варіантом моделі розпізнавання є Байєсівський класифікатор, заснований на обчисленні функції правдоподібності для кожного з класів, і на етапі розпізнавання подія відноситься до того класу, ймовірність якого максимальна серед усіх класів.

На відміну від класичного Байєсівського класифікатора, де для апроксимації кожного класу виступали параметри гауссівского розподілу, в якості опції щільності розподілу ймовірності в моделі виступає «суміш» з декількох гауссівских розподілів, параметри яких для кожного класу підбираються на етапі навчання з використанням алгоритму максимізації очікування.

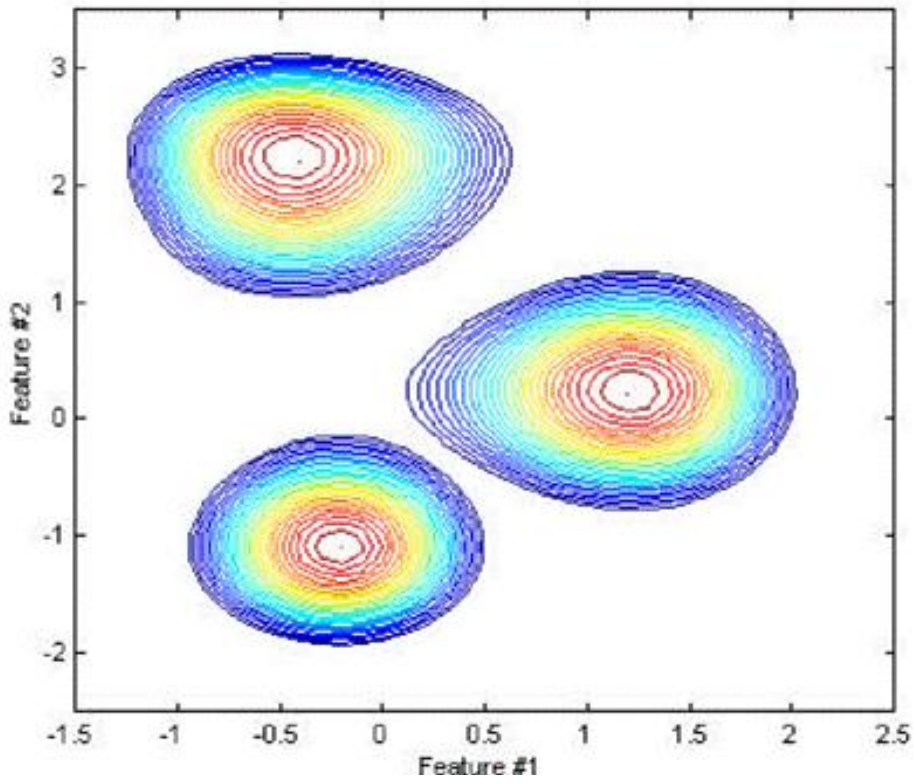


Рис. 2.14. Приклад класифікації трьох різних класів

Серед недоліків даної моделі можна відзначити високу чутливість до варіацій в навчальній вибірці даних при виборі великого числа гауссівских розподілів, що може призводити до процесу перенавчання.

2.4. Аналіз звукових сигналів

Аналіз звукових сигналів виконується з метою виявлення закономірностей в звуковій хвилі, що формується від різних джерел. Ці дані необхідні при проектуванні і оцінці якості апаратури, розробці систем кодування, вибору методів обробки, розпізнаванні мови і в багатьох інших випадках. При цьому використовуються амплітудно-часові і спектральні характеристики.

Звукова хвиля в деякій точці може бути однозначно представлена графіком залежності зміни тиску (або іншої вимірюваної величини), що створюється звуковою хвилею, від часу. Графічне зображення такої залежності називається осцилограмою (рис. 2.15) або сигналограмою.

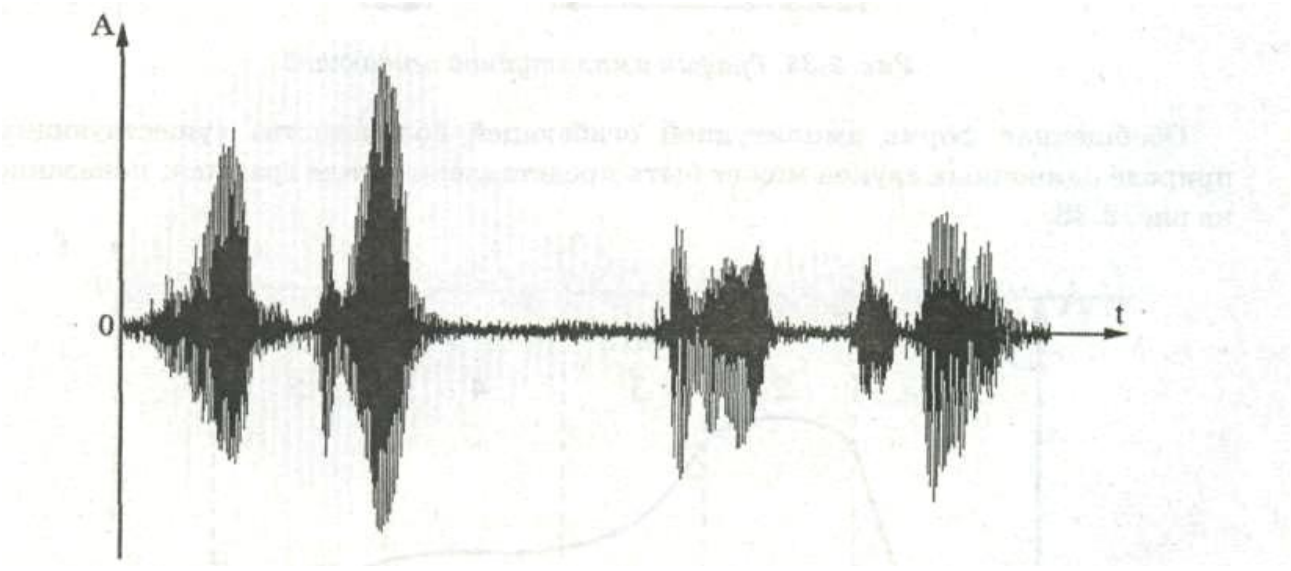


Рис. 2.15. Приклад осцилограми фрази "раз-два-[пауза]-три-чотири"

Простим видом звукових коливань є синусоїдальні (або гармонічні) коливання, які описуються математично за допомогою наступного рівняння:

$y(t) = A*\sin(\Omega t + \varphi)$

де y(t) — умовне позначення вимірюваної фізичної величини (тиск у фронті, напруга і тому подібне);

A — амплітуда коливання, тобто максимальне значення функції;

Ω=2*f/T — кутова (циклічна) частота коливань;

f — частота коливань;

φ — початкова фаза коливань, тобто зрушення по осі абсцис від початку координат функції y(t) у момент t=0.

Звукові сигнали складнішого виду можна характеризувати спектром, тобто набором синусоїдальних звукових хвиль, в результаті накладення яких утворюється результуюча звукова хвиля, співпадаюча з початковою.

Отримання даних про звуковий сигнал у вигляді спектру для подальшого аналізу і обробки набуло дуже широкого поширення, може бути навіть більше, ніж амплітудно-часова залежність. Для переходу до спектрального уявлення використовується перетворення Фур'є.

Сигналограма може бути представлена в дискретному вигляді як послідовність значень амплітуд. При переході до спектру сигнал характеризується набором амплітуд (і фаз) відповідних гармонік.

На рис. 2.16 та рис. 2.17 приведені сигналограма і амплітудно-частотна залежність (спектр) одного і того ж сигналу (музичний фрагмент).

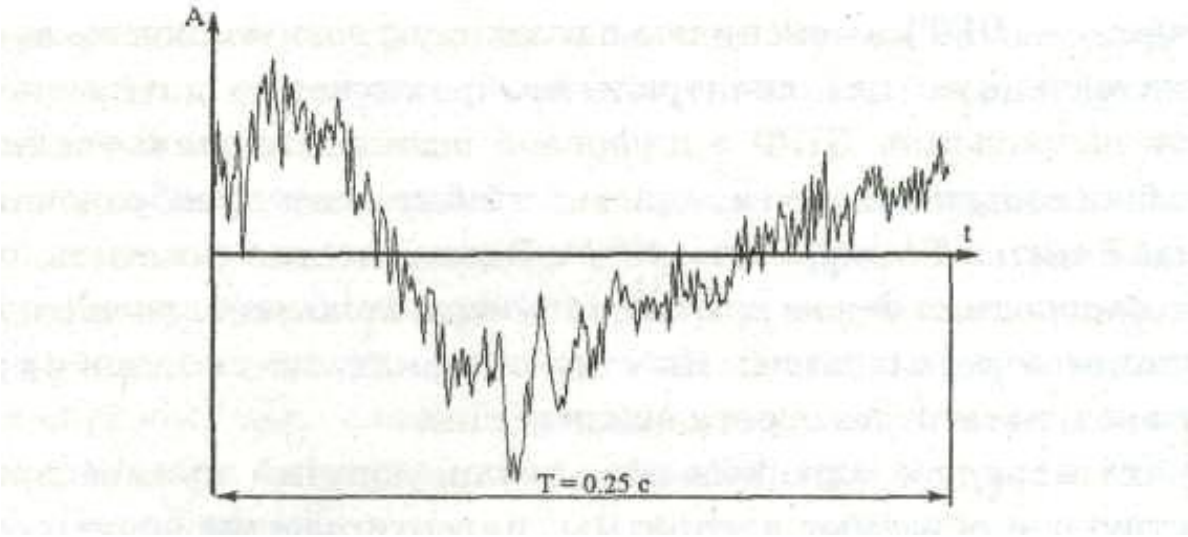


Рис. 2.16. Сигналограма музичного фрагмента

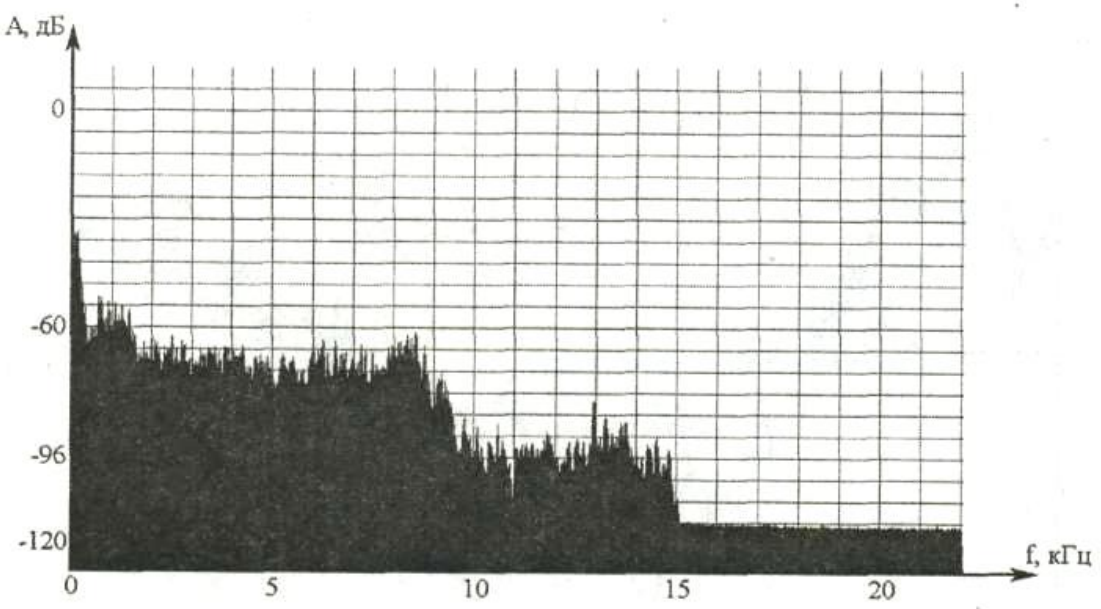


Рис. 2.17. Спектр музичного фрагмента

Якщо проаналізувати процес отримання сигналу із спектру (зворотне перетворення Фур'є), то можна відмітити, що низькочастотні складові спектру додають результуючій хвилі загальну правильність форми, тоді як високочастотні складові уточнюють форму хвилі, привносячи в неї дрібні деталі початкового сигналу.

Слід зазначити, що графічне представлення звукових сигналів грає ілюстративну роль, набагато важливіше порівнювати ці форми уявлення з погляду обробки, процесів, що протікають в реальних звукових системах. І тут необхідно враховувати ряд особливостей. Амплітудно-частотному перетворенню нереально піддати весь звуковий сигнал (наприклад, запис двогодинного концерту). По-перше, це задача дуже великої розмірності, та і обробка протікатиме не в режимі реального часу, а відкладено). По-друге, неможливо судити про динаміку зміни спектру, його розвитку в часі, а це важливо. Тому перетворенню піддається блок (частина сигналу на невеликому інтервалі).

Ідея блокового спектрального аналізу полягає в проведенні гармонічного аналізу реального звукового сигналу таким чином, який дозволив би бачити динаміку зміни спектру аналізованого сигналу в часі. При цьому на отримуваний спектр впливають як вид (форма) аналізованого блоку (чим гладкіша функція,

тобто форма кривої, тим менше високочастотних складових містить спектр), так і інтервал, на якому цей фрагмент (блок) розглядається.

Основною проблемою є вибір тривалості інтервалу для блоку. Проаналізувавши сигнал цілком, ми можемо отримати детальну спектральну картину, що несе максимально чітку інформацію про частотні складові. Проаналізувавши ж лише невеликий відрізок сигналу, ми отримуємо грубий спектр низького дозволу, що несе лише приблизну інформацію про основні частотні складові.

Іншими словами, чим менший відрізок сигналу ми розглядаємо, тим "простіше" його форма, тобто тим менше деталей початкової хвилі він несе. А чим менш "складну" форму мають звукові коливання, тим "простіше" їх спектр (тим менше в нім вищих гармонічних коливань). І навпаки, чим складніша форма коливань, тим складніший отримуваний спектр.

Ця дилема називається принципом невизначеності спектрального аналізу. Рішення шукається індивідуально, пов'язуючи його з особливостями сигналу і цілями аналізу у кожному конкретному випадку.

Існує ще одна особливість, пов'язана з проведенням блокового спектрального аналізу. Розділивши звуковий сигнал на блоки і провівши поблоковий спектральний аналіз, ми отримуємо частотний спектр, який, як правило, по своїх амплітудно-частотних характеристиках не збігається з частотним спектром цілого звукового сигналу (він "грубіший"). Проте існує ще одна важлива причина неспівпадіння згаданих частотних спектрів — це поява додаткових високочастотних складових в спектрах окремих блоків за рахунок меж інтервалів. Цей ефект (появи високочастотних складових в спектрі функції в околицях точок розриву) називається ефектом Гіббса. Щоб максимально послабити вплив цього ефекту при проведенні спектрального аналізу, досліджувану функцію (або реальний сигнал) прагнуть "згладити" в досліджуваному блоці так, щоб її значення в точках розриву на краях блоку (робочого інтервалу) відрізнялися мінімально. Сигнал множать на спеціальну віконну функцію (вагову функцію). Наприклад, функцію Хеммінга.

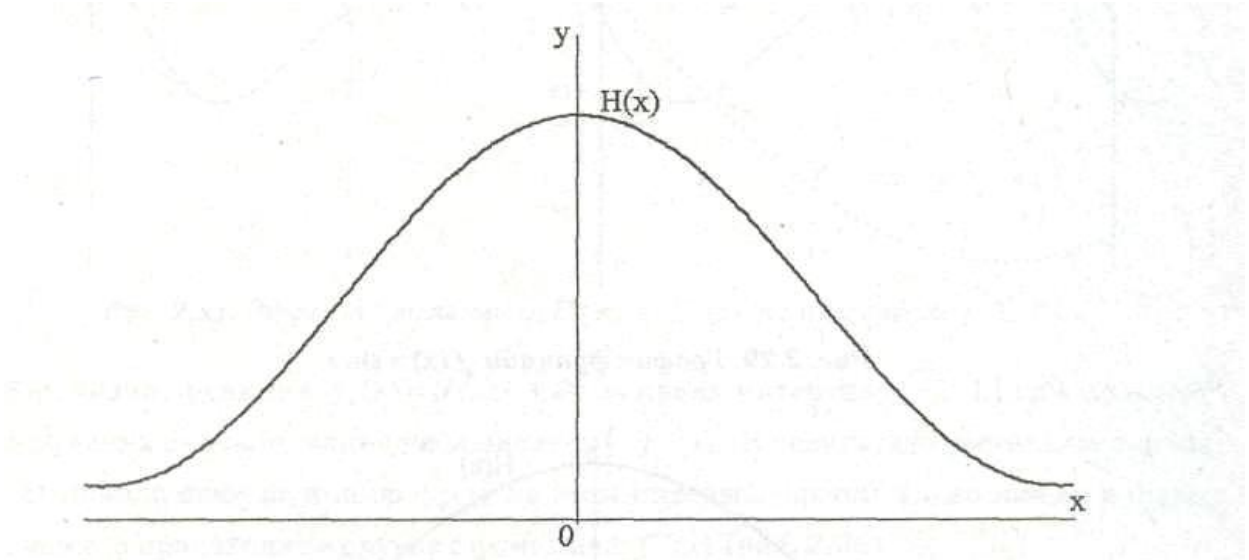


Рис. 2.18. Згладжуюча функція

Застосування віконних функцій широко поширене на практиці, проте і тут є свої недоліки. Зокрема, множення аналізованої функції на згладжуючу функцію приводить не тільки до згладжування країв, але також до деякого спотворення аналізованого фрагмента, що вносить певну погрішність до кінцевого результату. Крім того, інформація про аналізовану функцію по краях блоку після згладжування втрачається. Проте, незважаючи на ці витрати, використання згладжуючих функцій на практиці приносить більше користі, чим шкоди. Досить сказати, що при проведенні поблочного спектрального аналізу цифрових звукових сигналів із перекриттями рекомендується обов'язкове застосування згладжуючих функцій.

Всі розглянуті параметри можна знайти в будь-якій програмі отримання спектру на ПК. Як приклад розглянемо результати аналізу звукових сигналів різних джерел, що часто зустрічаються.

Перш за все, розрізняють одиночні звуки (наприклад коротке слово, вимовлене людиною, або короткий звук музичного інструменту) і тривале звучання (безперервна гра на музичному інструменті). Для більшості одиночних звуків, що існують в природі, амплітудні огинаючи виглядають типово і містить п'ять фаз звукової хвилі.

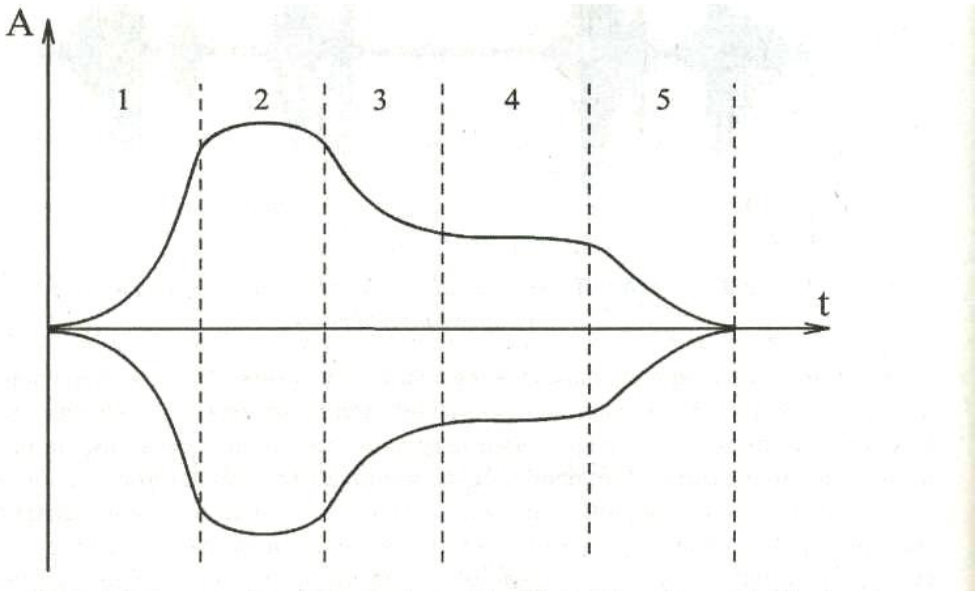


Рис. 2.19. Типова огинаюча амплітуда одиночного звукового сигналу

Ділення огинаючої амплітуди на п'ять частин, тобто п'ять фаз розвитку звукової хвилі:

- атака;
- стабілізація;
- спад;
- утримання;
- загасання.

Аналіз тривалості кожної фази може дозволити визначити джерело звуку. Наприклад музичні інструменти мають добре виражені фази, а більшість немузикальних звуків, як, наприклад, звук клацання пальцями, мають дуже нетривалу фазу стабілізації і майже нульову фазу утримання.

Джерелом людського голосу, точніше — основної частоти голосу, є голосові зв'язки. Звучання голосу представляється у вигляді складного періодичного сигналу. При формуванні звуків мови і співу, здійснюваному системою природних резонаторів мовного апарату, підкреслюються ті або інші групи довколишніх частот гармонічного спектру (спектральні максимуми). Таких спектральних максимумів в звуці може бути чотири і більше, проте розпізнавання кожного звуку пов'язане з одним або двома першими посиленими ділянками спектру, які називаються формантами.

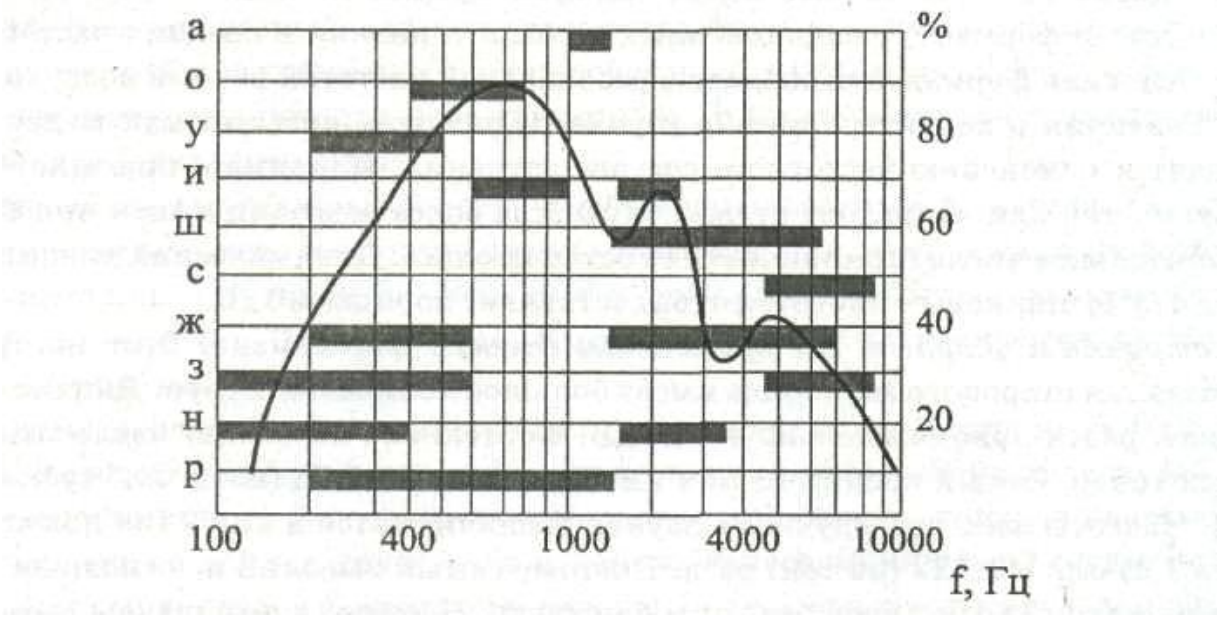


Рис. 2.20. Розміщення формантних областей

Всі ці особливості використовуються при розпізнаванні мови і обробці записів, що містять мову. Основний спектр музичних інструментів лежить в діапазоні 16 - 7000 Гц.

Одній з ключових тем, пов'язаних із звуком, є сприйняття звуку людиною. Тому слід розглянути комплекс питань, пов'язаних із сприйняттям звуку людиною, зокрема — як і в якому частотному діапазоні людина сприймає звук, які особливості і можливості слухового апарату людини, а також психофізіологічні акустичні параметри звуку. Знання цих питань потрібне при оцінці якості апаратури, розробці методів стискування звуку, створенні якісних записів і так далі.

Слуховий апарат людини здатний розрізняти частотні складові звуку приблизно в межах від 20-30 Гц до 20 кГц. Зазначена верхня межа може коливатися залежно від віку людини і інших чинників. Відмітимо, що мова йде саме про здатність слухового апарату. Частоти нижче 20-30 Гц (інфразвук) людина також здатна сприймати, але вже не вухом, а всім тілом, як вібрації. В даний час багато фахівців схильні вважати, що і вібрації на частотах, що набагато перевищують поріг 20 кГц (ультразвук) також сприймаються людиною, але вже не вухами або тілом, а безпосередньо мозком, на підсвідомому рівні.

Проте, основну інформацію про звукові коливання мозок отримує в області

частот до 4 кГц. Цей факт виявляється цілком логічним, якщо врахувати, що всі основні життєво необхідні людині звуки (голоси людей, тварин, шум води, вітру і ін.) знаходяться саме в цій спектральній смузі. Частоти вище 4 кГц є для людини допоміжними, що підтверджується багатьма дослідами. Водночас з цим чутність частот вище 4 кГц, як доповнення до основних частот, створює у людини відчуття якіснішого звучання. Тому прийнято вважати, що низькі частоти "відповідальні" за розбірливість і ясність аудіо інформації, а високі частоти — за суб'єктивну якість звуку.

Здатність аналізувати частоту звуку розподілена у людини нерівномірно. Досвідченим шляхом визначені так звані критичні смуги, частотні ділянки усередині яких здатність розрізняти частоти менше, ніж між ними. До основних психофізіологічних параметрів звуку належать: тон, висота тону і тембр звуку, гучність.

У спектрі звуку більшості музичних інструментів завжди присутня частотна складова, що найбільш виділяється по амплітуді і періоду. Її називають основною частотою або основним тоном. Тони, відповідні решті частот спектру, називаються обертонами. Якщо частоти обертонів кратні частоті основного тону, то обертони називають гармонічними складовими (гармоніками), при цьому основний тон називається першою гармонікою.

Висота звуку — це характеристика, що умовно розподіляє звуки за деякою шкалою від низьких до високих. Порівнюючи два тони, людина здатна виділити вищий і нижчий. На сприйману висоту звуку впливає, головним чином, частота основного тону, проте форма періоду звукової хвилі і її склад також можуть робити вплив на висоту звуку. Здатність слуху розрізняти два тони в нижній смузі частот набагато вище, ніж у верхній смузі. Іншими словами, частотна роздільна здатність слуху погіршується при переході від нижніх частот до верхніх. Досліди показали, що в смузі частот від 0 до 16 кГц слух людини здатний розрізняти до 620 градацій частот (залежно від інтенсивності звуку), при цьому приблизно 140 градацій знаходяться в проміжку від 0 до 500 Гц.

Висота звуку може визначатися слуховою системою і в складних сигналах,

але в тому випадку, якщо сигнал є періодичним (наприклад, звук пострілу не з'являється періодичним, і тому слух не здатний оцінити його висоту). Залежно від співвідношення амплітуд частотних складових спектру, звук може набувати різного забарвлення і сприйматися, як тон або як шум. У разі дискретного частотного спектру (тобто коли на графіці спектру присутні явно виражені списи) звук сприймається, як тон, якщо має місце один пік, або як співзвуччя, якщо мають місце декілька явно виражених піків. Якщо ж звук має суцільний спектр (тобто коли амплітуди частотних складових спектру приблизно рівні), то на слух звук сприймається, як шум.

Сприйняття висоти тону - достатньо суб'єктивне явище. Наприклад, низький чистий тон здається ще більш низьким, якщо збільшити інтенсивність його звучання. Зворотна ситуація спостерігається з високочастотним чистим тоном — збільшення інтенсивності звучання робить суб'єктивно сприйману висоту тону ще вищою. Тривалість звуку позначається на висоті тону критичним чином. Так, дуже короткочасне звучання (менше 15 мс) будь-якої частоти здається на слух просто різким клацанням — людина не зможе розрізнити висоту тону для такого сигналу. Висота тону починає сприйматися лише після 15 мс для частот в смузі 1000-2000 Гц і лише через 60 мс — для частот нижче 500 Гц. Це явище називається інерційністю слуху.

У природі ми майже не стикаємося з чистими тонами. Звучання будь-якого музичного інструменту складається з безлічі частотних складових. Навіть при дуже складних звукових коливаннях слух людини здатний розпізнати висоту звучання. Проте навіть при однаковій висоті звучання, наприклад, скрипки відрізняється на слух від звучання рояля. Це пов'язано з тим, що, окрім висоти звучання, слух здатний оцінювати також "забарвлення" звучання, тобто його тембр. Тембром звуку називається така якість звуку, яка, незалежно від частоти і амплітуди, дозволяє відрізнити одне звучання від іншого. Тембр звуку залежить від загального спектрального складу звуку, співвідношення амплітуд складових спектру і фактично не залежить від висоти основного тону. Іншими словами, тембр звуку з одним і тим же основним тоном визначається складом обертонів

(їх частотами і амплітудами), а також характером наростання амплітуд на початку звучання і їх спаду в кінці звучання. Чималий вплив на сприйманий тембр звучання надає явище інерційності слухової системи. Воно виражається, наприклад, в тому, що на розпізнавання тембру слуховій системі потрібно близько 200 мс.

Основний фізичний параметр звуку — інтенсивність (сила звуку), це кількість енергії, яка переноситься звуковою хвилею за одиницю часу через одиницю площі поверхні. Суб'єктивною оцінкою параметра інтенсивності людиною (міра сили слухового відчуття) є гучність звуку. Між інтенсивністю і гучністю звуку існує тісний, але нелінійний зв'язок. На сприйняття гучності впливають величина звукового тиску, частота і тривалість звукового сигналу.

Відчуття гучності наростає значно повільніше, ніж збільшується інтенсивність. Крім того, існує поріг чутності — найменше значення звукового тиску, що сприймається органами слуху (P0= 2*10-5 Н/м2, I0=10-12 Вт/м2). Поріг чутності також залежить від частоти звуку і може досягати свого мінімального значення в широкому діапазоні частот (700-6000 Гц).

Виходячи з необхідності роботи в такому великому діапазоні значень і з того факту, що слух має логарифмічну чутливість, в акустиці прийнято користуватися не абсолютними значеннями інтенсивності, а десятковим логарифмом відношення інтенсивності даного звуку I, до деякої стандартної інтенсивності I0, званою інтенсивністю нульового рівня, тобто

$$N = \lg\left(\frac{I}{I_0}\right)$$

Введена таким чином логарифмічна величина N називається рівнем інтенсивності звуку, одиницею її вимірювання є Бел, на честь винахідника телефону. На практиці частіше використовують десяту долю цієї величини — децибел (дБ). За інтенсивність нульового рівня прийнята величина I0=10-12 Вт/м2, відповідна звуковому тиску

$$p_0 = \sqrt{I_0 \rho_0 c_0} = 2.04 \times 10^{-5} Па$$

Ця величина лежить трохи нижче за поріг чутності на частоті 1000 Гц.

Гучність в децибелах — це L=20*log10 (P/P0), дБ.

Людина здатна відмітити приріст гучності приблизно в 1 дБ. Значення гучності різних звуків приведені в таблиці 3.

В результаті досліджень сприйняття людиною гучності чистих тонів на різних частотах, була побудована шкала одиниць вимірювання гучності звуку, так звані криві рівної гучності.

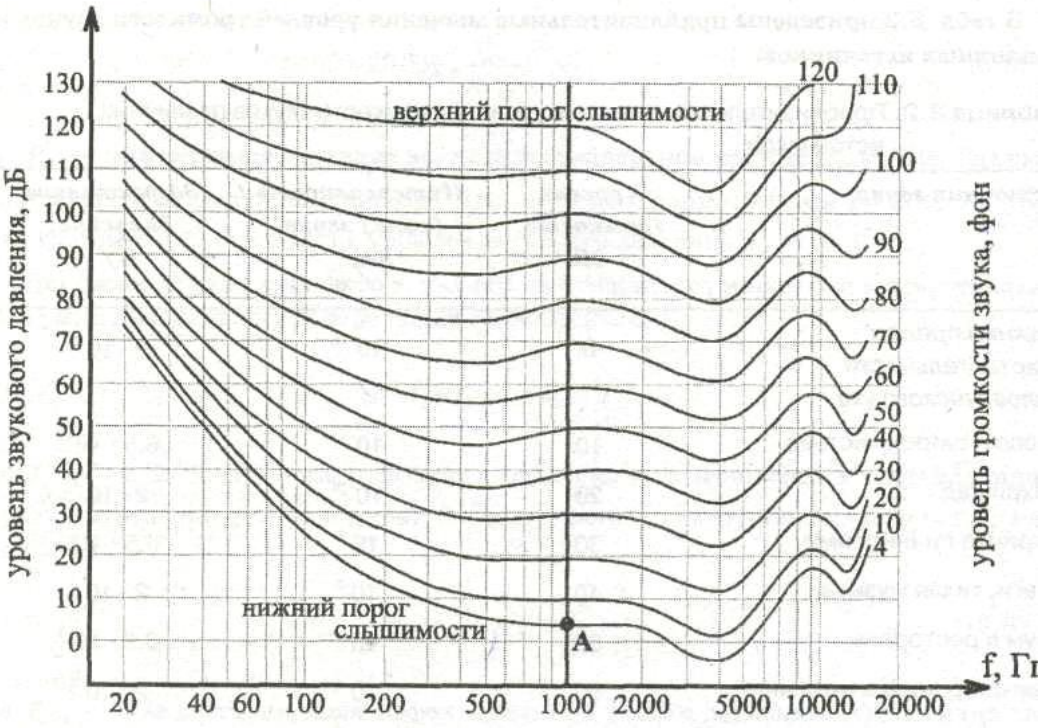


Рис. 2.21. Криві рівної гучності

Криві рівної гучності показують, що людина (середньостатистична) розпочинає чути звук з певного значення рівня звукового тиску, і на різних частотах рівень звукового тиску, починаючи з якого звук сприймається вухом, виявляється різним. Кожна крива на графіці показує рівень рівної гучності з початковою точкою відліку (точка А) на частоті 1000 Гц. Кожна лінія відповідає деякому значенню гучності. Приведені криві являються усередненими, а не еталонними. Сучасні дослідження свідчать, що вид кривих достатньою мірою залежить від умов проведення вимірювань, акустичних характеристик приміщення, а також від типу джерел звуку (гучномовці, навушники). Тому еталонного графіка кривої рівної гучності не існує.

З урахуванням розкиду значень побудована діаграма, що ілюструє усереднені характеристики сприйняття звуку різної гучності на різних частотах, т.з. діаграма слухових відчуттів (рис. 2.22).

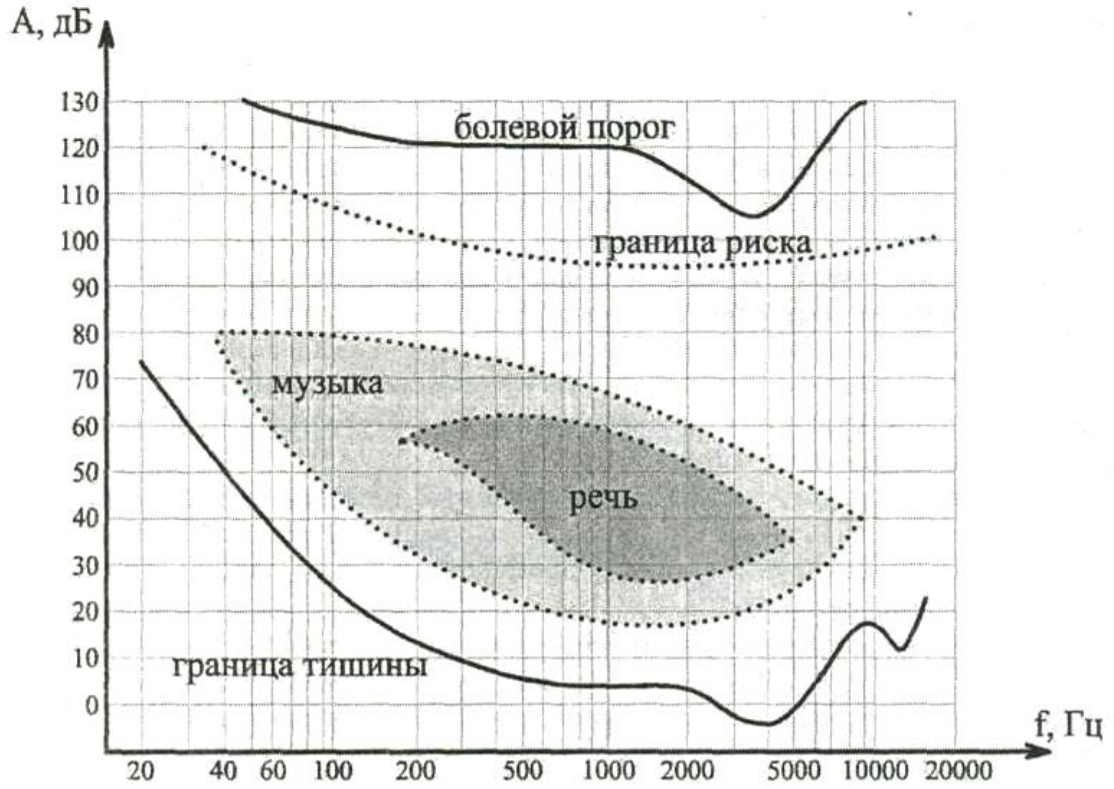


Рис. 2.22. Діаграма слухових відчуттів

Ще одним важливим явищем, пов'язаним із сприйняттям гучності звуку, є маскування.

Рівні рівної гучності і поріг чутності залежать від частоти. Крім того вони залежать від умов (наприклад, наявність фонового шуму). Цей ефект називається частотним маскуванням (рис. 2.23).

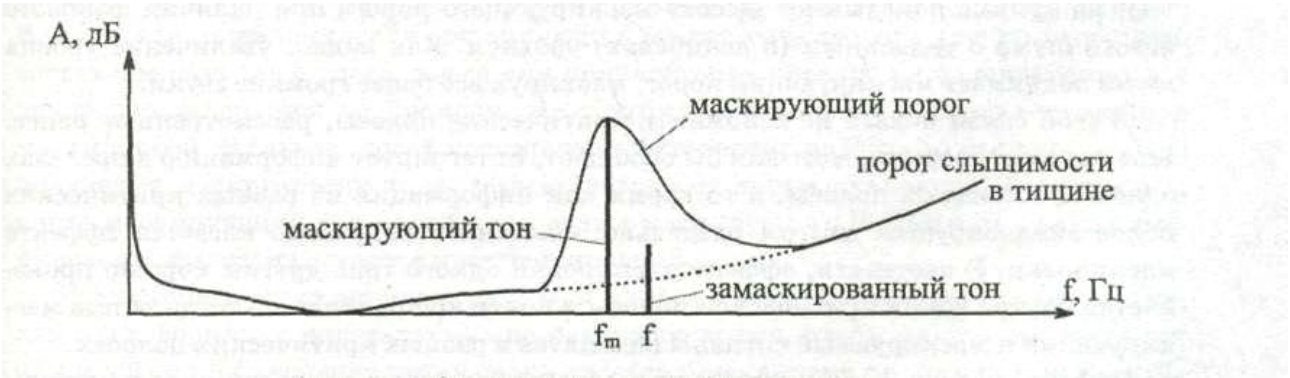


Рис. 2.23. Ефект частотного маскування

На рисунку чистий тон частоти fm створює маскуючий ефект і видозмінює криву порогу чутності в тиші. Частотна складова f має амплітуду вище за поріг чутності в тиші, а значить, була б виразно чутна в умовах тиші. Наявність же маскуючого тону fm змінює поріг чутності, і в результаті частота f на слух не відчувається (тоді як сама складова fm чутна добре).

Схожим образом тон може маскувати шум або інший сигнал в деякій частотній смузі, і навпаки — шум може маскувати тон.

Зважаючи на інерційність слуху ефект маскування може тягнутися і в часі. Зокрема, одна частотна складова може маскувати іншу частотну складову навіть тоді, коли вони з'являються в спектрі не одночасно, а з деякою затримкою в часі. Цей ефект називається частотно-часовим маскуванням.

Висновки

Основними етапами розпізнавання аудіо подій є:

1. Буферизація з перекриттям.
2. Передобробка.
3. Витяг ознак.
4. Постобробка ознак.
5. Навчання /класифікація.

Після вилучення необхідних ознак сигналу для їх подальшого використання проводиться нормалізація ознак, а потім відбувається вибір класифікатора (навчальної моделі).

До основних психофізіологічних параметрів звуку відносяться: тон, висота тону і тембр звуку, гучність. Висота звуку — це характеристика, що умовно розподіляє звуки за деякою шкалою від низьких до високих. Порівнюючи два тони, людина здатна виділити вищий і нижчий. Основний фізичний параметр звуку — інтенсивність (сила звуку), це кількість енергії, яка переноситься звуковою хвилею за одиницю часу через одиницю площі поверхні. Суб'єктивною оцінкою параметра інтенсивності людиною (міра сили слухового відчуття) є гучність звуку.

РОЗДІЛ 3
ПРОГРАМНИЙ МЕТОД РОЗПІЗНАВАННЯ АУДІОПОДІЙ

3.1. Планування розробки програмного продукту

Перед етапом розробки програми та тестуванням базової системи розпізнавання звуку, необхідно скласти загальну стратегію. Система має розпізнати звук із сукупністю певних характеристик та локалізувати джерело звуку. Отже робимо висновок, що складовими компонентами системи будуть:

- мікрофон для запису звуку;
- комп'ютер для обробки аудіо та надсилання команд;
- програмний код алгоритму розпізнавання;
- інтерфейс для роботи з алгоритмом.

Щоб розрізнити аудіо події (постріл, биття скла, крик тощо) від інших звуків, важливо зрозуміти, чим саме характеризуються такі звуки. Для людського вуха такими характеристиками є гучність та короткочасність звуку. За словами Майкла і Люсьєн Хааг, при вимірюванні енергетичної величини на відстані 1 метр, енергія звуку пострілу часто більша у децибелах за звуки ланцюгової пилки та відбійного молотка. Крім того, перехід від початку події до її піку відбувається майже миттєво. Одне з досліджень, зокрема, виявило, що звук пострілу часто триває менше 3 мілісекунд. Це означає відносну інтенсивність «форми хвилі». За допомогою візуального представлення звукового сигналу можливо відокремити звук пострілу від інших звуків.

3.2. Вимоги до проектної системи

Перш за все необхідно було визначити ключові функціональні та нефункціональні вимоги проекту, щоб зрозуміти, яку архітектуру підібрати для створення продукту.

3.2.1. Функціональні та нефункціональні вимоги

Як результат процесу розробки програмного забезпечення, фінальний продукт має виконувати наступні функції:

- Мати здатність отримувати аудіо потік зі сторонніх носіїв таких як звуко записуючі пристрої та передавачі аудіо потоку.
- Працювати з різноманітними форматами аудіо.
- Вміти оброблювати аудіо доріжку по заданим параметрам та видавати результат аналізу.
- Мати можливість запускатися на різних пристроях незалежно від встановленої операційної системи.
- Надсилати результати обробки на виділені пристрої в режимі реального часу.
- Певний час зберігати аудіо матеріал та результати його обробки на виділених носіях.
- Надавати доступ до адміністрування параметрі та роботи з аудіо матеріалами.
- Користувач може обрати пристрій, аудіо доріжки з якого хоче послухати.
- Користувач може бачити на мапі місце, де встановлено звукозаписуючий пристрій.
- Користувач може бачити на мапі ділянку із підвищеною частотою (> N).
- Розроблювана система повинна бути кросс-браузерним веб-застосунком.
- Система повинна підтримувати до 100 користувачів одночасно.
- Система повинна відповідати на запит протягом однієї секунди.

Таблиця 3.

Визначення функціональних вимог системи

Функціональні вимоги	
FR1	Вхідні дані
FR1.1	Вхідними даними є аудіо потік, що отримується зі звукозаписуючого пристрою
FR1.2	Форматами вхідних даних мають бути: MPEG, WAV, AAC
FR2	Обробка аудіо потоку
FR2.1	Програма повинна розпізнавати звук частотою вищою сталої N
FR2.2	Аудіо має сегментуватись, а у пам’яті мають зберігатись лише розпізнані ділянки звуку
FR2.3	Програма має зберігати час, у який відбувся запис аудіо сегменту
FR3	Інтерфейс користувача
FR3.1	Користувач може обрати пристрій, аудіо доріжки з якого хоче послухати
FR3.2	Користувач може бачити на мапі місце, де встановлено звукозаписуючий пристрій
FR3.3	Користувач може бачити на мапі ділянку із підвищеною частотою (> N)

Таблиця 4.

Визначення нефункціональних вимог системи

Нефункціональні вимоги	
NFR 1	Розроблювана система повинна бути кросс-браузерним веб-застосунком
NFR 2	Система повинна підтримувати до 100 користувачів одночасно
NFR 3	Система повинна відповідати на запит протягом однієї секунди
NFR 4	Система розроблятиметься застосовуючи методологію SCRUM та Kanban
NFR 5	Середня тривалість часу між двома послідовними проявами помилок у системі 8 годин

3.2.2. Компоненти і бізнес логіка системи

Архітектуру програмної системи було зображено у формі діаграми компонентів (рис. 3.1.).

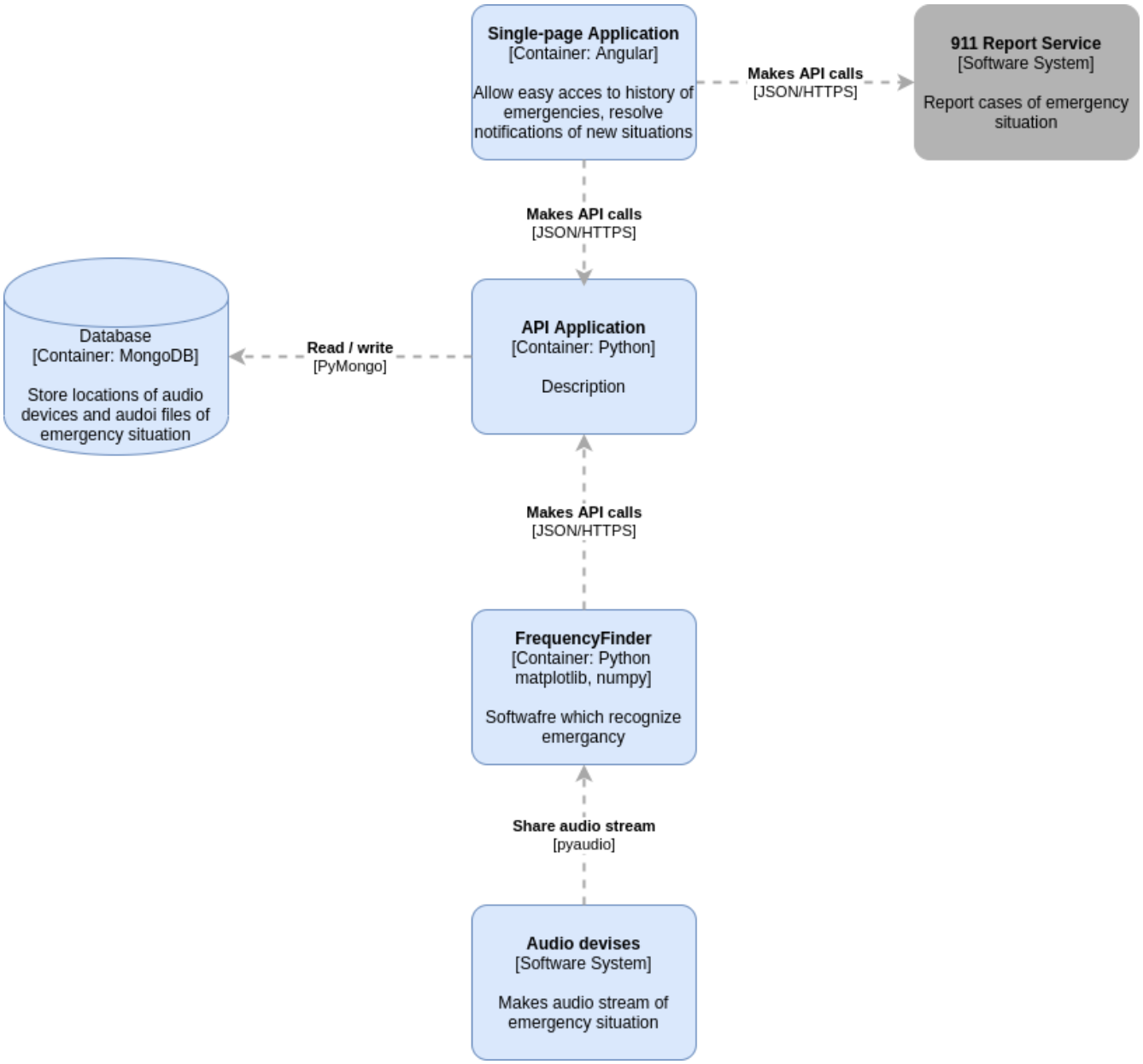


Рис. 3.1. Діаграма компонентів

Загалом у програмі можна виділити наступні компоненти та їх функції:

- Інтерфейс користувача
- Дивитись логи в процесі роботи програмного забезпечення.
- Отримувати записані та опрацьовані матеріали.
- Налаштовувати параметри обробки аудіо матеріалів.

Бізнес логіка

- Забезпечувати неперервну роботу протягом довгого часу.
- Забезпечувати коректну обробку аудіо матеріалів відповідно до прописаних алгоритмів та заданих користувачем параметрів.
- Забезпечувати зберігання інформації та її резервне копіювання.
- Забезпечувати роботи з периферійними приладами.

Компоненти клієнтської частини архітектури:

1. Відображення місце розташування мікрофона, звук з якого вище сталої “N” (map service component).

2. Інформація про виявлений звук вище сталої “N”

- Координати;
- Гучність;
- Номер.

3. Робота з файлом аудіо запису (Visualizer, Player)

- Прослухати;
- Зупинити;
- Збільшити гучність;
- Додати коментар;
- Надіслати сповіщення.

У ході розробки користувацької частини системи було застосовано наступні бібліотеки:

- jQuery
- wave surf

Компоненти серверної частини архітектури:

1. Аналізатор частоти потоку.

Після перевірки наявності підключених джерел звуку програма підтримує постійний зв'язок з ними. Отримує звуковий потік та конвертує його в об’єкт, яким він може оперувати. Раз у фіксований відрізок часу система записує та аналізує стан потоку в цей момент. Результати аналізу записуються в чергу і передаються для перевірки наступним компонентом.

2. Аналізатор розташування джерела звуку.

Приймає потік з модифікованими даними і слідкує за наявністю аномалій. В разі присутності аномалії в даних починає їх перевірку, і в разі виявлення співпадінь в базі шаблонів даних отримує координати пристрою з якого були зняті дані та відправляє повідомлення з координатами, даними та назвою шаблону з яким було виявлено співпадіння.

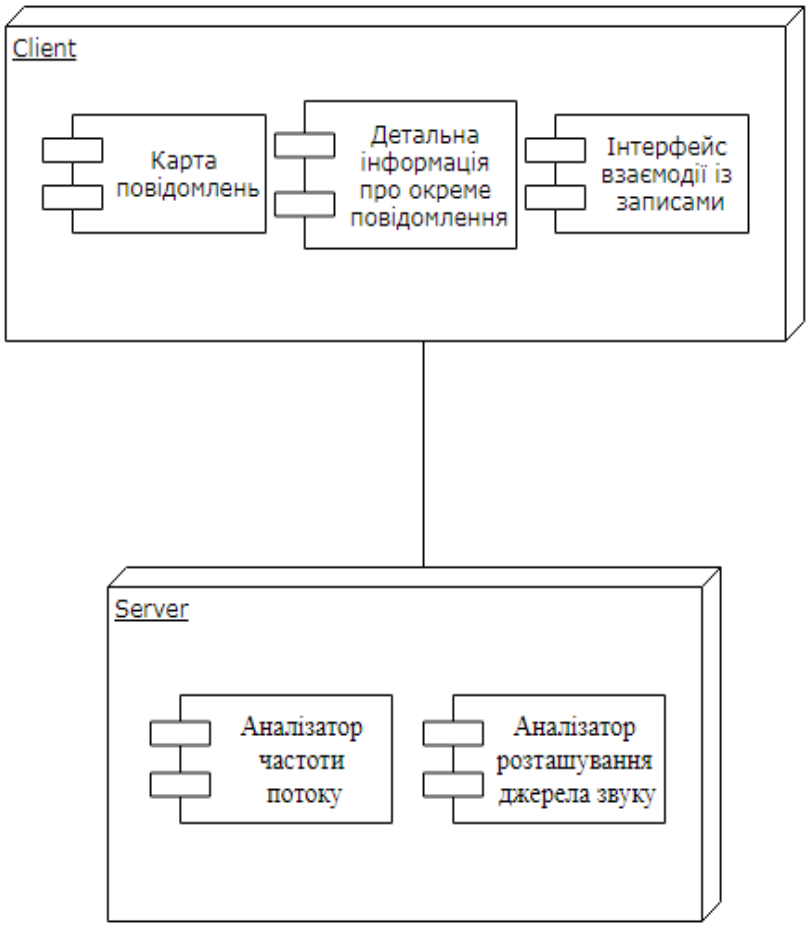


Рис. 3.2. Діаграма архітектури (клієнт-сервер)

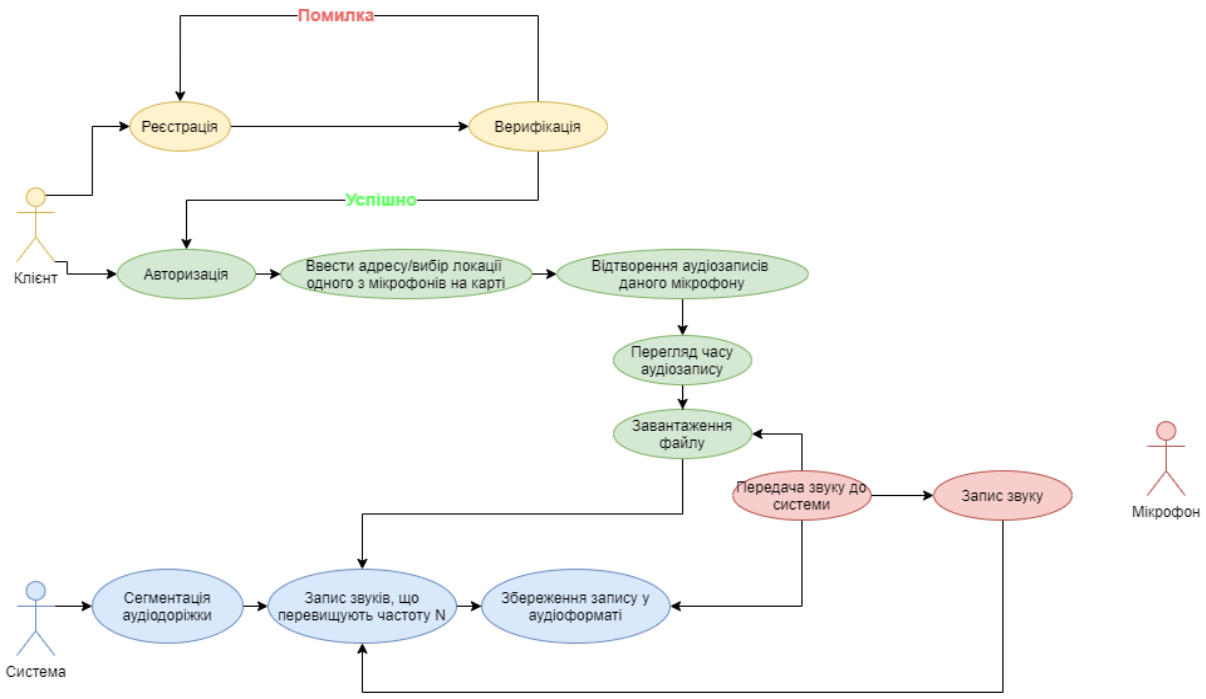


Рис. 3.3. Use Case діаграма

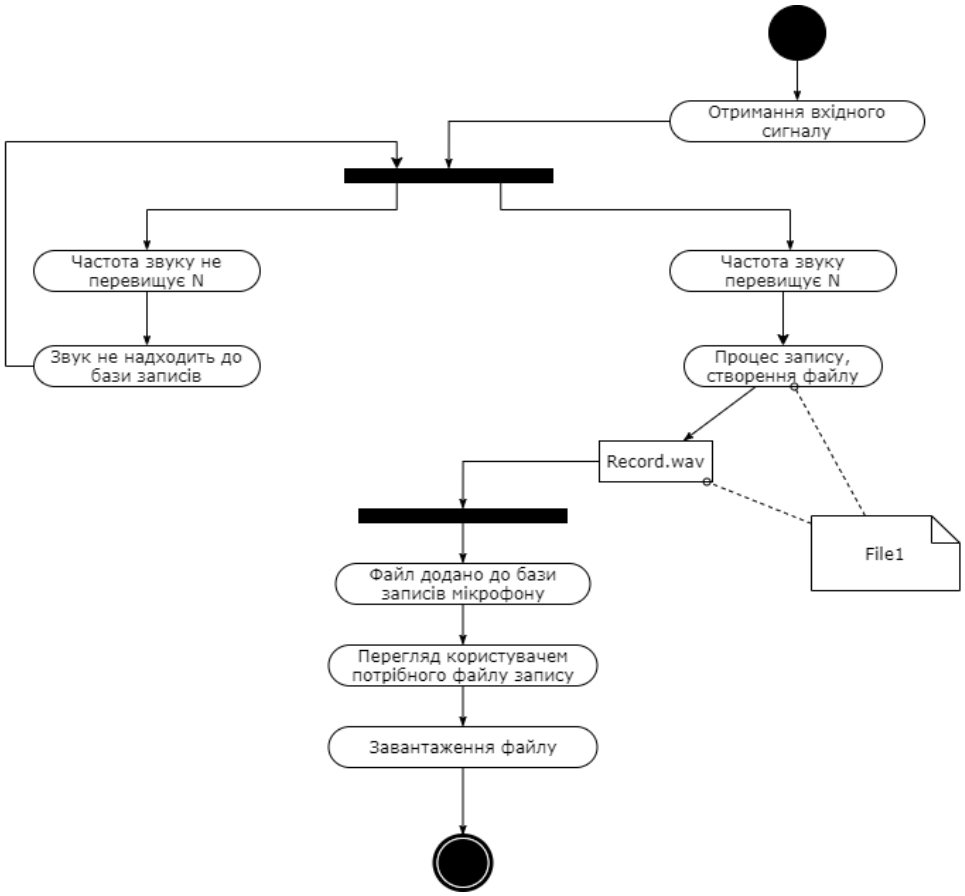


Рис. 3.4. Блок-схема алгоритмів системи

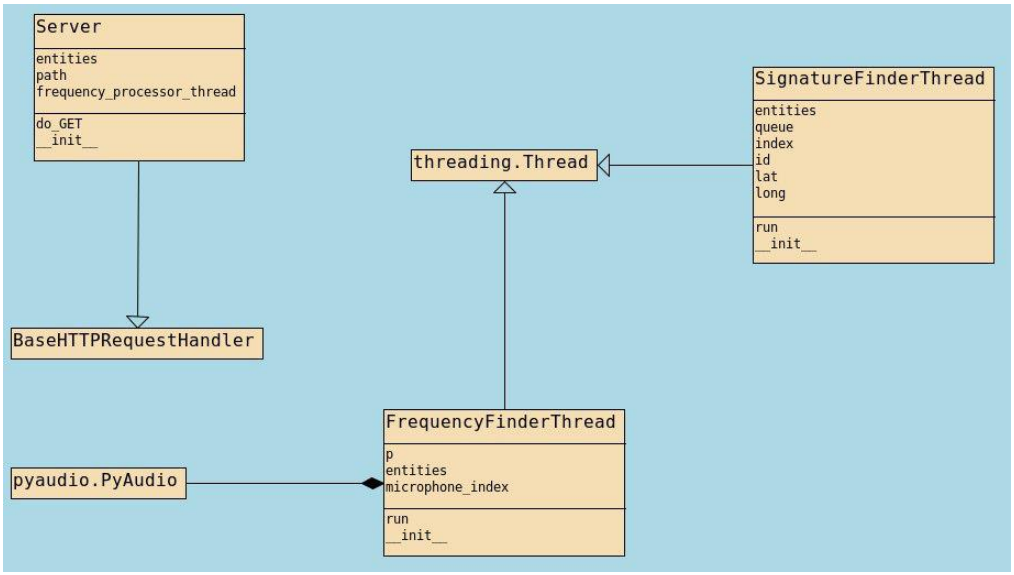


Рис. 3.5. Бізнес-логіка роботи програмної системи в вигляді діаграми класів (серверна частина)

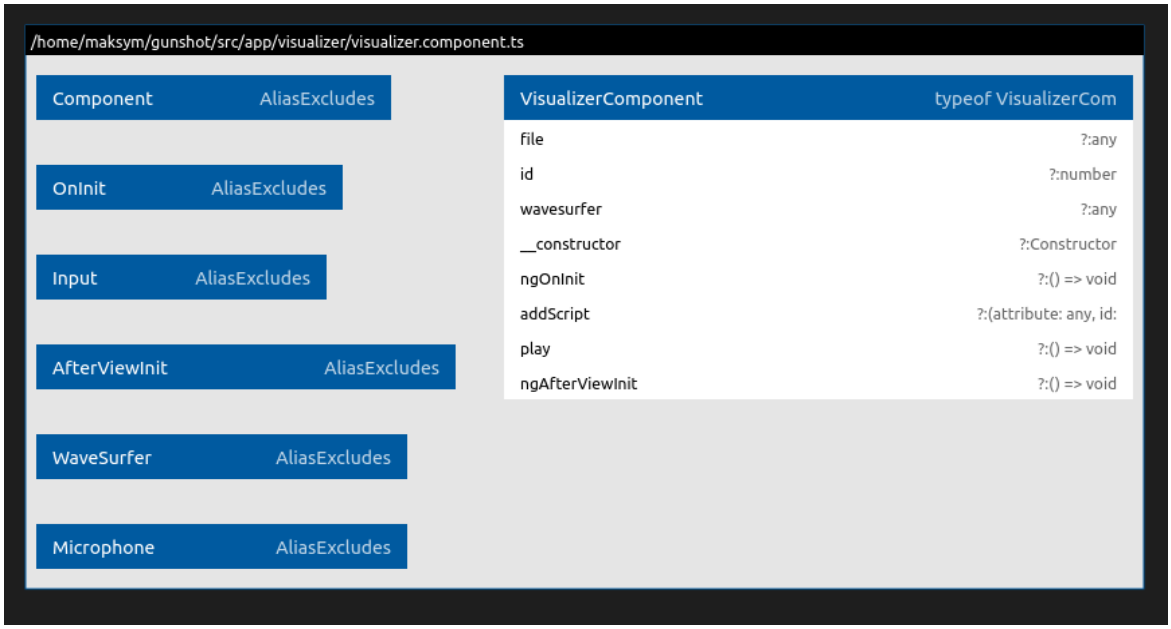


Рис. 3.6. Бізнес-логіка роботи програмної системи в вигляді діаграми класів (веб частина)

3.2.3. Дослідження ринку

Основними напрямками досліджень ринку були:

- Дослідження задля розуміння ступені конкуренції;
- Дослідження підходу до реалізації інтерфейсу користувачів у наявних конкурентів;
- Дослідження наукової літератури для отримання теоретичних міркувань щодо ергономічного використання існуючих алгоритмів і методів розробки.

Основним джерелом інформації стали звіти по аналізу ринку та наявних і використовуваних конкурентами ресурсів. Слід зазначити, що більшість продуктів, існуючих на ринку, виконані професіонально, оскільки предметна область пов’язана з безпекою. До розробки таких алгоритмів були залучені науковці, військові та служби внутрішньої безпеки. Однак, можна виділити кілька моментів, які могли би покращити як якість послуг, так і розуміння продукту в цілому.

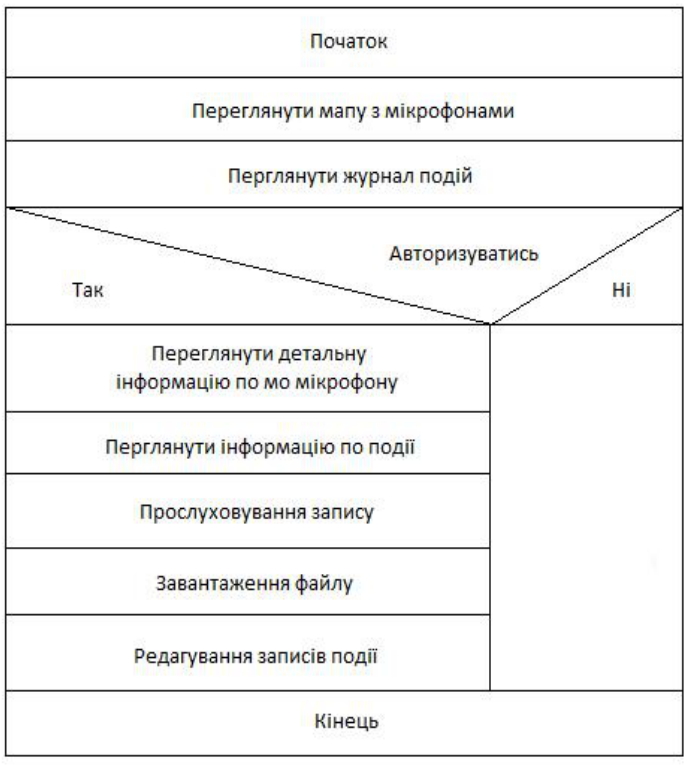


Рис. 3.7. Діаграма Насси — Шнейдермана

Аналіз продуктів конкурентів дає чіткий висновок, що більшість досліджених програмних продуктів виконані не ергономічно, мають відчутно складний інтерфейс користувача (не дивлячись на те, що багато з них підтримуються лише українську та російську мови, що також негативно відобразиться на показнику гнучкості продукту), з великим трудом зрозумілий навіть для користувача, що має теоретичні поняття предметної області. Також вартий уваги момент, що ряд досліджених продуктів не надають ніякої інформації про зафіксовані частоти, виконуючи таким чином чисто ілюстративну функцію. Інші, в цей же час, завантажують користувача великою кількістю неважливої інформації, яку останній може навіть не зрозуміти.

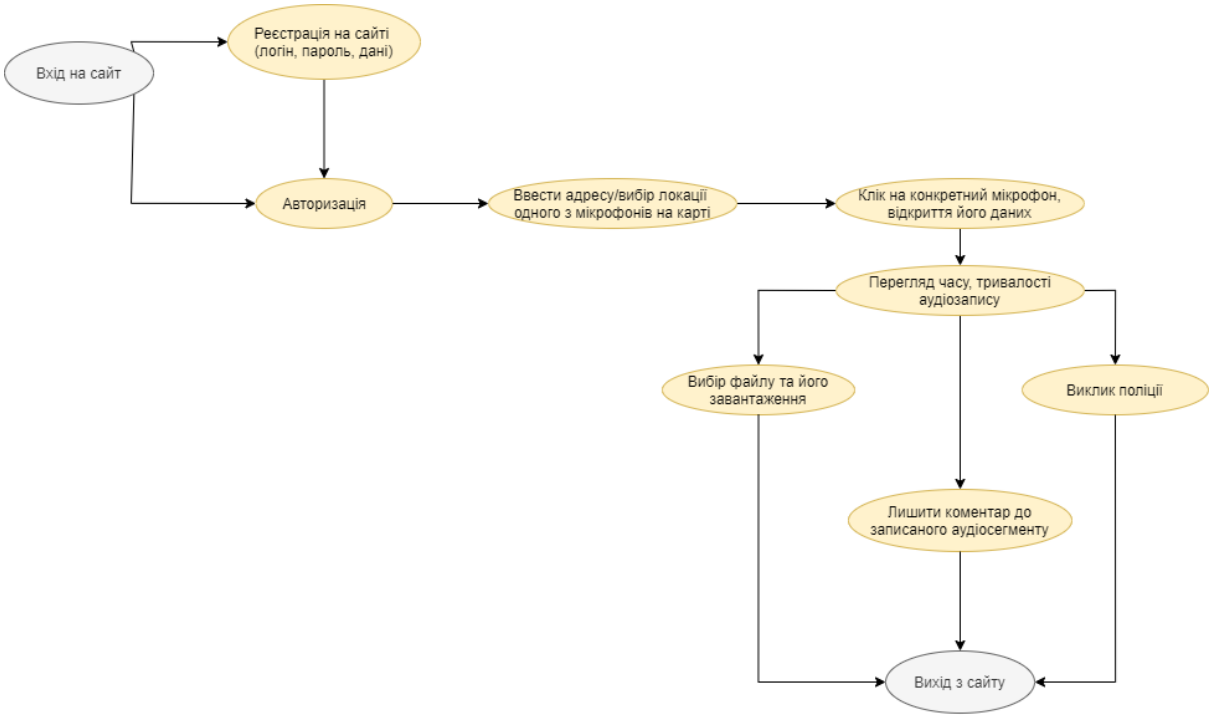


Рис. 3.8. Діаграма призначених для користувача сценаріїв

3.3. Аналіз інтерфейсу та стилістики продукту

Спираючись на дослідження та тренди останніх років головуючим стилем було обрано мінімалізм. Він дуже поширений у веб-просторі. Основний акцент — на простому і непомітному оформленні. Не повинно бути нічого зайвого. Інформативність проявляється не в кількості контенту, а навпаки. Іноді весь

текст на мінімалістичних сайтах замінений на відео або стиснутий до пари фраз або заклику до дії.

Лаконічність проголошується головним принципом, а кількість візуальних елементів зводиться до мінімуму. Цим правилам підкоряється все, від структури інтернет-ресурсу до вибору шрифтів. В основі мінімалізму лежить пряма логіка і повна відмова від надмірностей.

Структура інтерфейсу

Користувацький інтерфейс продукту складається з:

1. Головне меню (гамбургер-меню) для навігації сайтом.
2. Мапа (компонент мапи + вказані на ній мікрофони та їх координати)
3. Блок аудіо записів з мікрофону (список записів)
4. Блок детальної інформації про запис (аудіо програвач, інфо, коментарі)
5. Сторінка реєстрації/авторизації
6. Хідер (включає в себе меню + строка статусу)

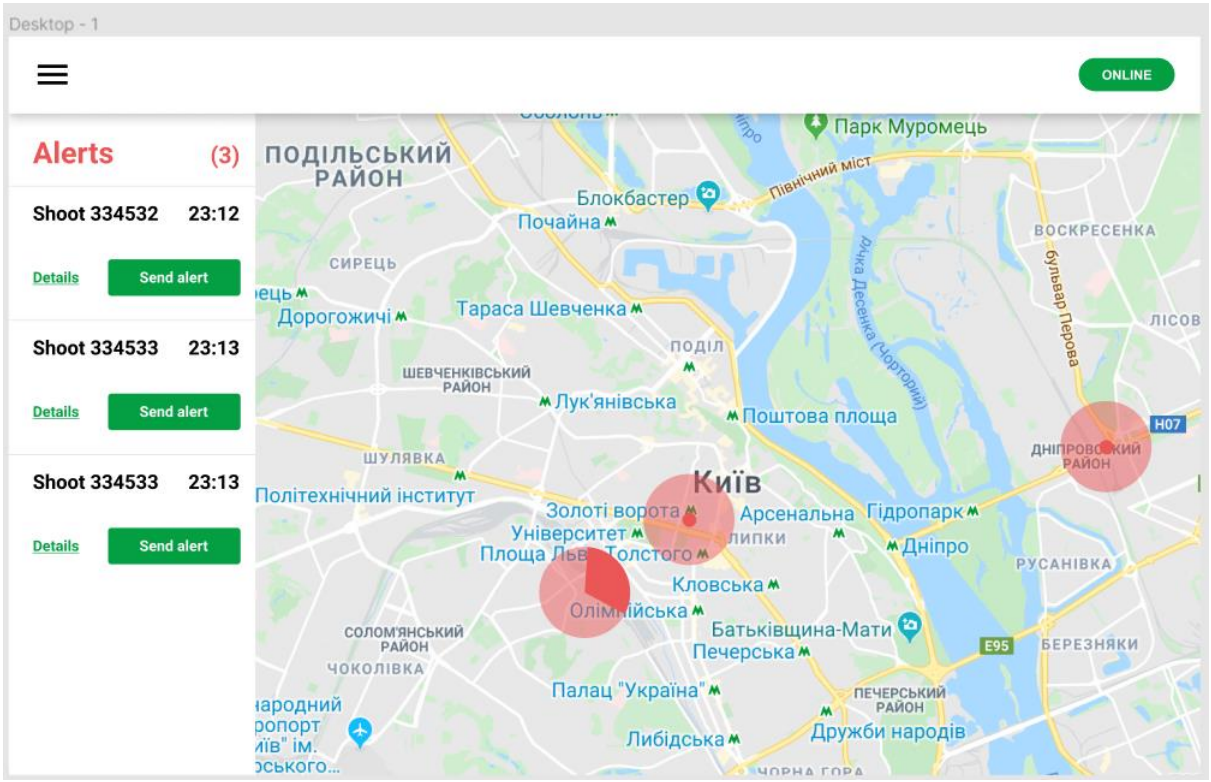


Рис. 3.9. Інтерфейс головного екрану програми

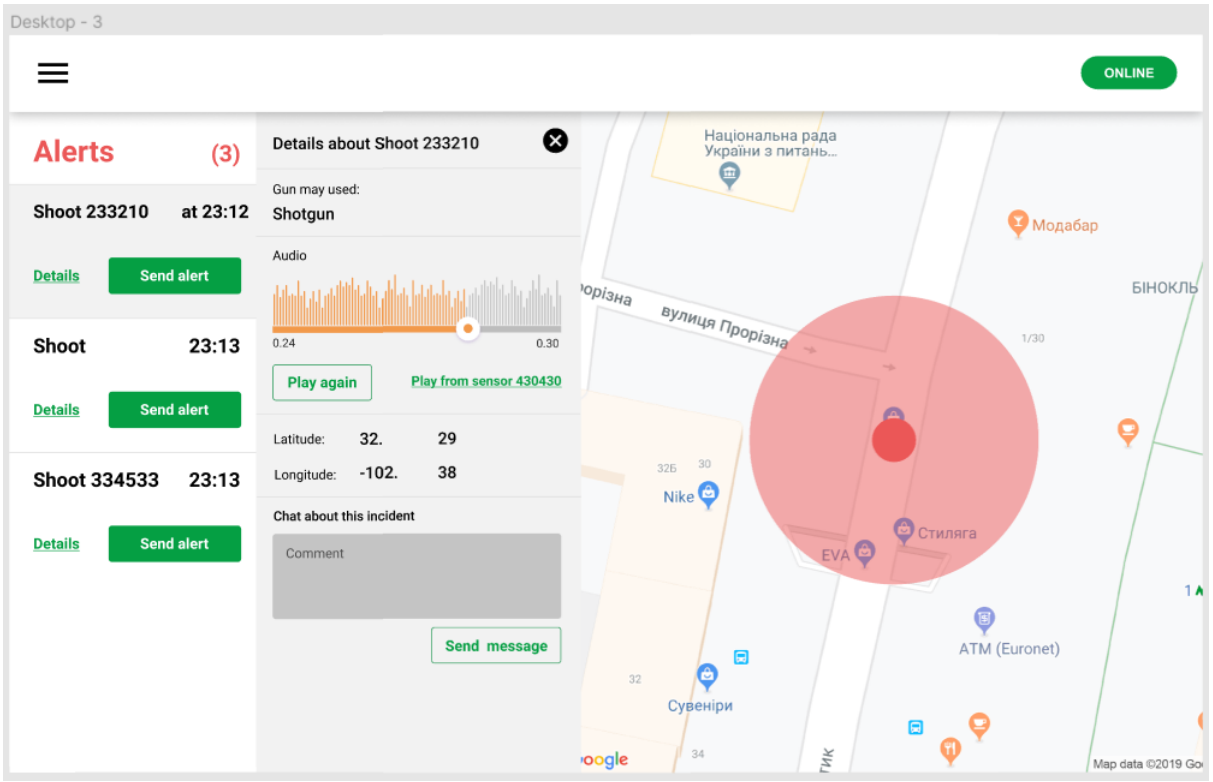


Рис. 3.10. Інтерфейс меню роботи з аудіо доріжкою

Висновки

Основними складовими системи розпізнавання аудіо подій є:

- мікрофон для запису звуку;
- комп’ютер для обробки аудіо та надсилання команд;
- програмний код алгоритму розпізнавання;
- інтерфейс для роботи з алгоритмом.

Вхідними даними програми є аудіо потік, що отримується зі звуко-записуючого пристрою у форматі MPEG, WAV, AAC. Програма може сегментувати аудіо файл, а в пам’яті зберігати лише розпізнані ділянки звуку. Через зручність користування програма буде кросс-браузерним веб-додатком. Більшість продуктів, існуючих на ринку, виконані професіонально, оскільки предметна область пов’язана з безпекою. Однак, можна виділити кілька моментів, які могли би покращити як якість послуг, так і розуміння продукту в цілому.

РОЗДІЛ 4
ІМПЛЕМЕНТАЦІЯ ПРОГРАМНОГО МЕТОДУ У ВИГЛЯДІ ПРОГРАМНОГО ДОДАТКУ

4.1. Обґрунтування і вибір мови програмування

Бібліотека PyAudio – кросплатформенна бібліотека для роботи з вхідними та вихідними аудіопотоками. За допомогою цієї бібліотеки розробник має змогу використовувати Python для відтворення та запису звуку на різних платформах, аналізувати довжину аудіохвилі. Перспектива використання цієї бібліотеки у алгоритмах розпізнавання частот звуку полягає у її гнучкості, легкому налаштуванні та можливістю роботи з широким спектром аудіоформатів, зокрема, із форматом wave.

Для запису чи відтворення звуку необхідно відкрити потік на потрібному пристрої із необхідними параметрами звуку за допомогою функції pyaudio.PyAudio.open(). Записати аудіодані в потік можна, використовуючи, функцію pyaudio.Stream.write(). Щоб прочитати аудіодані з потоку, необхідно викликати pyaudio.Stream.read(). В режимі зворотнього виклику PyAudio викличе вказану функцію, коли будуть потрібні нові аудіодані (для відтворення), або якщо з’являються нові (записані) аудіодані. PyAudio є незмінним помічником у алгоритмах аналізу звуку різної складності (рис. 4.1).

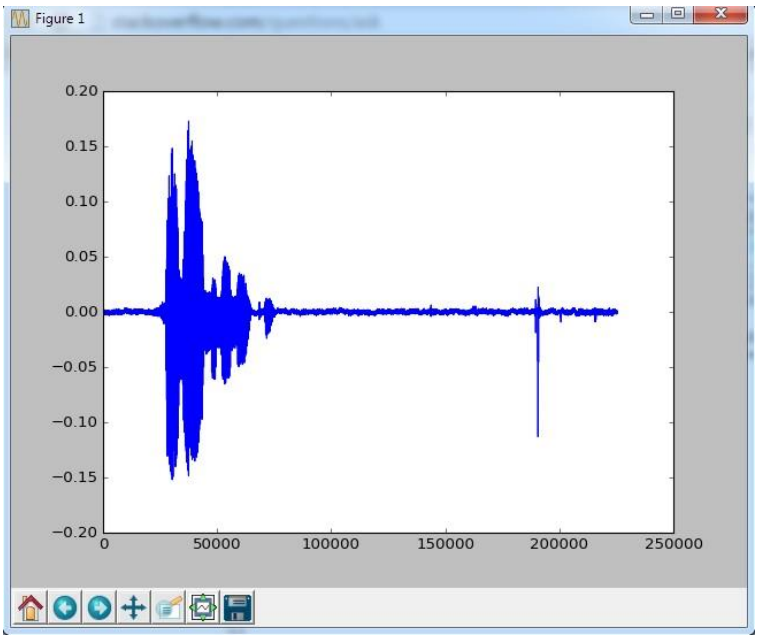


Рис. 4.1. Функція побудови графіку звукової хвилі на мові Python

Майбутній додаток матиме в своїй основі клієнт-серверну архітектуру, відповідно до цього факту доцільним буде описати вибір та обґрунтування мови програмування як для front-end частина, так і для back-end.

Отож front-end частина написана за допомогою TypeScript. Його використання має свої переваги:

- Із-за статичної типізації код на TypeScript більш передбачуваний і тому його легко редагувати.
- Завдяки гарній підтримці ООП, модулів та просторів імен, проект, навіть при великому об'ємі, буде впорядкованим.
- Компілятор на стадії збирання знаходить багато помилок, ще до того як вони потрапляють в рантайм і стануть причиною збоїв.
- Angular 2 написана на TypeScript і рекомендує його як мову розробки.

Angular 2 буде використовуватись в якості фреймворку. І не дивлячись на те, що він також підтримує JavaScript, TypeScript підходить для проекту набагато краще, код буде чистішим та більш зрозумілим для інших розробників.

Використання технологій:

- Bootstrap
- JQuery

Bootstrap - це відкритий і безкоштовний HTML, CSS і JS фреймворк, який використовується веб-розробниками для швидкої верстки адаптивних дизайнів сайтів та веб-додатків. Фреймворк Bootstrap використовується по всьому світу не тільки незалежними розробниками, але іноді й цілими компаніями. На Bootstrap створено дуже багато різних сайтів, подивитися їх можна на сторінці Bootstrap Expo. Основна область його застосування - це фронтенд розробка сайтів та інтерфейсів адмінок. Bootstrap дозволяє верстати сайти в кілька разів швидше, ніж це можна виконати на «чистому» CSS і JavaScript. Крім цього, його популярність ще обумовлена доступністю. Вона полягає в тому, що на ньому навіть початківець розробник може верстати досить якісні макети, які важко було б виконати без глибоких знань веб-технологій і достатньої практики.

Фреймворк Bootstrap являє собою набір CSS і JavaScript файлів. Щоб його використовувати ці файли необхідно просто підключити до сторінки. Але, Bootstrap - це не просто набір готових інструментів (HTML фрагментів, класів, компонентів і плагінів), а добре спроектований фронтенд фреймворк, який досить просто можна налаштувати під себе.

Bootstrap включає в себе:

- набір інструментів для створення макета (обгортковий контейнерів, потужної системи сіток, гнучких медіа-об'єктів, адаптивних класів);
- класів для стилізації базового контенту: тексту, зображень, коду, таблиць;
- готових компонентів: кнопок, форм, горизонтальних і вертикальних навігаційних панелей, слайдерів, випадаючих списків, модальних вікон, спливаючих підказок тощо;
- класів для вирішення традиційних завдань найбільш часто виникають перед веб-розробниками: вирівнювання тексту, відображення та приховування елементів, завдання кольору, фону відступів тощо.

jQuery - це бібліотека, написана на мові JavaScript, яка призначена для кроссплатформенного маніпулювання HTML-сторінкою після того як вона відобразиться в браузері. Крім цього, jQuery містить інструменти, що допомагають перехоплювати і обробляти події, створювати анімацію на веб-сторінці і методи, які дозволяють взаємодіяти з сервером без перезавантаження сторінки (AJAX).

Основна причина популярності бібліотеки jQuery полягає в тому, що при реалізації типового функціоналу не потрібно нагромаджувати велику конструкцію на мові JavaScript, яка до того ж може не працювати у всіх браузерах. Це відбувається, тому що деякий об'єкт, властивість або метод JavaScript може підтримуватися в одному браузері, а в іншому немає. В цьому випадку для цього браузера доводиться писати деякий додатковий (обхідний) код, щоб можливість, яку було закладено в першій версії коду, підтримувалася в цьому браузері. Щоб цього уникнути, можна скористатися бібліотекою jQuery,

яка містить велику кількість функцій, призначених для вирішення типових завдань, що стоять перед веб-розробниками і підтримуються усіма браузерами.

Back-end частина написана на мові програмування Python. Серед основних її переваг можна назвати такі:

- чистий синтаксис (для виділення блоків слід використовувати відступи);
- переносимість програм (що властиве більшості інтерпретованих мов);
- стандартний дистрибутив має велику кількість корисних модулів (включно з модулем для розробки графічного інтерфейсу);
- можливість використання Python в діалоговому режимі (дуже корисне для експериментування та розв'язання простих задач);
- стандартний дистрибутив має просте, але разом із тим досить потужне середовище розробки, яке зветься IDLE і яке написане мовою Python;
- зручний для розв'язання математичних проблем (має засоби роботи з комплексними числами, може оперувати з цілими числами довільної величини, у діалоговому режимі може використовуватися як потужний калькулятор);
- відкритий код (можливість редагувати його іншими користувачами).

Слід також виділити, що серверна частина буде написана засобами NodeJS. До основних переваг цієї платформи можна віднести модель вводу-виводу, швидкість роботи, синтаксис знайомого нам JavaScript, а також постійне оновлення та підтримка з боку розробників.

Як асинхронне подієве JavaScript–оточення, Node.js спроектований для побудови масштабованих мережевих додатків. Для кожного з’єднання викликається функція зворотнього виклику, проте коли з’єднань немає, Node.js засинає.

Це контрастує з більш загальною моделлю в якій використовуються паралельні OS потоки. Такий підхід є відносно неефективним та дуже важким у використанні. Більше того, користувачі Node.js можуть не турбуватись про

блокування процесів, оскільки немає жодних блокувань. Майже жодна з функцій у Node.js не працює напряму з I/O, тому процес не блокується ніколи. Оскільки нічого не блокується на Node.js легко розробляти масштабовані системи.

HTTP є об'єктом першого роду в Node.js, розробленим з потоковістю та малою затримкою. Це робить Node.js хорошою основою для веб–бібліотеки або фреймворку.

Те що Node.js спроектований без багатопоточності, не означає, що ви не можете використовувати можливості кількох ядер у вашому середовищі. Ви можете створювати дочірні процеси, якими легко керувати з допомогою API child_process.fork(). Модуль cluster побудований на цьому інтерфейсі і дозволяє вам ділитись сокетами між процесами та розподіляти навантаження між ядрами.

4.2. Тестування та підтримка програмного продукту

Результати первинного тестування модулів

В першу чергу було проведене Smoke-тестування, яке показало працездатність модулів програмного забезпечення.

Наступним етапом було проведено тестування “Чорної скриньки”. Інтерфейс дає змогу виконувати наступні функції:

- Можна обрати пристрій і подивитися його логи
- Можна обрати пристрій, аудіо доріжки
- Мапа показує місця, де встановлено звукозаписуючі пристрої
- На мапі відображаються ділянки із зафіксованими перевищеннями частоти

Після цього було проведено тестування “Білої скриньки”. Програма змогла виконати наступні функції:

- Модуль аналізу звукової доріжки справляється з обробкою аудіопотоку
- Модуль аналізу звукової доріжки отримує звукову доріжку з під’єднаних звуко записуючих пристроїв
- Модуль аналізу звукової доріжки працює з різноманітними форматами даних.

- Модуль аналізу частоти звукової доріжки розпізнає відхилення від контрольних значень.
- Модуль аналізу частоти звукової доріжки зберігає час та координати значних відхилень від контрольних значень.

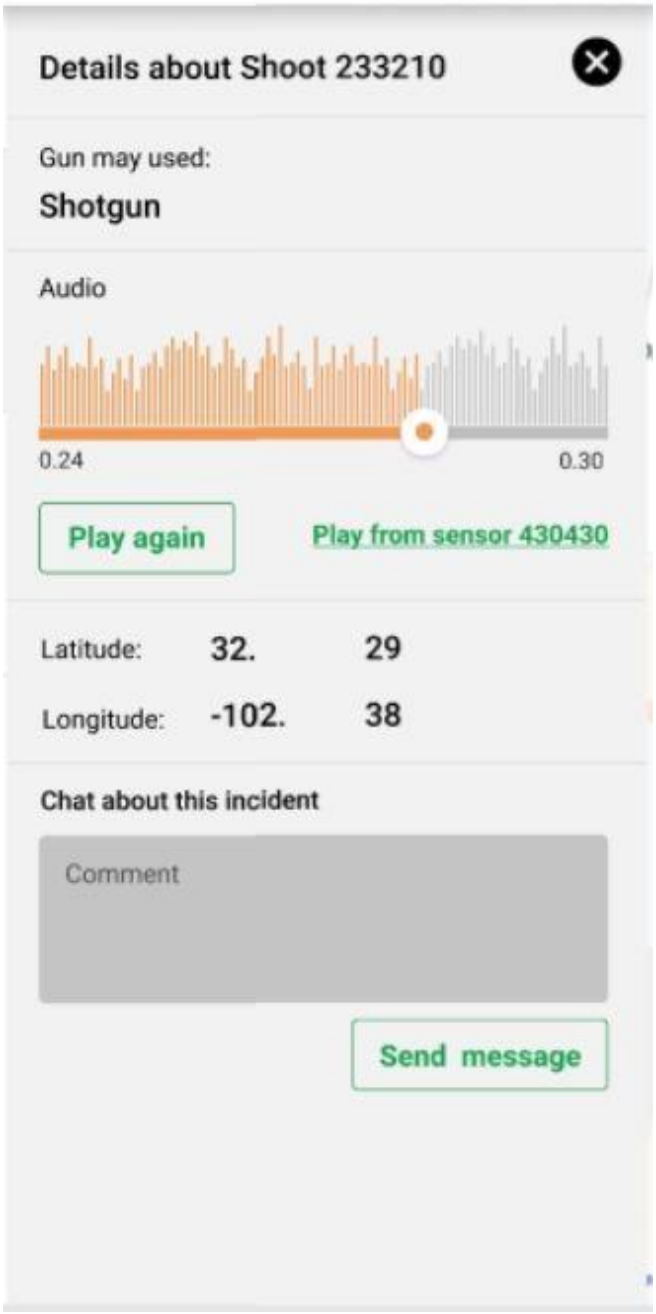


Рис. 4.2. Відображення інформації про аудіо подію

4.3. Перший запуск програми

Запуск серверної частини програми:

1. Завантажити python 3:

sudo apt-get install python3

2. Завантажити pip(package manager):

sudo apt-get install python3-pip

3. Виконати імпорт потрібних модулів:

queue
matplotlib
numpy
pyaudio
datetime
http.server
pip3 install -r requirements.txt pip install some-package-name

4. Запустити server.py:

sudo python3 server.py

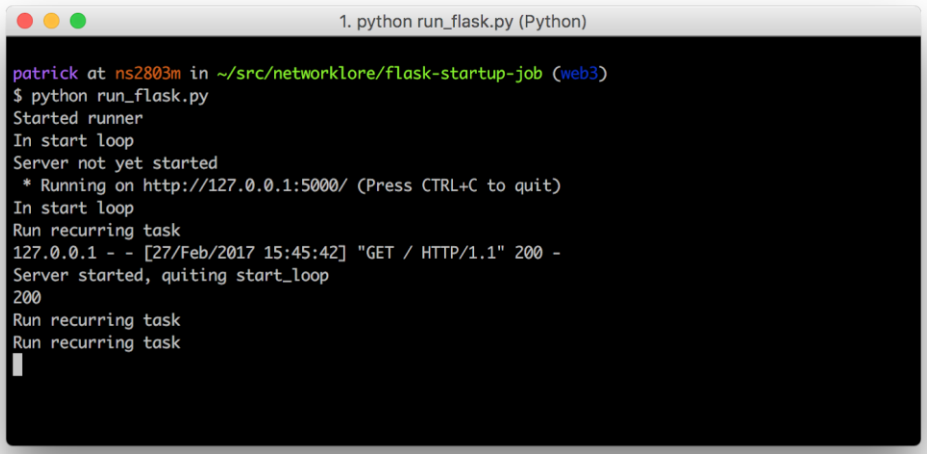


Рис. 4.3. Запуск серверної частини

Запуск інтерфейсної частини програми:

1. Завантажити Node.js

   sudo apt-get install nodejs
2. Завантажити npm

   sudo apt-get install npm
3. Завантажити AngularJS

   npm install -g @angular/cli

Рис. 4.4. Запуск фронт-енд частини

Висновки

PyAudio – кросплатформенна бібліотека мови Python, для відтворення та запису звуку на різних платформах. За її допомогою в рамках задачі розпізнавання звуку певної частоти можливо аналізувати довжину аудіохвилі, побудувати графік звукової хвилі та працювати з аудіофайлами формату wave.

Front-end частина програми написана за допомогою TypeScript із використанням фреймворку Angular 2. Back-end частина написана на мові програмування Python. Серверна частина написана засобами NodeJS.

ВИСНОВКИ

В даній роботі був проведений аналіз існуючих алгоритмів та систем розпізнавання аудіо подій. Була проведена порівняльна характеристика таких алгоритмів розпізнавання, як алгоритм динамічної трансформації часової шкали, прихована марковська модель, векторне квантування, метод опорних векторів, використання нейронних мереж та модель гаусових сумішей. Були виділені переваги та недоліки кожного алгоритму. Було детально досліджено етапи розпізнавання звуку, та проведений аналіз звукових характеристик. Було проведено порівняння існуючих програмних систем з цього сегменту, та виділено переваги та недоліки, серед яких складний інтерфейс, відсутність локалізації та підтримки продукту. Було сформовано детальний список вимог для розробки додатку. Була розроблена програма розпізнавання аудіо подій з використанням алгоритму розпізнавання звуку високої частоти. Програмний продукт було розроблено з використанням мови програмування Python, інтерфейсна складова була написана за допомогою мови програмування TypeScript. Були наведені результати тестування, які показують результати роботи модулів програми.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Davies, K. H., Biddulph, R. and Balashek, S. (1952) Automatic Speech Recognition of Spoken Digits, J. Acoust. Soc. Am. 24 (6) 637 — 642 с

2. Microsoft Speech SDK 5.4. Документація.

3. Ing-Jr Ding, Chih-Ta Yen, Yen-Ming Hsu. Development so Machine Learning Schemes for Dynamic Time-Wrapping-Based Speech Recognition // Mathematical Problems in Engineering. 2013.

4. Daniel Ram age. Hidden Markov Models Fundamentals // CS229 Section Notes. 2007.

5. Кульбак С. Теория информации и статистика. М.: Наука, 1967. 408 с.

6. X. Huang, A. Acero, H. Hon. Spoken language processing: a guide to theory, algorithm, and system development. – Prentice Hall PTR, 2001. P. 936.

7. ГОСТ Р 51061-97. ПАРАМЕТРЫ КАЧЕСТВА РЕЧИ. СИСТЕМЫ НИЗКОСКОРОСТНОЙ ПЕРЕДАЧИ РЕЧИ ПО ЦИФРОВЫМ КАНАЛАМ. Архивировано 30 апреля 2013 года.

8. И. А. Шалимов, М. А. Бессонов Анализ состояния и перспектив развития технологий определения языка аудиосообщения.

9. Мазуренко И. Л. — Компьютерные системы распознавания речи – 102 с

10. Forum S. Digital Speech Processing, Synthesis and Recognition // Marcel Dekker, New York, 1989.

11. ShotSpotter: Gunshot Detection [Electronic resource] / http://www.shotspotter.com/.

12. P. K. Atrey. Audio based event detection for multimedia surveillance.

13. F. Capman. Abnormal audio event detection.

14. Микулович В.И. Цифровая обработка сигналов. БГУ, 2011.

15. L. Gerosa, G. Valenzise, M. Tagliasacchi, F. Antonacci, A. Sarti. Scream and gunshot detection in noisy environments. Dipartimento di Elettronica e Informazione, Politecnico di Milano.

16. S. Ntalampiras. On acoustic surveillance of hazardous situations.

17. Вороновский Г. К., Махотило К. В., Петрашев С. Н., Сергеев С. А. Генетические алгоритмы, искусственные нейронные сети и проблемы виртуальной реальности. — Харьков: Основа, 1997. — 112 с. — ISBN 5- 7768-0293-8.
18. Зорич В. А. Математический анализ. — М.: Физматлит, 1984.

ДОДАТОК А. ЛІСТИНГ КОДУ ЗАСТОСУНКУ

FrontEnd:

```
import { environment } from './../environments/environment';
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { PanelComponent } from './panel/panel.component';
import { PlayerComponent } from './player/player.component';
import { VisualizerComponent } from './visualizer/visualizer.component';
import { PlayButtonComponent } from './play-button/play-button.component';
import { HeaderComponent } from './header/header.component';
import { GoogleMapsModule } from '@angular/google-maps'
import { MapServiceComponent } from 'src/_services/maps.service';
import { HttpClient} from '@angular/common/http';
import { HttpClientModule } from '@angular/common/http';
import { AgmCoreModule, GoogleMapsAPIWrapper, CircleManager } from '@agm/core';
@NgModule({
  declarations: [
    AppComponent,
```

```
  PanelComponent,
  PlayerComponent,
  VisualizerComponent,
  PlayButtonComponent,
  HeaderComponent
],
imports: [
  BrowserModule,
  AppRoutingModule,
  GoogleMapsModule,
  HttpClientModule,
  AgmCoreModule.forRoot({
    apiKey: environment.mapsApiKey
  })
],
providers: [
  MapServiceComponent,
  GoogleMapsAPIWrapper,
  CircleManager
],
```

```
  bootstrap: [AppComponent]
})
export class AppModule { }
```

FrequencyFinderThread

```
import threading
import time
from queue import Queue
import matplotlib.pyplot as plt
import numpy as np
import pyaudio
from webserv.SignatureFinderThread import SignatureFinderThread
np.set_printoptions(suppress=True)
CHUNK = 882  # number of data points to read at a time
RATE = 44100  # time resolution of the recording device (Hz)
MICROPHONE_NAME = "Samson C01U Pro"
class FrequencyFinderThread(threading.Thread):
    def __init__(self, entities: list):
        threading.Thread.__init__(self)
        self.p = pyaudio.PyAudio()  # start the PyAudio class
```

```
        microphone_index = None
        for i in range(self.p.get_device_count()):
            device_info = self.p.get_device_info_by_index(i)
            if MICROPHONE_NAME in device_info.get("name"):
                microphone_index = device_info.get("index")
                print("Found microphone index")
                print(device_info)
        if microphone_index is None:
            raise Exception("Failed to find microphone
{}".format(MICROPHONE_NAME))
        self.microphone_index = microphone_index
        self.entities = entities
    def run(self):
        stream = self.p.open(format=pyaudio.paInt16,
input_device_index=self.microphone_index, channels=1, rate=RATE,
                             input=True, frames_per_buffer=CHUNK)
        plt.axis([None, None, 0, 4000])
        start_t = time.time()
        count = 0
        chunks_queue = Queue()
```

```
        signature_processor_thread = SignatureFinderThread(chunks_queue,
self.entities)
        signature_processor_thread.setDaemon(True)
        signature_processor_thread.start()
        while True:
            data_b = stream.read(CHUNK, exception_on_overflow=False)
            data = np.fromstring(data_b, dtype=np.int16)
            data = data * np.hanning(len(data))  # smooth the FFT by windowing
data
            fft = abs(np.fft.fft(data).real)
            fft = fft[:int(len(fft) / 2)]  # keep only first half
            freq = np.fft.fftfreq(CHUNK, 1.0 / RATE)
            freq = freq[:int(len(freq) / 2)]  # keep only first half
            count = count + 1
            current_t = time.time()
            chunks_queue.put((fft, data_b))
            plt.plot(fft, freq)
            plt.pause(0.005)
        # close the stream gracefully
        stream.stop_stream()
        stream.close()
```

```
        self.p.terminate()
```

SignatureFinderThread

```
import threading
from datetime import datetime
from queue import Queue
import numpy
class SignatureFinderThread(threading.Thread):
    lat = [50.4491242, 50.4525763, 50.4361837]
    long = [30.5341315, 30.4689006, 30.5210961]
    def __init__(self, queue: Queue, entities: list):
        threading.Thread.__init__(self)
        self.queue = queue
        self.id = 0
        self.entities = entities
        self.index = 0
    def run(self):
        print("Started signature finder")
        time = None
        while True:
```

```
fft, data_b = self.queue.get(block=True)
freq = 0
for x, value in numpy.ndenumerate(fft):
    value_int = int(value)
    if (value_int > 10_000) and (2500 < freq < 3000):
        if time is None:
            time = datetime.now()
        else:
            curr = datetime.now()
            if (curr - time).total_seconds() < 2:
                time = curr
                continue
            else:
                time = curr
        print("Found new sound {} / {}".format(value_int, freq))
        print(datetime.now().second)
        entity = {
            "id": self.id,
            "date": datetime.now().strftime("%I:%M%p on %B %d, %Y"),
            "gun": "Shotgun",
```

```
            "lat": SignatureFinderThread.lat[self.index],
            "long": SignatureFinderThread.long[self.index],
            "file": "shot.wav"
        }
        self.entities.append(entity)
        self.id = self.id + 1
        if self.index == 2:
            self.index = 0
        else:
            self.index = self.index + 1
        '"print("freq {}".format(freq))
        print(value_int)'"
    freq = freq + 50
```