Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)

Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів та моделей автоматичного розпізнавання учасників аудіо розмови
(тема)

Виконав:
студент (ка) 2 курсу, групи ІПЗм-22-6

Андрєєв І.Г.
(прізвище, ініціали)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)

Тип програми освітньо-наукова

Керівник доц. Каук В.І.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ З.В.Дудар
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерних наук
Кафедра програмної інженерії
Рівень вищої освіти другий (магістерський)
Спеціальність 121 – Інженерія програмного забезпечення
Тип програми освітньо-наукова програма
Освітня програма Інженерія програмного забезпечення
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Андрєєву Івану Георгійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи «Дослідження методів та моделей автоматичного розпізнавання учасників аудіо розмови»
Затверджена наказом по університету від 29.03.2024р. № 250 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 20.06.2024
3. Вихідні дані до роботи набір дата сетів для навчання, тестування та реальної перевірки, інформація щодо типів нейронних мереж, існуючі моделі розпізнавання промовців, технології PyTorch, мова програмування Python3 з допоміжними бібліотеками та залежні до них, середовище розробки PyCharm CE 2024
4. Перелік питань, що потрібно опрацювати в роботі
аналіз предметної галузі дослідження, визначення проблеми та постановки задачі, порівняння основних існуючих моделей та сервісів з розпізнавання учасників розмови, планування та проведення експериментальних досліджень, написання програмних рішень, аналіз отриманих результатів.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі та постановка задачі	23.01 – 10.02.24	виконано
2	Аналіз існуючих сервісів	11.02 – 24.02.24	виконано
3	Формування потенційної схеми системи розпізнавання промовців	26.02 – 7.03.24	виконано
4	Планування експериментів	8.03 – 15.03.24	виконано
5	Програмна реалізація модулю розпізнавання промовців	16.03 – 10.04.24	виконано
6	Експериментальні дослідження	11.04 – 25.04.24	виконано
7	Аналіз результатів експериментальних досліджень та розробка рекомендацій	26.04 – 31.04.24	виконано
8	Написання та оформлення тез доповіді	1.04 – 15.04.24	виконано
9	Підготовка пояснювальної записки	16.04 – 31.05.24	виконано
10	Підготовка презентації та доповіді	1.06 – 7.06.24	виконано
11	Нормоконтроль	7.06 – 12.06.24	виконано
12	Рецензування	12.06 – 16.06.2024	виконано
13	Занесення диплома в електронний архів	16.06.2024	виконано
14	Попередній захист	17.06.2024	виконано
15	Допуск до захисту у зав. кафедри	18.06.2024	виконано

Дата видачі завдання 20 січня 2024р.

Студент _____ Андрєєв І.Г.
(підпис)

Керівник роботи _____ доц. Каук В.І.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ / ABSTRACT

Пояснювальна записка містить: 68 с., 43 рис., 7 табл., 21 джерело.

АУДІО АНАЛІЗ, ІДЕНТИФІКАЦІЯ ПРОМОВЦЯ, КЛАСИФІКАЦІЯ, КЛАСТЕРИЗАЦІЯ, МАШИННЕ НАВЧАННЯ, НЕЙРОННІ МЕРЕЖІ, СЕГМЕНТАЦІЯ, СИСТЕМА ДІАРИЗАЦІЇ.

Об'єктом дослідження є процес розпізнавання учасників розмови (Speaker Diarization).

Мета роботи – дослідження етапів процесу розпізнавання учасників розмови, аналіз методів та моделей машинного навчання та нейронних мереж, а також сучасних фреймворків навчання моделей розпізнавання мовлення та промовців.

На основі проведеної роботи був проведений аналіз існуючих моделей та системи розпізнавання мовлення та промовців, наведена порівняльна характеристика деяких існуючих систем розпізнавання промовців, побудована первинна схема системи та компонентів для створення власного програмного рішення, визначені етапи експериментів та складові елементи магістерського дослідження, проведено тестування існуючих бібліотек, як базового механізму процесу.

Базуючись на теоретичному опрацюванні матеріалів та існуючих досліджень було проведено дослідження щодо ефективності побудови та використання нейронної мережі розпізнавання промовців, порівняна результативність її роботи з різними наборами даних та сформовані відповідні результати та потенційні рекомендації щодо подальших досліджень.

MACHINE LEARNING, AUDIO ANALYSIS, DIARIZATION SYSTEM, CLASSIFICATION, CLUSTERIZATION, SPEAKER IDENTIFICATION, NEURAL NETWORKS, SEGMENTATION.

The object of the study is the process of speaker recognition (Speaker Diarization).

The purpose of the work is to study the stages of the process of speech recognition and speakers, the analysis of methods and models of machine learning and neural networks, as well as modern frameworks for training speech recognition models and diarization.

On the basis of the work carried out, an analysis of existing models and systems of speech and speaker recognition was carried out, comparative characteristics of some existing systems of speaker recognition were given, an initial scheme of the system and components for the creation of its own software solution was built, stages of experiments and components of master's research were determined, testing of existing libraries was carried out , as the basic mechanism of the process.

Based on the theoretical processing of materials and existing research, a study was conducted on the effectiveness of the construction and use of a neural network for speaker recognition, the effectiveness of its work with different data sets was compared, and relevant results and potential recommendations for further research were formed.

Я, Андрєєв Іван Георгійович, студент гр. ІПЗм-22-6, здобувач вищої освіти на другому (магістерському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Дослідження методів та моделей автоматичного розпізнавання учасників аудіо розмови», що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений(на) з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ASR – Automatic Speech Recognition

CNN – Convolutional Neural Network

EER – Equal Error Rate

EM – Expectation Maximization

FAR – False Acceptance Rate

FRR – False Rejection Rate

GMM – Gaussian Mixture Model

HMM – Hidden Markov model

MFCC – Mel-Frequency Cepstrum

MSDD – Multiscale Diarization Decoder

LSTM – Long Short-Term Memory

NLP – Natural Language Processing

RNN – Recurrent Neural Network

ROC – Receiver Operating Characteristic

RTTM – Rich Transcription Time Marked

SD – Speaker Diarization

STT – Speech-To-Text

VAD – Voice activity detection

ВСТУП

У сучасному цифровому світі, де аудіо розмови і мовлення є невід'ємною частиною нашого повсякденного життя, проблема розпізнавання учасників розмови стає все більш актуальною і важливою. Ця проблема має великий потенціал застосування в різних сферах, включаючи телефонну комунікацію, відеоконференції, медіа і багато інших галузей.

Машинне навчання та нейронні мережі є ключовими компонентами сучасних процесів розпізнавання мовлення та визначення учасників розмови. Машинне навчання дозволяє моделі навчатися на основі великої кількості даних та робити прогнози, а нейронні мережі, зокрема глибокі нейронні мережі, є потужними інструментами для обробки аудіо даних та розпізнавання мовлення.

Завданням цієї науково-дослідницької роботи є дослідження методів та моделей автоматичного розпізнавання учасників аудіо-розмови з використанням машинного навчання та нейронних мереж і як наступний етап, побудова своєї моделі для розпізнавання промовців.

Наступним кроком буде встановлення шляхів покращення окремих етапів та всього процесу розпізнавання учасників розмови загалом. Для цього цілями дослідження є аналіз найбільш прогресивних моделей та підходів до процесу розпізнавання промовців та виділення можливостей щодо їх удосконалення на основі різних наборів даних та параметрів безпосереднього навчання нейронних мереж.

Результат цієї роботи пропонується як потенційно вагома складова для розвитку технологій обробки аудіо даних та розпізнавання мовлення. Ціль даного дослідження це використати можливість краще зрозуміти тематику та галузь, а також спробувати застосувати знання з області машинного навчання та нейронних мереж для вирішення реальних завдань автоматичного розпізнавання учасників розмови та удосконалення цього процесу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Визначення проблеми та постановка задачі

Необхідність у вдосконаленні систем розпізнавання учасників аудіо розмов стає дуже важливим з ряду причин: сучасний світ генерує надзвичайно великий об'єм аудіо та відеоданих і важливо зробити ці дані доступними та аналізованими; віртуальні асистенти, технології голосового управління та інші, автоматичне розпізнавання мови допомагає підвищити продуктивність та ефективність бізнес-процесів; автоматичне розпізнавання мови може бути критичним для виявлення аномалій, визначення протоколів та ідентифікації осіб, тощо.

Мета магістерського дослідження – дослідження моделей та підходів щодо розпізнавання мови та учасників розмови шляхом використання основних та доступних сервісів, а також створення власної моделі та її подальша оптимізація з урахуванням порівняння результатів роботи існуючих систем.

Для цього необхідно буде виконати наступні дії:

– проаналізувати методи розпізнавання учасників аудіо розмов, моделі та алгоритми, які використовуються при цьому процесі;
– зробити огляд готових сервіси для визначення схожості підходів та алгоритмів їх дії;
– визначити сильні та слабкі сторони кожного з підходів, для подальшого планування системи та експерименту;
– в ході магістерського дослідження використати найбільш оптимальну існуючу модель з її подальшою модифікацією та навчанням. Порівняти результати власного підходу з результатами готових сервісів для визначення його ефективності.

1.2 Дослідження галузі розпізнавання учасників аудіо розмови

Розпізнавання промовця (SP) – це процес поділу окремих промовців в аудіо потоці таким чином, щоб у розшифровці автоматичного розпізнавання мовлення (ASR) висловлювання кожного промовця були розділені. Кожен промовець розділений своїми унікальними звуковими характеристиками, а їхні

висловлювання об’єднані разом. Цей тип функції також можна назвати мітками промовців або виявленням зміни промовців [3]. Розпізнавання промовця використовується, щоб покращити читабельність стенограми та краще зрозуміти, про що йде мова. Розпізнавання промовця може допомогти виділити з розмови важливі моменти чи дії та визначити, хто що сказав. Це також допомагає визначити, скільки промовців було на аудіо.

Покращення процесу розпізнавання учасників розмови та оптимізація використання цих технологій можуть дати позитивний вплив у таких напрямах:

- аутентифікація та безпека: застосування аудіо-розмов у різних галузях, таких як фінанси, медицина та бізнес, підкреслює важливість точної ідентифікації учасників. Питання безпеки стає особливо актуальним у контексті конфіденційності та захисту від несанкціонованого доступу;
- розмовний штучний інтелект: голосовий робот, який замовляє їжу, намагається визначити, хто розміщує замовлення їжі, коли розмовляють кілька дорослих і дітей;
- навчання: транскрипція запитань і відповідей студента для аналізу відповідей, наданих професором або студентами;
- здоров’я: окремі коментарі пацієнтів і лікарів як для особистих зустрічей, так і для телефонних консультацій;
- ідентифікація шахраїв: зростаюча кількість шахрайських дій в аудіо-комунікаціях, таких як фінансові афери чи соціальний інжиніринг, вимагає вдосконалення методів виявлення несанкціонованого доступу та обману;
- підтримка продажів: відстеження того, хто що сказав на зустрічі з продавцями, і навчання продавця, що говорити та коли мовчати;
- аналіз доповідача: відстежуйте поточні та попередні коментарі певного доповідача під час зустрічей або відстежуйте час розмови під час телефонної розмови;
- управління аудіо/відео/подкастами: стенограми або субтитри, розділені доповідачами, дозволяють легше шукати атрибуцію компанії/продукту

та краще розуміти глядачів або слухачів.: з ростом можливостей машинного навчання та обробки природної мови, є великий потенціал для створення більш точних та швидких систем розпізнавання учасників аудіо розмов;

– ділові узгодженості: визначення того, що клієнт погодився на певні ділові умови під час зустрічі з кількома особами.

Перейдемо від аналізу викликів та напрямів застосування до визначення як автоматичні системи транскрипції мовлення забезпечують розпізнавання промовця.

До основних підходів визначення хто та коли виступив з промовою можна виділити такі:

– підхід під наглядом (класифікація): модель навчена розпізнавати обмежену кількість промовців (тобто класів). Таким чином, очікується, що вона працює лише з промовцями, які використовувалися серед навчальних даних. Наприклад, транскрипція щотижневої зустрічі команди, система навчена розпізнавати різних членів команди, коли обробка аудіо залучає завжди тих самих промовців (класи). Цей підхід менш гнучкий, але може бути дуже ефективним, особливо для великої кількості промовців (>5);

– підхід без нагляду (кластеризація): модель групує аудіо сегменти відповідно до промовця на основі виділених аудіо характеристик. Це найбільш універсальний підхід, оскільки він може визначити кількість залучених промовців (кількість кластерів) і призначити кожен голосовий сегмент певному кластеру. Наприклад, аналітика колл-центру, система працює для будь-якого промовця, який бере участь у розмові, і вона не має попередніх знань про учасників розмови, будь то оператор чи клієнт [20].

У контексті розпізнавання промовця з підходом без нагляду (найпоширеніший підхід) послідовність виконання включає кілька підзавдань для

виявлення голосової активності та сегментації аудіо на мовні та немовні сегменти, а потім групування їх відповідно до промовця (див. рис. 1.1).

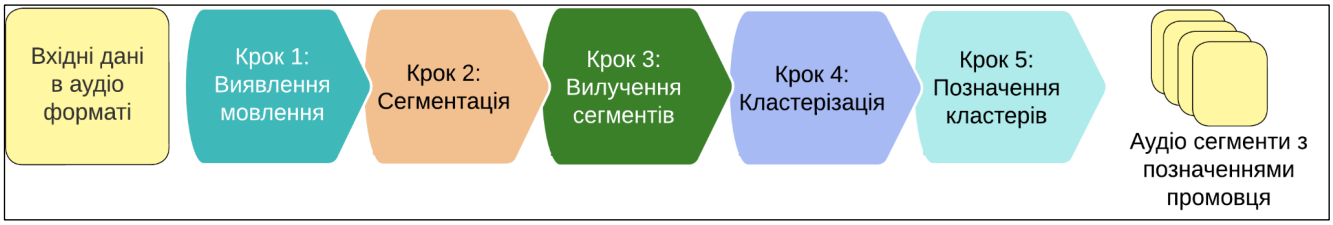


Рисунок 1.1 – Послідовність кроків розпізнавання промовців з використанням підходу без нагляду (виконано самостійно)

У базовому визначенні, процес розпізнавання промовців зумовлює розбиття сегментів на підсегменти. Зробимо опис для наведених кроків:

– виявлення мовлення - на цьому етапі використовується технологія для відділення мовлення від фонового шуму аудіозапису;
– сегментація мовлення - цей крок передбачає виділення невеликих сегментів аудіо файлу. Зазвичай для кожного промовця є сегмент тривалістю приблизно одну секунду;
– вилучення внутрішніх сегментів мовлення - цей крок передбачає розміщення всіх внутрішніх сегментів мови, створених і зібраних на другому кроці, для подальшого створення нейронної мережі для цих сегментів. Потім ці сегменти можна перевести в інші формати даних, такі як текст, зображення, документи тощо. Ці різні типи даних потім можуть використовуватися для глибокого навчання;
– кластеризація - після визначення внутрішніх сегментів, наступний крок передбачає їх кластеризацію;
– позначення кластерів - після створення кластерів, вони позначаються, як правило, кількістю промовців;
– транскрипція (розширення базового функціоналу) - коли кластери створено та позначено належним чином, аудіо можна сегментувати на окремі кліпи для кожного промовця. Потім ці кліпи надсилаються через

програму (STT) або систему розпізнавання мовлення, яка видає транскрипцію.

Ранні підходи до розпізнавання мовців базувалися на техніці цифрової обробки сигналів, а пізніше – на основі статистичного моделювання (HMM, GMM, тощо) [4].

Загальним вибором щодо підходів на основі моделі є застосування прихованої Марковської моделі (HMM) і моделі суміші Гауса (GMM). GMM часто вибирають, тому що вони є універсальними апроксиматорами щільності, тобто вони можуть моделювати довільну функцію розподілу ймовірностей за даними. Модель GMM використовує гауссівські розподіли для моделювання акустичних характеристик звуків та фонем мови. Наведемо формулу 1.1 розподілу GMM:

$$p(x) = \sum_{i=1}^{K} \pi_i \mathcal{N}(x|\mu_i, \Sigma_i) \tag{1.1}$$

де x - акустичний вектор;

K - кількість гауссових компонентів;

π_j - ваги компонентів. Вказують на важливість кожного гауссового компонента у суміші. Вони вказують на ймовірність того, що вибірка даних належить до певного компонента;

μ_i - середнє значення гауссового компонента. У контексті мовлення, це може представляти середній спектральний профіль особливостей голосу;

Σ_i - коваріаційні матриці компонентів. Визначає розсіяність даних навколо середнього. У розпізнаванні мовлення, це описує, наскільки великі варіації у характеристиках голосу для кожного компонента.

EM-алгоритм використовується для навчання параметрів GMM на основі тренувальних даних. Алгоритм починається з початкових значень параметрів та потім ітеративно виконує два кроки: E-крок (очікування), де обчислюється очікувана ймовірність належності кожного спостереження до кожної компоненти

GMM, та M-крок (максимізація), де оновлюються параметри GMM на основі очікуваних значень [21].

Прихована модель Маркова (HMM) у контексті розпізнавання мовлення використовується для моделювання тимчасових послідовностей і зв'язку між різними звуками у мовленні. HMM дозволяє моделювати мовлення як послідовність станів, кожен з яких відповідає певному звуку або фонемі. У цій моделі спостереження залежать від поточного стану, але не від попередніх станів або спостережень. Це дозволяє HMM ефективно моделювати мовлення з урахуванням його динамічної та непередбачуваної природи. Наведемо формулу 1.2 для HMM моделі:

$$P(O|\lambda) = \sum_{i=1}^{N} \pi_i \cdot b_i(O) \cdot \prod_{t=1}^{T} a_{i,i_{t+1}} \quad (1.2)$$

де P(O|λ) - загальна ймовірність спостережуваної послідовності O при заданих параметрах моделі λ;

O - спостережувана послідовність, яка складається з акустичних векторів або інших спостережуваних характеристик;

λ - параметри HMM;

N - кількість станів в HMM (відповідає можливим учасникам розмови);

πi - початкова ймовірність перебування в стані i на початку послідовності (визначають початкові ймовірності перебування учасників в розмові.);

bi(O) - ймовірність спостереження послідовності O при перебуванні в стані i (може відображати ймовірність того, що акустичні характеристики O відповідають учаснику i);

T - довжина спостережуваної послідовності;

it - стан в момент часу t;

ai,it+1 - ймовірність переходу зі стану i в стан it+1 в момент часу t (визначає ймовірності переходу від одного учасника до іншого в часі).

Алгоритм Baum-Welch використовується для навчання параметрів HMM на основі тренувальних послідовностей. Він включає в себе обчислення апостеріорних ймовірностей, що вказують, які стани відповідають кожному елементу спостереження, а потім оновлює параметри моделі на основі цих апостеріорних ймовірностей.

Більш прогресивні та сучасні методи розпізнавання промовці спираються на розширене глибоке навчання та нейронні мережі [14].

1.3 Аналіз типів нейронних мереж у розпізнаванні мовців

Під час огляду сучасного стану галузі та методів розпізнавання учасників розмови було визначено що глибоке навчання та нейронні мережі, на рівні із базовим машинним навчанням, є найбільш передовими та мають найбільший потенціал для подальшого розвитку. Найбільш популярні сервіси з розпізнавання мови та визначення учасників розмови базують навчання своїх моделей на основі згорткових (CNN) і рекурентних типів нейронних мережах (RNN) [11]. Для нашого дослідження будемо також націлені на використання цих типів. Ще одним плюсом в користь їх використання є підтримка AWS SageMaker цих типів навчання. Для кращого розуміння, як буде виконуватися робота наведемо опис цих типів.

Згорткові нейронні мережі і рекурентні нейронні мережі є двома основними типами нейронних мереж, які широко використовуються в глибокому навчанні. Хоча обидва є потужними інструментами для обробки та аналізу даних, вони мають різні архітектури та підходять для різних типів завдань.

CNN має кілька рівнів ієрархії, що складаються з шарів маршрутизації та рівнів групування, які визначаються великою різноманітністю діаграм. Загалом CNN починається зі згорткового рівня, який приймає дані вхідного рівня. Для згорткових операцій із кількома картами фільтрів однакової розмірності відповідає рівень згортки [8]. Крім того, вихід із цього шару переноситься на шар зразка, що зменшує масштаб наступних шарів. CNN локально пов’язаний із великою різноманітністю методів глибокого навчання. Потім ці мережі

реалізуються на основі архітектури GPU на кількох сотнях ядрах. Рольові карти будуть розподілені на основі блоків знань попереднього рівня. Це залежить від розмірів карт. Однак кожен потік зв’язаний з одним нейроном за допомогою одного блоку з багатьох потоків. Подібним чином згортання нейронів, індукція та підсумовування виконуються протягом решти методу [18]. Нарешті, глобальна пам'ять зберігає продуктивність вищевказаних процесів. Для ефективної обробки результатів використовується зворотна модель і модель поширення. Однак окреме поширення не дасть позитивних результатів, тому операції витягування або переміщення сприяють паралельному розповсюдженню. Крім того, нейрони одного шару взаємодіють з окремою кількістю нейронів, впливаючи на граничні ефекти.

Наведемо загальну архітектуру CNN, яка описує роботу цієї нейронної мережі глибокого навчання (див. рис. 1.2).

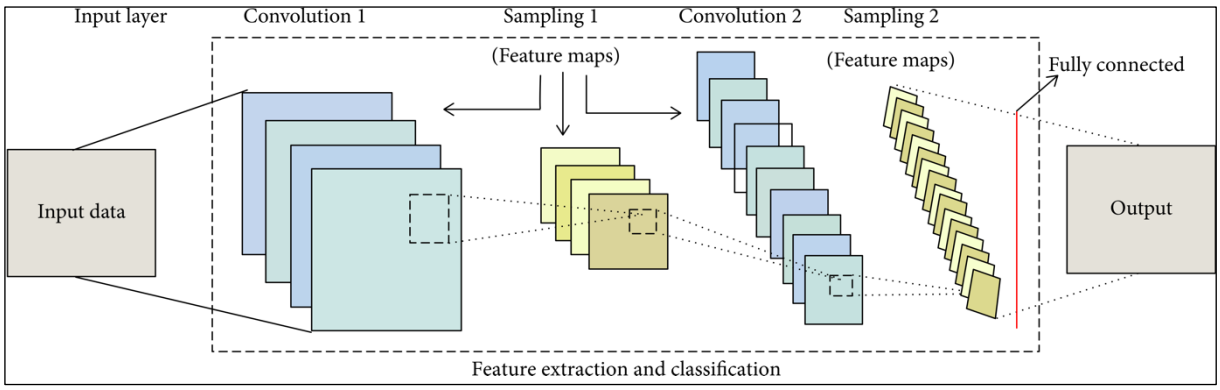


Рисунок 1.2 – Архітектура CNN (посилання [15])

Алгоритм глибокого навчання включає попередню обробку вхідних даних, навчання моделі глибокого навчання, зберігання вивченої моделі та останню фазу реалізації моделі. CNN особливо ефективні в обробці сіткових даних, таких як зображення чи аудіо сигнали. Вони призначені для фіксації просторових зв’язків і локальних моделей у даних. CNN складаються зі згорткових шарів, шарів об’єднання та повністю зв’язаних шарів. Згорткові шари застосовують фільтри до вхідних даних, вилучаючи релевантні функції в різних просторових масштабах. Об’єднання шарів зменшує дискретизацію карт функцій, що зменшує розмірність.

У завершенні, повністю з’єднані шари обробляють витягнуті функції та створюють бажаний результат.

Стосовно іншої моделі, RNN спеціально розроблені для послідовної обробки даних. Вони добре підходять для завдань із використанням даних часових рядів або природної мови. RNN використовують повторювані з’єднання, які дозволяють інформації зберігатися та перетікати від одного кроку до наступного. Це дозволяє їм фіксувати тимчасові залежності та послідовні шаблони в даних [9]. RNN мають приховані стани, які зберігають пам’ять про минулі вхідні дані, що робить їх здатними навчатися з історичного контексту. Наведемо схему базової RNN архітектури (див. рис. 1.3).

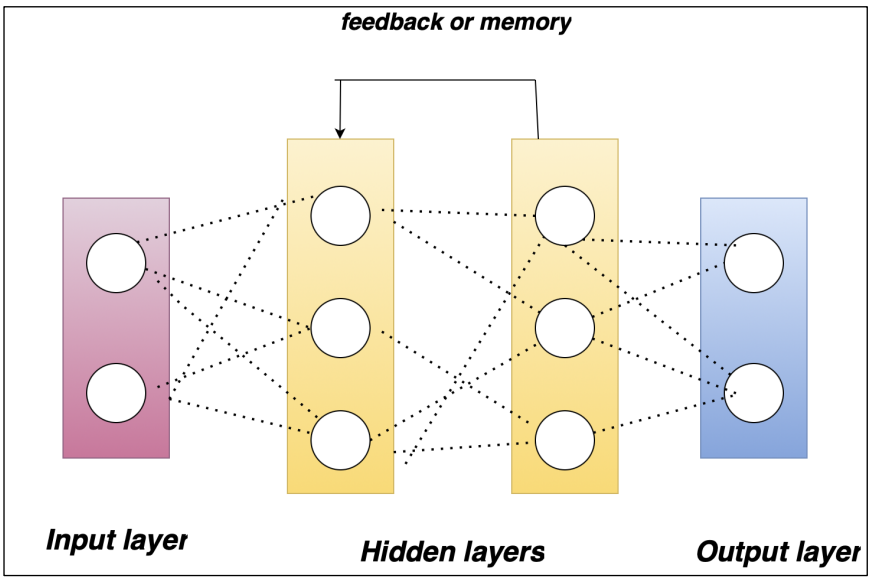


Рисунок 1.3 – Архітектура RNN (посилання [13])

CNN використовують фільтри всередині згорткових шарів для перетворення даних, тоді як RNN є прогнозними, повторно використовуючи функції активації з інших точок даних у послідовності для генерації наступного виходу в серії.

1.4 Огляд існуючих продуктів та сервісів

На ринку галузі існує багато продуктів та сервісів для розпізнавання промовців. Кожен з них має свої особливості, підходи та переваги. Виділимо сім сервісів, які за опосередкованим значенням, є найбільш популярними та

розвинутими в сфері розпізнавання мови та промовців, а також відносяться до стимуляторів прогресу цього напрямку:

– Amazon Transcribe від AWS;
– Speech Service від MS Azure;
– Speech-to-Text від Google Cloud;
– AssemblyAI;
– IBM Watson.

Для наглядного порівняння деяких властивостей та можливостей цих сервісів, які мають певні відмінності, наведемо їх порівняльну характеристику (див. табл. 1.1).

Таблиця 1.1 – Характеристики сервісів розпізнавання мови та промовців (таблиця виконана самостійно)

Аспект	Amazon Transcribe	Speech Service	Speech-to-Text	AssemblyAI	IBM Watson
Модифікація	Словник, пунктуація, ідентифікація мови	Використання кастомної моделі	Зміна голосу, адаптація мови	Аналіз настроїв, виявлення теми	Адаптація до специфічної лексики
Сфери використання	Контакт-центри, створення медіа-контенту	Різні напрями у підприємництві	Системи навігації, онлайн зустрічі	Віртуальні асистенти, чат боти	Бізнес-аналітика, маркетинг
Особливі властивості	Генеративне підсумування дзвінків, рівень впевненості слів	Кастомні нейронні моделі голосу, переклад у реальному часі	Маркування аудіо даних	Використання аудіо інтелекту	Виявлення ключових слів, визначення емоцій
Легкість використання	Мінімальні зміни для існуючих клієнтів	Простота використання для розробників	Спрощена обробка та аналіз даних	Проста інтеграція з API	Зручний інтерфейс та інтеграція

Так як більшість сучасних сервісів мають схожі властивості та способи тренування моделей, не будемо виділяти їх при аналізі. Натомість наведемо опис моделей та алгоритмів, які використовуються цими сервісами. Алгоритми та моделі, які використовуються різними службами, для розпізнавання промовців часто складні й базуються на передових техніках машинного навчання. Хоча детальна специфіка цих алгоритмів зазвичай є запатентованою та повністю не розкритою, можна описати деякі загальні підходи, які використовуються в цих службах:

- AWS (Amazon Transcribe): розпізнавання промовців AWS базується на передових моделях нейронних мереж, навчених на великих наборах даних. Вони зосереджені на визначенні чітких голосів мовців за допомогою таких функцій, як тон, шаблони мовлення та, можливо, методи глибокого навчання для більш точного розділення мовців;
- Speech Service (Microsoft Azure): технологія розпізнавання промовців Azure може використовувати аналогічні підходи до нейронних мереж, інтегруючи їх із когнітивними службами. Вони можуть використовувати такі методи, як кластеризація та виділення вбудованих промовців, щоб розрізняти їх поміж іншими;
- Speech-To-Text (Google Cloud): функція розпізнавання промовців є частиною API перетворення мовлення в текст, заснована на глибоких нейронних мережах. Вони використовують моделі виявлення голосової активності (VAD) і розпізнавання промовців, щоб сегментувати та позначати аудіо на основі різних учасників розмови;
- AssemblyAI: система розпізнавання мовців AssemblyAI, використовує комбінацію моделей розпізнавання мовлення та обробки природної мови. Вони можуть використовувати такі методи, як алгоритми вбудовування та кластеризації для розділення промовців;
- IBM Watson Speech to Text: технологія розпізнавання мовців IBM, включає вдосконалені моделі машинного навчання, включаючи

кластеризацію та нейронні мережі, для ідентифікації та розрізнення промовців у аудіо файлі.

Загалом у більшості цих сервісів використовується поєднання методів виявлення голосової активності, розпізнавання мовлення, визначення промовців та методів кластеризації. Вони використовують вдосконалені нейронні мережі, такі як CNN і RNN, навчені на великих наборах даних, щоб точно ідентифікувати та розрізняти різних промовців у сегменті аудіо.

2 ПРОЕКТУВАННЯ СИСТЕМИ РОЗПІЗНАВАННЯ ПРОМОВЦІВ

2.1 Опис інфраструктури системи

Одним з вихідних результатів виконання магістерського дослідження та проведення експерименту є моделювати потенційної системи розпізнавання промовців, завдяки якій буде можливо виконувати комплексні та паралельні процеси з опрацювання аудіо файлів. Враховуючи технічні потреби та властивості машинного навчання одним з найкращим місцем з достатніми технічними характеристиками та забезпечувальними елементами інфраструктури є хмарні провайдери. Беручи до уваги актуальні дані кращих провайдерів хмарних сервісів та враховуючи існуючий досвід роботи, для побудови схеми інфраструктури системи було обрано AWS Cloud провайдера.

Для створення схеми системи на базі AWS Cloud для навчання та тестування моделі розпізнавання учасників аудіо розмови, наведемо список та опис потенційних сервісів. Архітектура системи буде складатися з:

- зберігання необхідних файлів для роботи системи у Amazon S3. Amazon Simple Storage Service (S3) надає можливість зберігати велику кількість даних із високою доступністю та надійністю. Файли аудіо записів, моделі для тренування, набори даних для навчання та тестування можуть зберігатись в S3 бакетах (свого роду контейнерах);
- додаткові обчислення та обробка запитів або подій у AWS Lambda trigger та Amazon S3 event - поєднання сервісу для попередньої обробки даних з сервісом зберігання файлів. Автоматична обробка аудіо файлів при їх завантаженні в S3. Використовуючи Lambda функції, можна автоматизувати процес обробки аудіо, наприклад, конвертування форматів, видалення шумів, визначення мета-інформації та багато інших доповнюючих операцій;
- навчання моделі у Amazon SageMaker - навчання та оцінка моделей машинного навчання. SageMaker дозволяє легко створювати, навчати та розгортати моделі машинного навчання. Можна використовувати попередньо побудовані алгоритми або писати власні. SageMaker також

надає можливість проведення експериментів, автоматизації та оптимізації процесу навчання [17];

- тестування та оцінка моделі у Amazon SageMaker. Після навчання моделі в SageMaker, можна використати тестові набори даних для оцінки її ефективності та точності. SageMaker надає зручні інструменти для оцінки та аналізу результатів моделі;
- розгортання моделі у серверній оболонці у Amazon EC2 елементах або ECS контейнерах для вирішення реальних завдань. За допомогою EC2 можна розгорнути сервери для обробки запитів на розпізнавання мовців у реальному часі. Для мінімізації конфігурування та підтримки серверних налаштувань можна використати ECS контейтери з типом Fargate (на базі Docker контейнеризації). Для менш затратних задач можна використовувати AWS Lambda образи;
- ефективне управління повідомленнями між компонентами системи за допомогою AWS Simple Queue Service (SQS). За допомогою черги подій система зможе безпечно передавати дані між різними частинами у вигляді повідомлень. Це може бути використано для забезпечення послідовної обробки завдань, таких як передача інформації про завантажені аудіо файли, виклик конвертування файлів чи ділення його на малі відрізки та багато іншого;
- сповіщення про важливі події у системі за допомогою AWS Simple Notification Service (SNS). Це дозволить надсилати сповіщення, наприклад, коли модель успішно пройшла процес навчання або коли виявлено помилку і дії адміністратора мають бути виконані перед наступною ітерацією. Також це може бути використано для інформування команди розробників або автоматизації наступних кроків в обробці даних;
- зберігання та швидкий доступ до неструктурованих даних за допомогою AWS DynamoDB. Цей ресурс - це швидка і гнучка NoSQL база даних. Вона може бути використана для зберігання метаданих про аудіо файли,

результатів навчання, та іншої інформації, яка вимагає швидкого доступу та високої доступності. Також хмарні технології забезпечують підтримку даних за рахунок якісного механізму шифрування для цілісності та безпеки даних [2];

– створення та управління API через AWS API Gateway. API Gateway дозволяє створювати, публікувати, підтримувати, слідкувати та захищати API будь-якого масштабу. Ми зможемо використати його для створення API, через який користувачі або інші системи можуть завантажувати аудіо файли для обробки, отримувати статус обробки, або доступатися до результатів розпізнавання;
– моніторинг та запис логів роботи системи та дій користувачів за допомогою Amazon CloudWatch. CloudWatch дозволяє стежити за роботою системи, включаючи ефективність EC2 машин, Lambda функцій та SageMaker. Можна налаштувати сповіщення про помилки або нестандартну поведінку системи.

Наведемо високорівневу схему потенційної системи розпізнавання промовців (див. рис. 2.1).

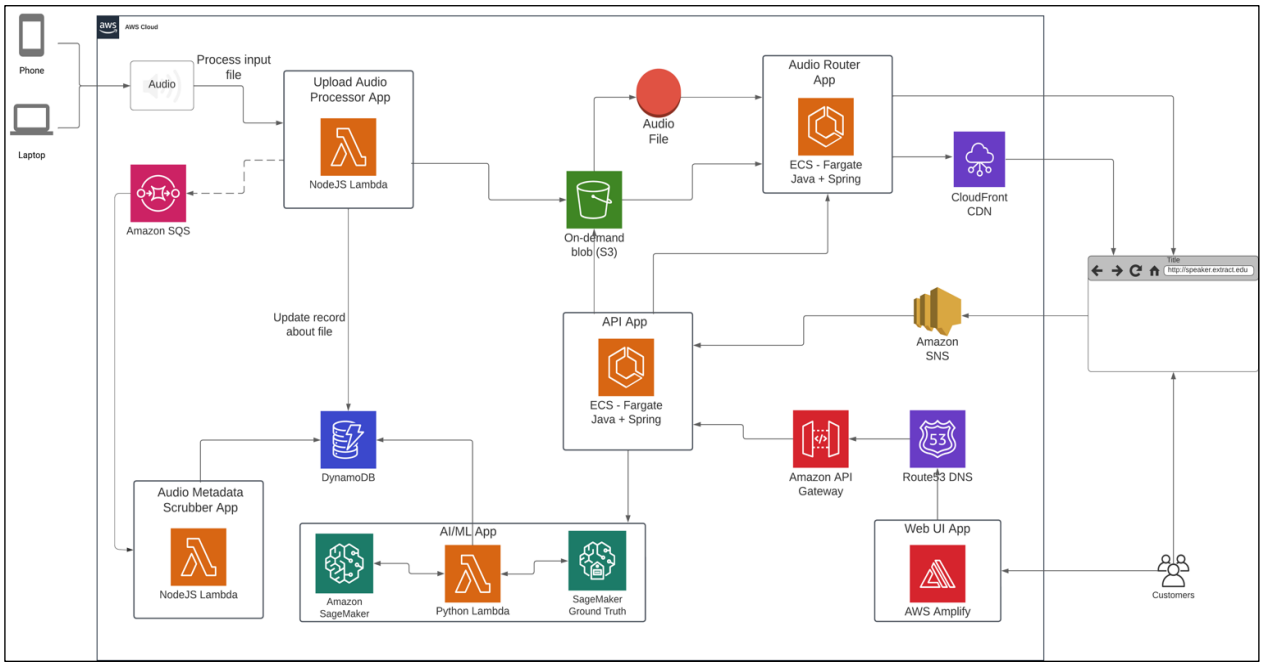


Рисунок 2.1 – Високорівнева схема побудови системи для розпізнавання учасників розмови з додатковими ресурсами (виконано самостійно)

Використання AWS сервісів забезпечує масштабованість, гнучкість та ефективність обробки даних та навчання моделей.

2.2 Огляд фреймворків для системи розпізнавання учасників розмови

Покращення розпізнавання промовця зараз є добре відомою та досить загальною проблемою. Таким чином, можна легко використовувати вже існуючі фреймворки та моделі та адаптувати їх до конкретних випадків використання.

Так як мова програмування Python зараз відноситься до базових мов у використанні машинного навчання, наведемо список одних з популярних фреймворков для розпізнавання промовця з підтримкою Python: Pyannote.audio, NVIDIA NeMo та Reseblyzer [5]. Всі вони мають відносно легку інтеграцію з бібліотеками Python та PyTorch. PyTorch - це відкрита бібліотека машинного навчання та обчислень на основі глибокого навчання.

Проаналізуємо перші два фреймворки та визначимо компоненти, які використовуються на різних етапах розпізнавання промовців (див. табл. 2.1).

Таблиця 2.1 – Порівняння компонентів на різних етапах розпізнавання учасників розмови (посилання [7])

Етапи	Pyannote	NeMo
Виявлення мовлення	Pyannet	MarbleNet
Вилучення сегментів	ECAPA-TDNN	TitaNet
Кластерізація	Прихована Марковська модель (HMM)	Багато масштабна кластеризація (MSDD)

Загалом, фреймворки використовують ту саму базову послідовність, приймаючи аудіо файли як вхідні дані та генеруючи файли RTTM, як вихідні дані. Можна зауважити, що NeMo є значно більшою бібліотекою для розробки додатків автоматичного розпізнавання мовлення (ASR) і, отже, постачається з чітко визначеною структурою для навчання, оцінювання та висновків за допомогою

моделей, тоді як Pyannote розроблено в основному для розпізнавання промовців і здається більш придатним за метою нашого дослідження [1].

Модель виявлення мовлення від Pyannote, Pyannet, опрацьовує необроблений аудіосигнал, тоді як MarbleNet від NeMo приймає Mel-спектрограму як вхідні дані (стандартний вхід для ASR і загалом завдань обробки аудіо).

Моделі вилучення сегментів дуже схожі на згорткові нейронні мережі (CNN). Однак завдання кластеризації від Nemo базується на багато масштабній кластеризації, що означає, що воно використовує кілька прихованих представлень аудіо вхідного сигналу в різних масштабах часу, а потім виконує кластеризацію на основі множинного прихованого вбудовування голосів [6]. Pyannote має більш простий підхід зі стандартною прихованою моделлю Маркова.

Представимо порівняння сильних та слабких сторін кожного фреймворку на основі існуючих досліджень (див. табл. 2.2)

Таблиця 2.2 – Порівняння сильних та слабких сторін фреймворків (посилання [7])

Характеристика	Pyannote	NeMo
Доступні попередньо навчені моделі	+	+
Якісна сегментація промовців за декількома мітками	+	-
Легкість виконання ASR та NLP завдань	-	+
Можливість вказати кількість промовців як параметр припущення	+	+
Автоматичне визначення кількості промовців	+	+
Доступні моделі для специфічних випадків	-	+
Кастомізація послідовності етапів	-	+

Для дослідження може бути використані і інші бібліотеки з більш підходящими характеристиками для спеціальних потреб та поставлених задач.

3 ПЛАНУВАННЯ ДОСЛІДЖЕННЯ

3.1 Опис та етапи проведення дослідження

Перейдемо до планування проведення дослідження враховуючи проведений аналіз моделей та базових шаблонів у цій галузі. Попередній план дослідження має наступні фази:

а) визначення цілі - чітко визначити масштаби та контекст проведення розпізнавання промовця (наприклад, розпізнавання під час зустрічей, телефонних дзвінків, тощо). На базі цього можна буде зупинитись на типах нейронних мережевих підходів та новіших моделях глибокого навчання;

б) збір та анотація даних:
   1) для збору аудіо даних можна використовувати записи телефонних розмов, аудіозаписи співбесід, подкасти або відео-аудіо файли;
   2) приділити увагу треба на різноманітність у кількості промовців, стилях мовлення та акустичному середовищі;
   3) можна використати наявні анотовані набори даних, якщо вони є, або провести анотацію вручну;

в) підготовка даних для навчання:
   1) визначення ознак з аудіо даних, такі як спектрограми або MFCC коефіцієнти, які представляють звукові характеристики в числовому вигляді, щоб передати їх на вхід нейронній мережі;
   2) встановлення розмітку даних з вказівкою, хто і коли говорив (час початку, час завершення, ідентифікатор учасника);
   3) аудіо файли можуть бути перетворені в текст для використання як учительські дані для навчання моделей або для оцінки точності;
   4) розділення набору даних для навчання, перевірки та тестування;

г) навчання моделі та налаштування параметрів:
   1) використання фреймворки Pyannote або NVIDIA NeMo для тренування моделі на підготовлених даних. Як альтернативу можна використати популярний фреймворк TensorFlow або PyTorch;

2) експериментування з різними гіперпараметрами, такими як швидкість навчання, розмір партії, кількість шарів тощо;
3) методи регулярізація: вилучення або збільшення даних, щоб запобігти перебільшення використання моделі з первинним набором та некоректну роботу для нового набору даних;

д) оцінка моделі:
1) помилкове прогнозування (false alarm) - передбачений сегмент мовлення, де немає промовця (хибний результат від моделі VAD);
2) аналіз показника частоти помилок розпізнавання промовця (DER) – базується на основі очікуваного та прогнозованого результату розпізнавання;
3) пропущені сегменти (missed detection);
4) помилкова кластерізація промовця (confusion);
5) перехресна перевірка для забезпечення надійності і можливості узагальнення моделі [12];

е) експериментування та ітерація:
1) ітеративне тестування через проведення кількох раундів тестування та налаштування моделі та її параметрів на основі продуктивності;
2) тестування A/B: порівняння різних моделей та конфігурації, щоб знайти найефективніший підхід;

ж) тестування в реальних умовах - тестування моделі на реальних аудіо даних або в реальних сценаріях використання;

з) формування висновків та оформлення результатів - підготовка звіту, включаючи результати, метрики, графіки, інструкції щодо використання та оптимізації моделі.

3.2 Приклади програмних елементів для практичних етапів дослідження

Як вже було визначено, процес розпізнавання мовлення та промовців може мати різні реалізації в залежності обраної моделі машинного навчання, типу нейронної мережі чи бібліотек з частковим або повним покриттям функціоналу.

Побудова програмної реалізації базувався на основі різних існуючих систем розпізнавання, тож можемо навести структуру функціональної реалізації основних етапів цього процесу на одному з цих прикладів [10].

Для початку наведемо опис шагів реалізації:

– виявлення голосової активності (VAD), цей крок визначає, чи містить сегмент аудіо мову чи ні;
– вилучення особливостей, тут визначаються значні специфіки з аудіо даних, які можна використовувати для розрізнення промовців;
– кластеризація промовців, на цьому кроці подібні функції мовлення групуються разом, як правило, за допомогою алгоритму кластеризації. Кожен кластер ідеально представляє унікального промовця.

Беручи за основу цю спростовану послідовність наведемо приклад програмного коду на мові програмування Python з використанням бібліотек librosa для обробки аудіо файлу та sklearn для кластеризації промовців (див. рис. 3.1).

librosa_processor.py

```python
import librosa
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
import os

# Завантаження аудіофайлу
directory_path = os.path.join(os.path.expanduser("~"), 'Documents/Nure/Mag_robota/DataForProcessing')
audio_path = directory_path + '/Conference.wav'
audio, sr = librosa.load(audio_path, sr=None)

# Визначення mfcc ознак
mfccs = librosa.feature.mfcc(audio, sr=sr)

# Масштабування mfcc ознак
scaler = StandardScaler()
mfccs_scaled = scaler.fit_transform(mfccs.T)

# Кластеризація мовців
kmeans = KMeans(n_clusters=2)
speaker_labels = kmeans.fit_predict(mfccs_scaled)

for i, label in enumerate(speaker_labels):
    print(f"Time Segment {i}: Speaker {label}")
```

Рисунок 3.1 - Код з використанням librosa та sklearn (виконано самостійно)

На рисунку 3.1 відображені кроки для визначення промовці. Наведемо невеликий опис для них:

- MFCC - це невеликий набір характеристик (зазвичай близько 10–20), які описують загальну форму спектральної обвідної. Вони отримані з перетворення Фур'є логарифмічного спектра потужності звуку;
- StandardScaler використовується для стандартизації функцій шляхом видалення середнього значення та масштабування до одиничної дисперсії, що є загальною вимогою для багатьох оцінювачів машинного навчання;
- KMeans - неконтрольований (без нагляду) алгоритм машинного навчання, який групує дані в k кластери.

Наведений приклад програмного коду звісно не проводить повного визначення промовців та текстової транскрипції але надає приклад базової послідовності. Resemblyzer вдало використовується для виявлення мовлення, сегментації мовлення та вилучення вбудовування (закодовані представлення даних, які машини можуть інтерпретувати та які фіксують часову, просторову та контекстну інформацію залежно від програми).

3.3 Характеристика набору даних для проведення дослідження

Окрім самого машинного навчання та побудови моделі розпізнавання промовців одним з головних компонентів дослідження є набір даних. За базу дата сетів для проведення операцій з даними щодо навчання, перевірки результатів навчання та тестування реальних записів були обрані наступні відкриті набори даних:

- LibriSpeech – це великий набір аудіо записів та суміжних метаданих для розпізнавання мовлення, який базується на публічно доступних аудіо книгах з проекту LibriVox;
- VoxCeleb – це великий набір аудіо записів відомих людей, призначений для завдань розпізнавання промовців;
- CN-Celeb – це великий набір аудіо записів знаменитостей китайського походження, призначений для завдань розпізнавання промовців.

Порівняння характеристик цих дата сетів наведемо у форматі таблиці (див. табл. 3.1)

Таблиця 3.1 – Порівняння характеристик наборів даних (виконано самостійно)

Характеристика	LibriSpeech	VoxCeleb	CN-Celeb
Розмір (години)	1000	VoxCeleb1: 352	1300
Джерело	Аудіокниги	Інтерв'ю, новини, ток-шоу, YouTube	Телевізійні шоу, інтерв'ю, новини, документальні фільми, фільми
Якість звуку	16 кГц	16 кГц	16 кГц
Анотації	Точні текстові транскрипції	Метадані про мовців	Метадані про мовців, часові мітки для мовленнєвих сегментів
Різноманітність	Професійні диктори	Широкий спектр акцентів, стилів мовлення, умов запису	Широкий спектр умов запису, емоційних станів, акцентів, шумів
Переваги	Висока якість транскрипцій, транскрипцій, велика кількість годин мовлення	Велика різноманітність мовців та умов запису, реальні умови розпізнавання мовлення	Велика різноманітність умов запису, специфічна спрямованість на китайських мовців, великий обсяг

Кінець таблиці 3.1

Характеристика	LibriSpeech	VoxCeleb	CN-Celeb
			даних
Недоліки	Професійні диктори можуть не відображати реальні умови	Може містити шумові артефакти	Може містити шумові артефакти, обмежена універсальність для інших мов

LibriSpeech, VoxCeleb та CN-Celeb є цінними ресурсами для досліджень у галузі розпізнавання мовців та мовлення, кожен з яких має свої унікальні характеристики та переваги. У контексті даного дослідження були використанні здебільшого дані з LibriSpeech, насамперед для навчання та тестування моделей, у свою чергу VoxCeleb та CN-Celeb використовувались для тестування моделі та перевірки реальних записів на негативний сценарій.

Хоч наявні датасети і мають різноманітні комбінації доріжок, щоб покращити якість моделі для подальшої перевірки даних був використаний механізм аудіо аугментації для створення різноманітних варіацій звучання.

Аугментація аудіо доріжок може здійснюватися наступними методами:

– додавання шуму;
– зміна швидкості відтворення;
– зміна висоти тону;
– реверберація;
– часова зміна;
– комбінація методів.

Приклад програмного коду для аугментації аудіо даних наведено нижче (див. рис. 3.2).

audio_augmentation.py

```python
import numpy as np
import librosa
import soundfile as sf
import os

# Завантаження аудіофайлу
directory_path = os.path.join(os.path.expanduser("~"), 'Documents/Nure/Mag_robota/DataForProcessing')
file_path = directory_path + '/Conference.wav'
y, sr = librosa.load(file_path, sr=None)

# Додавання білого шуму
noise = np.random.randn(len(y))
y_noisy = y + 0.005 * noise

# Зміна швидкості відтворення
y_fast = librosa.effects.time_stretch(y, rate=1.25)
y_slow = librosa.effects.time_stretch(y, rate=0.75)

# Зміна висоти тону
y_pitch_up = librosa.effects.pitch_shift(y, sr=sr, n_steps=4)
y_pitch_down = librosa.effects.pitch_shift(y, sr=sr, n_steps=-4)

# Збереження результатів
sf.write(directory_path + '/Conference_noisy.wav', y_noisy, sr)
sf.write(directory_path + '/Conference_fast.wav', y_fast, sr)
sf.write(directory_path + '/Conference_slow.wav', y_slow, sr)
sf.write(directory_path + '/Conference_pitch_up.wav', y_pitch_up, sr)
sf.write(directory_path + '/Conference_pitch_down.wav', y_pitch_down, sr)
```

Рисунок 3.2 – Python код для аугментації аудіо даних з використанням librosa та soundfile бібліотек (виконано самостійно)

Використання цих наборів даних у поєднанні з додатковими змінами характеристик через процес аугментації націлені на покращення продуктивності моделей, роблячи їх більш універсальними та стійкими до різних умов запису і акцентів.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

4.1 Огляд компонентів машинного навчання у програмному продукті

Для навчання своєї моделі розпізнавання промовців був використаний фреймворк з відкритим кодом PyTorch, який базується на мові програмування Python. Для навчання нейронної мережі PyTorch використовується довга короткострокова пам'ять (LSTM). Як вже було вказано раніше, довга короткострокова пам’ять – це тип архітектури штучної рекурентної нейронної мережі (RNN). На відміну від стандартних нейронних мереж прямого зв’язку, LSTM має зворотний зв’язок, що дозволяє використовувати тимчасові залежності між послідовностями даних. LSTM розроблено для вирішення проблеми зникнення або розриву градієнтів, які можуть виникнути під час навчання традиційних RNN на послідовностях даних [19]. Завдяки цьому вони добре підходять для завдань, пов’язаних із послідовними даними, таких як обробка природної мови (NLP), розпізнавання мовлення та прогнозування часових рядів. Мережі LSTM представляють блоки пам’яті, які мають здатність зберігати інформацію протягом довгих послідовностей. Кожний блок пам'яті має три основні компоненти: вхідні ворота, ворота забування і вихідні ворота. Ці ворота допомагають регулювати потік інформації в блок пам’яті та з неї (див. рис. 4.1).

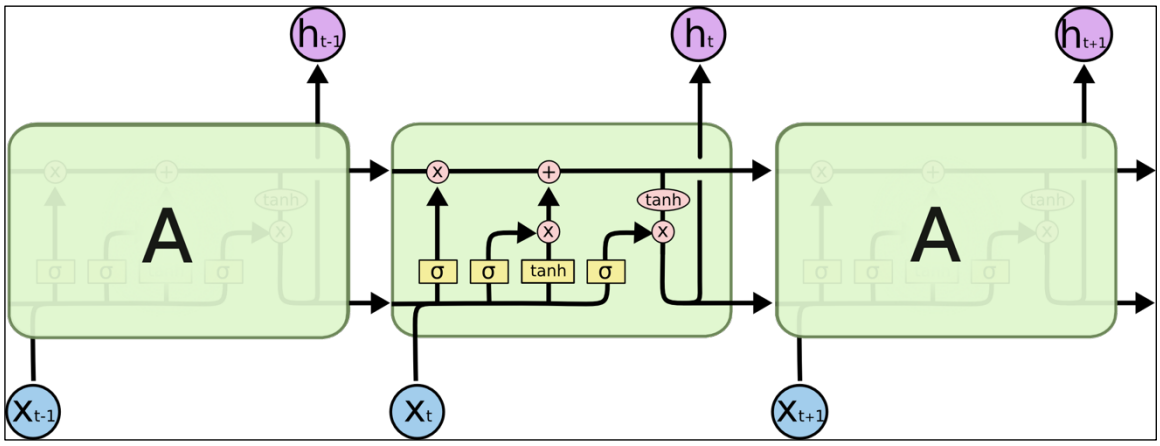


Рисунок 4.1 – Повторюваний модуль у LSTM (посилання [15])

Вхідні ворота визначають, скільки вхідної інформації має зберігатися в блоці пам'яті. Вони приймають поточну вхідну інформацію та попередній

прихований стан як вхідні параметри та виводить значення від 0 до 1 для кожного елемента блоку пам’яті.

Ворота забування вирішують, яку інформацію відкинути з блоку пам’яті. Як і вхідні ворота, ворота забування приймають поточну вхідну інформацію і попередній прихований стан та виводить значення від 0 до 1 для кожного елемента блоку пам’яті. Значення 0 означає, що інформація ігнорується, а значення 1 означає, що вона зберігається. Вихідні ворота контролюють, скільки вмісту комірки пам’яті слід використовувати для обчислення прихованого стану. Використовуючи ці ворота, мережі LSTM можуть вибірково зберігати, оновлювати та діставати інформацію в довгих послідовностях. Це робить їх особливо ефективними для завдань, які вимагають моделювання довгострокових залежностей, таких як розпізнавання мовлення, переклад мови та аналіз настроїв [16].

Підготовка вхідних даних для LSTM передбачає формування даних у форматі, який модель LSTM може отримати та ефективно обробити. Основні кроки для підготовки вхідних даних для LSTM такі:

– організація вхідних даних в послідовності фіксованої довжини. Наприклад, у контексті часових рядів, якщо є щоденні дані, можна створити послідовності даних, скажімо, за 10 днів як одну вхідну послідовність;
– зміна форми даних у 3D-форматі: зразки, часові кроки, характеристики. Наприклад, якщо ваші дані мають форму 2D-матриці: зразки та характеристики, потрібно змінити їх так, щоб LSTM міг інтерпретувати їх як послідовності даних;
– зразки: кількість точок даних у наборі даних;
– часові кроки: кількість часових кроків у кожній послідовності;
– характеристики: кількість характеристик на кожному кроці часу.

Щоб перетворити звичайний набір даних у формат, придатний для введення в модель LSTM, зазвичай потрібно реструктурувати дані в послідовності, щоб сформувати вхідні характеристики та цільові значення.

Наступним кроком навчання моделі є визначення погрішностей на кожному шагі та вдосконалення моделі в наступних. Для цього було обрано тип функції втрати – триплетні втрати. Вона використовується, щоб знайти, який вектор подібний до опорного вектору та призначена для вимірювання продуктивності моделі, яка може виявляти подібні точки даних. Для визначення подібності використовується косинусна подібність, щоб визначити, наскільки «близькими» є два фрагменти звуку за їхніми характеристиками.

Триплетна втрата працює, порівнюючи точки схожості як із схожим позитивним вектором (схожим на опорний вектор), так і з несхожим негативним вектором (не схожим на опорний вектор). Мета полягає в тому, щоб мінімізувати відстань між опорним вектором та позитивним вектором, одночасно максимізуючи відстань між опорним вектором та негативним вектором (див. рис. 4.2).

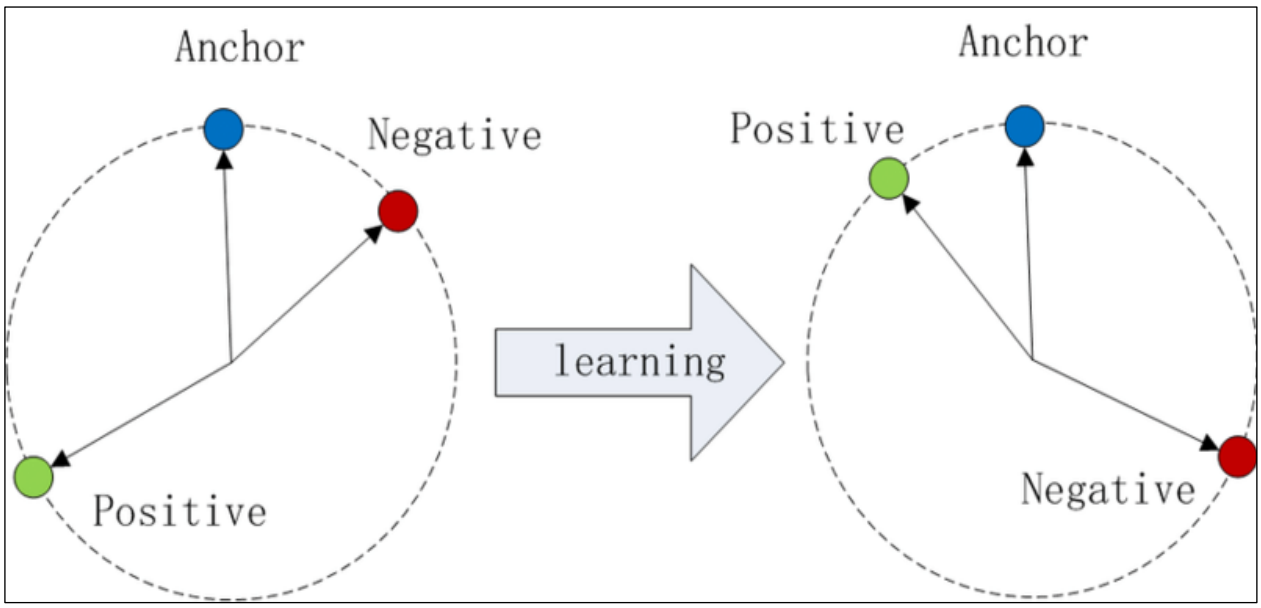


Рисунок 4.2 - Триплетна втрата з косинусною подібністю (посилання [13])

На основі машинного навчання з довгою короткостроковою пам’яттю та триплетною втратою і інших допоміжних обробок аудіо даних і планувалось створення моделі розпізнавання промовців.

4.2 Приклади програмного коду розпізнавання промовців

Основуючись на наведеному теоретичному опису компонентів машинного навчання приведемо частки програмного коду для різних етапів побудови, тестування та використання моделі на реальних даних.

Для початку виділимо структуру коду яка включає 3 основні папки: допоміжні засоби (utils), з існуючими моделями готових реалізацій (with_existing_model) та побудовою власної моделі (with_training_model). Коренева папка проекту включає ці внутрішні папки з функціональним кодом (див. рис. 4.3)

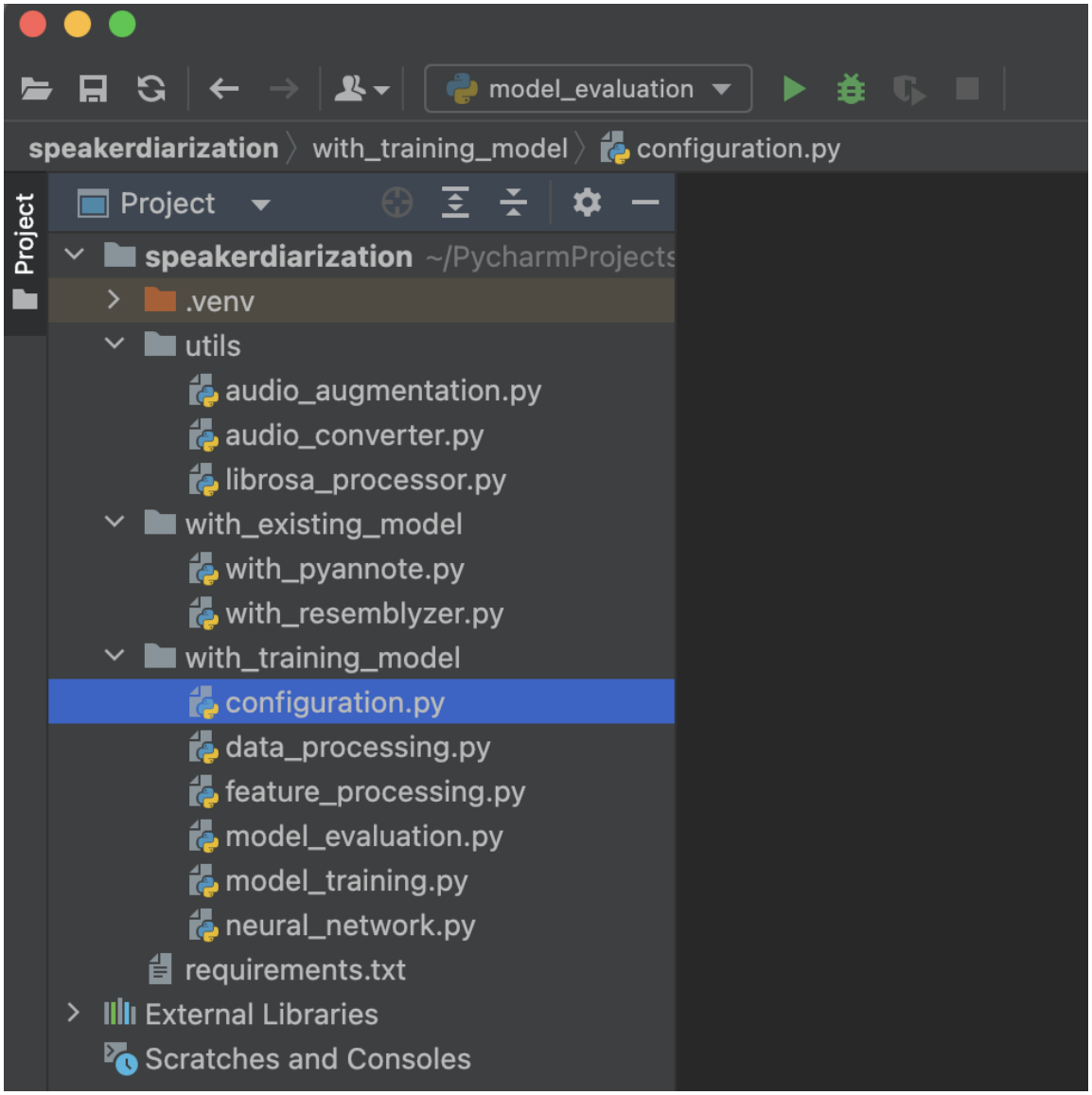


Рисунок 4.3 – Структура файлі коду Python для визначення промовців з існуючими моделями та створення своєї (виконано самостійно)

Допоміжні засоби призначені для обробки аудіо файлі з метою аугментації та конвертації у потрібні аудіо формати. Головна мета використання реалізацій з готовими моделями це порівняння їх результатів виконання зі власноруч створеною моделлю. Наведемо приклади програмного коду з використання готової моделі для версії Reseblyzer (див. рис. 4.4).

with_resemblyzer.py

```
encoder = VoiceEncoder()
_, cont_embeds, wav_splits = encoder.embed_utterance(wav, return_partials=True, rate=16)

print(f'cont_embeds = ${cont_embeds}')

clusterer = SpectralClusterer(
    min_clusters=3,
    max_clusters=6)

labels = clusterer.predict(cont_embeds)
print(f'labels = ${labels}')

labelling = create_labelling(labels, wav_splits)
labeled_speakers = build_speakers(labelling)

print(f'labelling = ${labeled_speakers}')

speaker_embeds = [encoder.embed_utterance(speaker_wav) for speaker_wav in speaker_wavs]
similarity_dict = {name: cont_embeds @ speaker_embed for name, speaker_embed in
                   zip(speaker_names, speaker_embeds)}

## Run the interactive demo
interactive_diarization(similarity_dict, wav, wav_splits)
```

Рисунок 4.4 - Код Python для визначення промовців з використанням готової моделі Reseblyzer (виконано самостійно)

На рисунку наведені кроки щодо декодування аудіо доріжки та визначення характеристик аудіо сегментів. На основі знаходження найближчих значень характеристик та відношення їх до середніх показників проводиться їх кластеризація для виділення специфічних груп, які шляхом подальшого позначення та маркування формують відрізки промовців визначенні моделлю машинного навчання. І на останньому кроці проводиться візуалізація оцінки впевненості в розпізнавання того чи іншого промовця. При використанні різних варіацій даних, включаючи ті самі записи але з різними модифікаціями щодо шуму на задньому фоні або швидкості запису, результати мають певні відхилення від сереньіх очікувань.

Далі продемонструємо програмний код з використання готової моделі для версії Pyannote (див. рис. 4.5).

with_pyannote.py

```python
from pyannote.audio import Pipeline
import os

pipeline = Pipeline.from_pretrained(
    checkpoint_path: "pyannote/speaker-diarization-3.1",
    use_auth_token="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")

# run the pipeline on an audio file
wav_fpath = os.path.join(
    os.path.expanduser("~"),
    'Documents/Nure/Mag_robota/DataForProcessing/Conference.wav'
)
diarization = pipeline(wav_fpath)

# dump the diarization output to disk using RTTM format
with open("Conference.rttm", "w") as rttm:
    diarization.write_rttm(rttm)

for turn, _, speaker in diarization.itertracks(yield_label=True):
    print(f"start={turn.start:.1f}s stop={turn.end:.1f}s speaker_{speaker}")
```

Рисунок 4.5 – Код Python для визначення промовців з використанням готової моделі Pyannote (виконано самостійно)

Відмінність цієї версії у порівнянні з Reseblyzer кодом полягає в прихованні подібних кроків з вилучення характеристик, кластеризації та маркування промовців в контексті «трубопроводів» (pipeline) – налаштованих послідовностей дій над вхідними даними та конфігураціями та повернення вже сформованих груп даних, які можуть бути використані для створення презентацій або подальшого аналізу та обробки. В наданому прикладі наведені дві дії з отриманими результатами, це створення файлу у форматі RTTM, популярний в роботі з розпізнавання мовців, та виведення результату у потік виведення записів на екран.

Для демонстрації використання готових моделей проведемо аналіз невеликих файлів аудіо запису конференції (див. рис. 4.6).

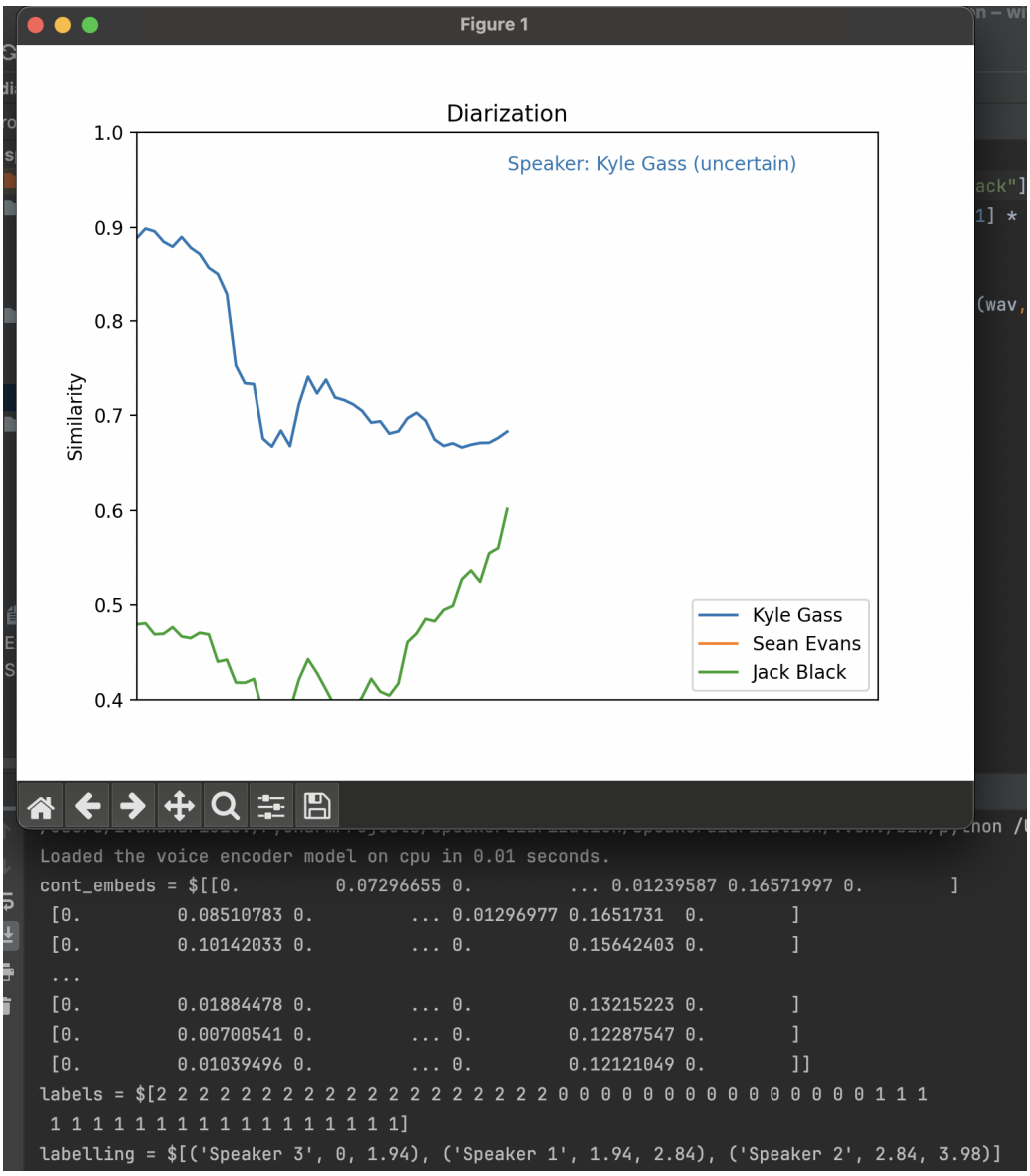


Рисунок 4.6 – Результат обробки аудіо запису конференції реалізаціями з готовими моделями (виконано самостійно)

Аналізуючи отриманий результат можна стверджувати, що певні ознаки аудіо характеристик одного з учасників аудіо запису мають більшу схожість з одним з трьох промовців, але рівень впевненості в цьому результаті є не достатнім, щоб повністю підтвердити його ідентичність. На основі цих результатів задля отримання більш точної відповіді треба проводити більше тестувань з різними аудіо записами цього промовця для знаходження позитивних збігів і інших промовців для виявлення негативних розбіжностей і звісно поліпшувати модель нейронної мережі шляхом варіативного навчання.

Далі перейдемо до шляху створення своєї моделі нейронної мережі з використанням PyTorch фреймворку. Серцем всіх операцій з даними є машинне навчання нейронної мережі (див. рис. 4.7)

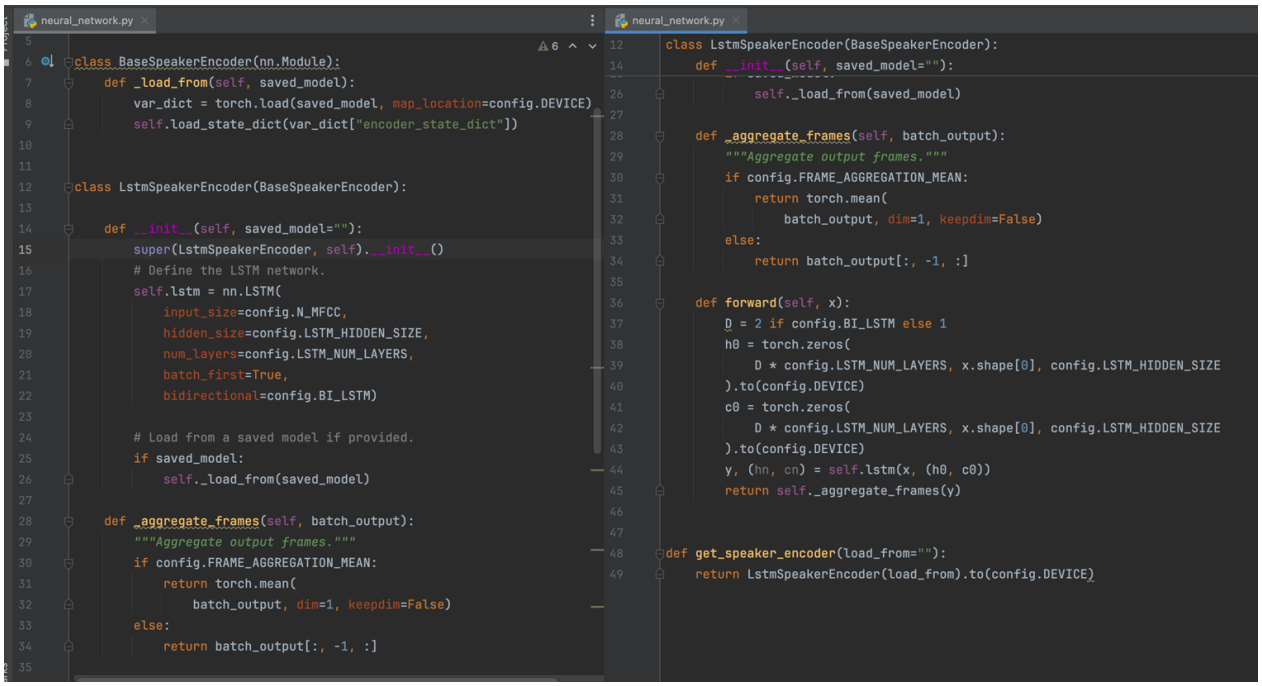


Рисунок 4.7 – Код для енкодування запису на основі LSTM (виконано самостійно)

Ці елементи коду використовуються для навчання моделі та її перевірки для визначення точності її роботи. Переходячи від одного інтервалу запису до іншого та використовуючи більше даних для покращення якості даних у пам’яті та їх багаторазового використання дає результат більш для створення більш надійної моделі. Всі програмні компоненти по створенню моделі та її перевірки використовують один конфігураційний файл, завдяки якому можна експериментувати з параметрами у різних ділянках процесу. У цьому коді ми можемо змінювати кількість прихованих шарів між вхідними даними та вихідним результатом.

Функціонал цього файлу використовуються як залежність у файлі навчання моделі (див. рис. 4.8)

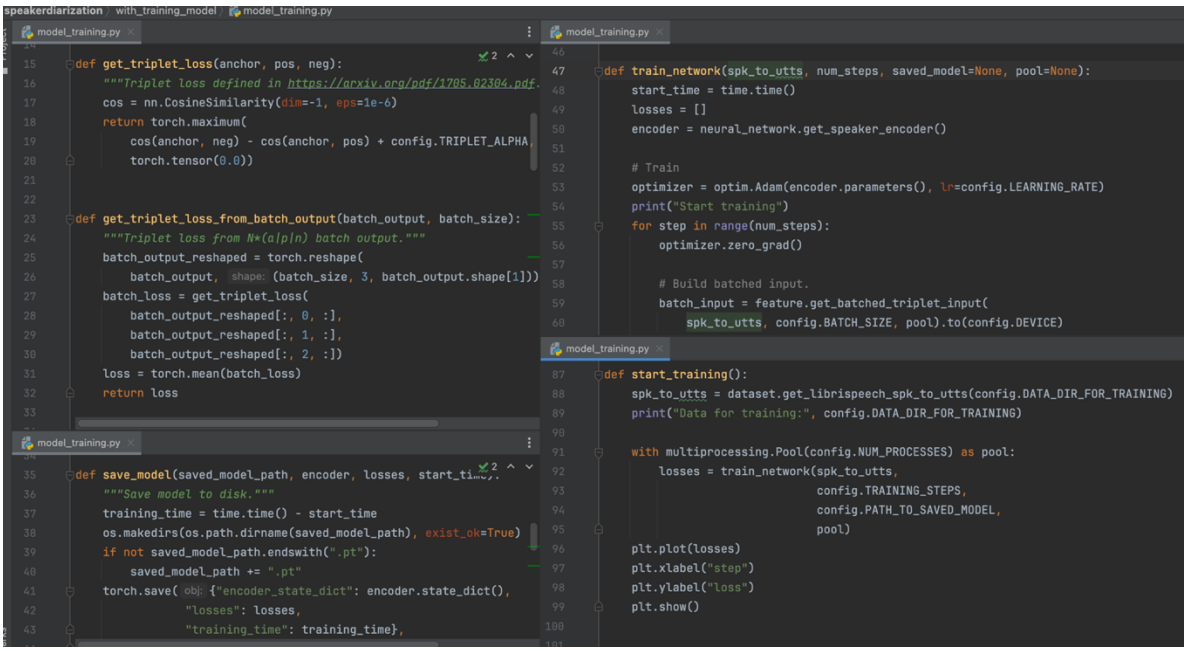


Рисунок 4.8 – Код для навчання моделі розпізнавання промовця (виконано самостійно)

В даному коді наведена реалізація раніше оговореної функції триплетної втрати з визначенням косинусної подібності векторів. Наведемо приклад результату тренування моделі при конфігурації у 100 кроків (див. рис. 4.9).

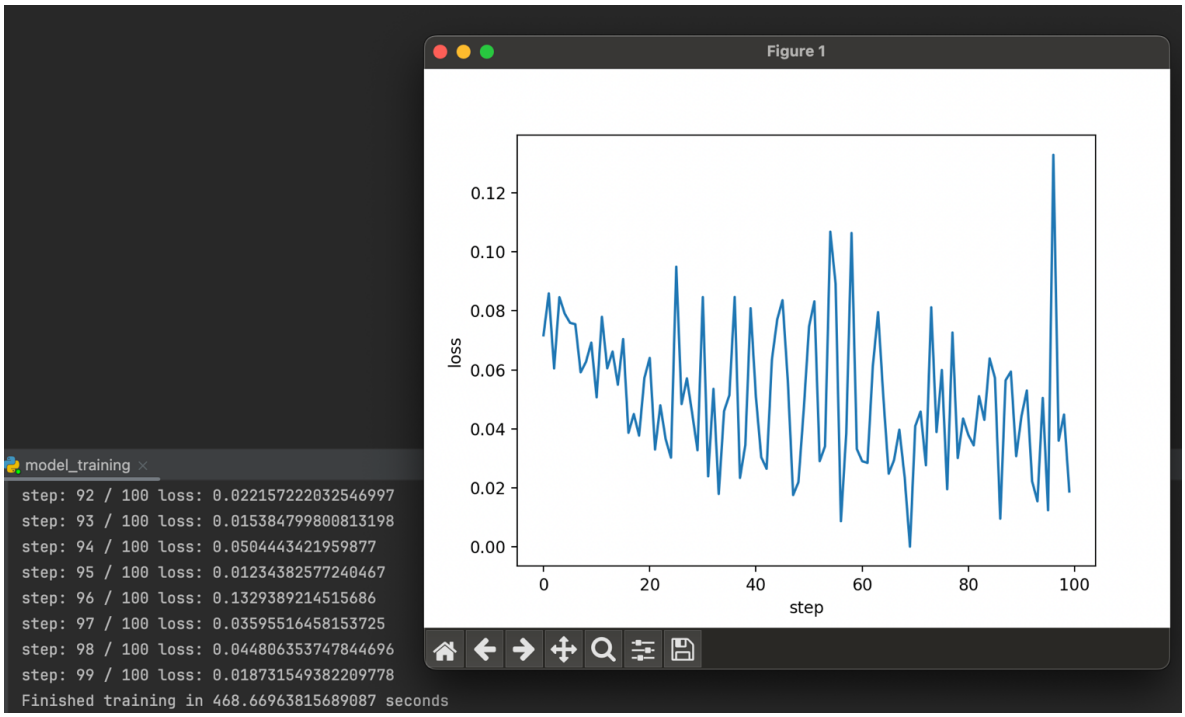


Рисунок 4.9 – Результати навчання моделі зі 100 кроками та отриманим рівнем функції втрати (виконано самостійно)

Як бачимо, при з кожним наступним кроком загальна тенденція йде на спад, хоча на деяких кроках результат функції втрати може бути сильно завищена, але одразу після порівняння з середнім значенням та визначенням некоректного напряму наступні результати мають тренд на зменшення цього показника і як результат підвищення якості створеної моделі.

Наступним кроком для оцінки надійності та коректності створеної моделі є її перевірка на тестовому наборі даних. Функціональність для тестування моделі наведена нижче (див. рис. 4.10).

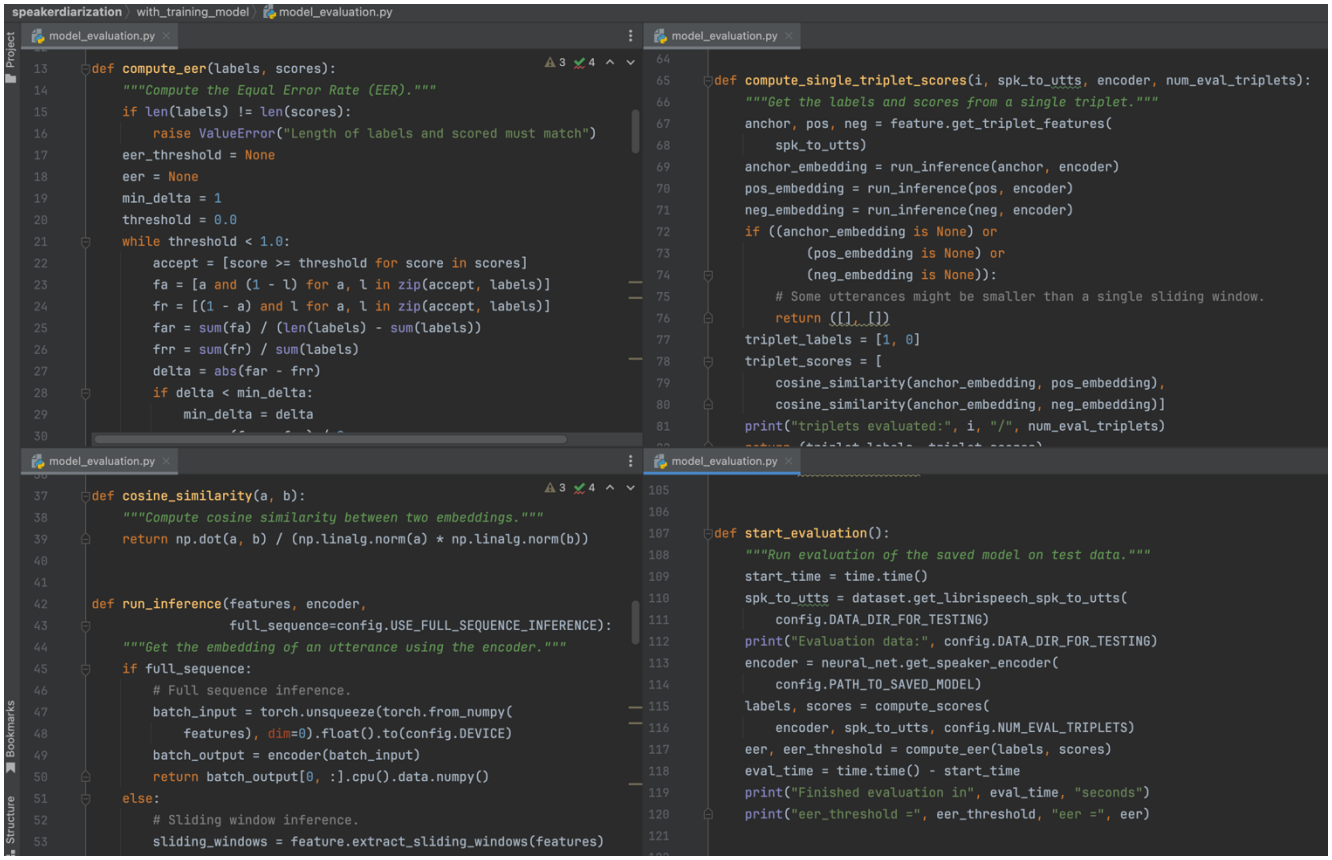


Рисунок 4.10 – Код для перевірки моделі розпізнавання промовця (виконано самостійно)

На цьому етапі також використовується загальний функціонал файлу нейронної мережі, як і для файлу тренування моделі, бо логіка визначення характеристик має бути однаковою. Результативним значенням цього процесу є показник EER – це метрика, яка використовується для оцінки продуктивності систем розпізнавання, зокрема систем розпізнавання мовців, біометричних систем

та систем безпеки. EER визначається як точка на кривій ROC, де частота помилкових прийнять (FAR) дорівнює частоті помилкових відхилень (FRR). Іншими словами, це точка, де кількість помилок прийняття (прийняття неправильного результату) дорівнює кількості помилок відхилення (відхилення правильного результату).

Наведемо приклад результату перевірки моделі при конфігурації у 1000 триплетів (див. рис. 4.11).

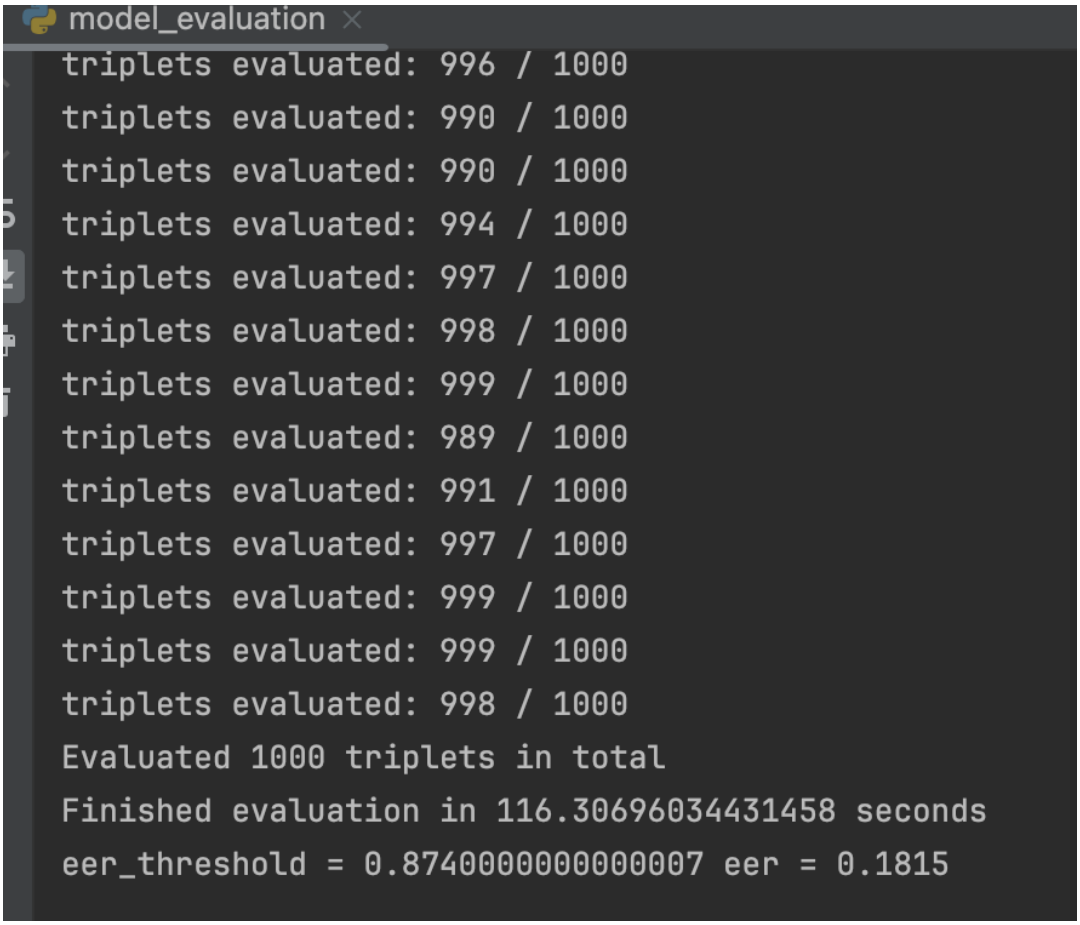


Рисунок 4.11 – Результати перевірки моделі із 1000 триплетами та отриманим рівнем EER (виконано самостійно)

При збільшені кількості перевірочних комбінацій даних (триплетів), кожна наступна перевірка дає більш оптимістичне значення EER, особливо якщо маркери мовців у даних перевірки частіше збігаються з маркерами мовців з навчального набору даних. Зміна показника EER має тенденцію на спад при кожною підвищенні конфігураційного параметра кількості триплетів.

Далі наведемо приклад коду для роботи з аудіо файлами, визначенням їх відрізків для аналізу, формування триплетів та оптимізаційна обробка цих даних в сформованих групах.

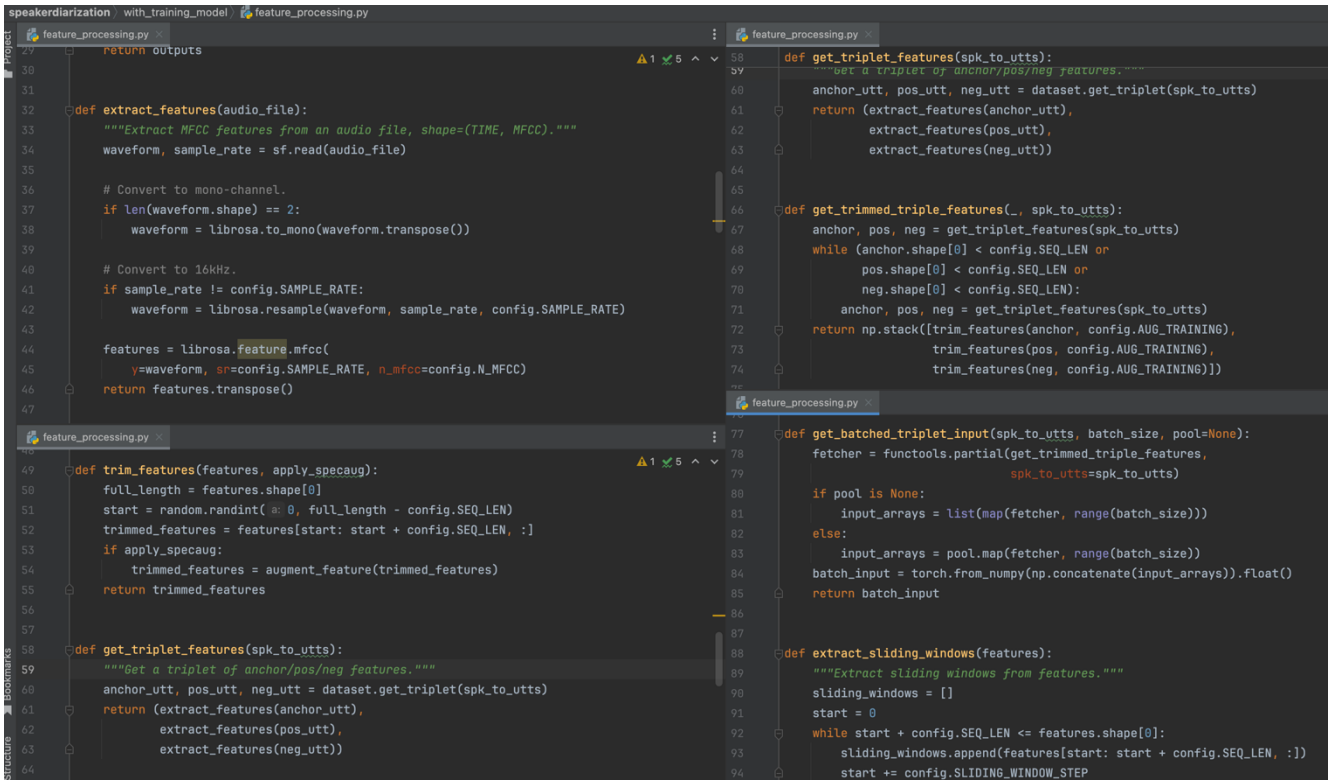


Рисунок 4.12 – Код для роботи з характеристиками аудіо файлів (виконано самостійно)

В цьому прикладі можна виділити опрацювання файлів за допомогою бібліотек functools, numpy, librosa та torch при потребі зміни частоти файлу або аудіо каналу, додатковій аугментації або виділення ознак відрізків.

Ці приклади файлів з функціональною логікою є базовим необхідним наповненням для роботи по створенню та оптимізації моделі розпізнавання промовців.

5 АНАЛІЗ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Підведемо підсумки різних запусків тренування, тестування та перевірки моделей на реальних даних шляхом формування результатів та їх аналізу. Для початку проведемо аналіз ефективності навчання та тестування власноруч створеної моделі з урахуванням використання різних наборів даних.

Наведемо результати роботи програми при використанні тільки одного набору дата сету від LibriSpeech та зміни параметрів конфігурації щодо кількості кроків навчання та шарів нейронної мережі. В таблицях наведені значення функції втрати (див. табл. 5.1).

Таблиця 5.1 – Порівняння результату тренування моделі з використанням одного ресурсу набору даних (виконано самостійно)

Кількість шарів / Кількість кроків	16	32	64	128	256
100	0.0247	0.0194	0.0187	0.0162	0.0151
500	0.0193	0.0173	0.0156	0.0148	0.0134
1000	0.0176	0.0159	0.0137	0.0122	0.0113
2000	0.0162	0.0144	0.0121	0.0105	0.0093
3500	0.0147	0.0128	0.0109	0.0097	0.0079
5000	0.0133	0.0116	0.0101	0.0088	0.0065

При аналізі отриманих результатів видно, що збільшення кількості кроків навчання та шарів LSTM моделі результат покращується через зменшення функції втрат, що в свою чергу описує очікувану високу якість отриманої моделі.

Далі перейдемо до результатів роботи програми при використанні декількох наборів даних та зміни параметрів конфігурації щодо кількості кроків навчання та шарів нейронної мережі (див. табл. 5.2).

Таблиця 5.2 – Порівняння результату тренування моделі з використанням декількох ресурсів набору даних (виконано самостійно)

Кількість шарів / Кількість кроків	16	32	64	128	256
100	0.0467	0.0414	0.0398	0.0365	0.0341
500	0.0432	0.0403	0.0387	0.0332	0.0315
1000	0.0416	0.0362	0.0331	0.0304	0.0293
2000	0.0401	0.0352	0.0329	0.0285	0.0261
3500	0.0387	0.0341	0.0305	0.0259	0.0221
5000	0.0323	0.0276	0.0235	0.0202	0.0179

Тенденція результатів при використані більше одного набору даних є схожою на попереднє дослідження але через більшу різноманітність даних функція втрат є більшою. Наведемо графічне відображення порівняння зміни середніх показників для цих двох досліджень на основі таблиць 5.1 та 5.2 (див. рис. 5.1).

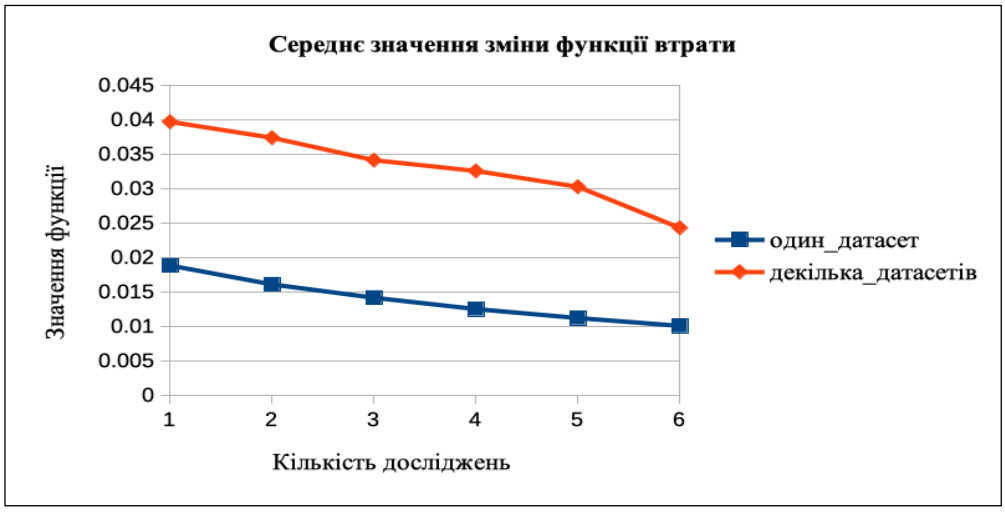


Рисунок 5.1 – Візуалізація результатів на основі таблиць 5.1 та 5.2 (виконано самостійно)

Як бачимо, початкове значення функції для варіанту з більшим набором даних є вищім у порівнянні з варіантом використання тільки одного. У той самий

час збільшення кількості ітерацій навчання буде покращувати ці показники в обох варіантах, але у ситуації з декількома наборами даних модель може стати більш якісною за рахунок ширшого спектру навчальних та перевірочних даних.

Далі перейдемо до результатів перевірки якості моделі при використанні одного та декількох наборів даних та зміни параметрів конфігурації щодо кількості триплетів оцінки даних (див. табл. 5.3).

Таблиця 5.3 – Порівняння результату тестування моделі з використанням одного ресурсу набору даних (виконано самостійно)

Кількість триплетів	Результат ERR	
	З одним набором даних	З декількома наборами даних
50	0.3416	0.5256
100	0.2932	0.4137
500	0.2021	0.3328
1000	0.1815	0.2316
2000	0.1398	0.1412
3000	0.0743	0.0998

Аналізуючи отримані значення видно, що процес навчання може бути оптимізованим за рахунок модифікації параметрів обробки аудіо файлів, формування та трансформації проміжних даних, таких я к триплетів, а також через розширення набору даних через застосування різних тренувальних або перевірочних сетів.

Треба виділити, що ефективність та швидкість роботи з навчанням нейронної мережі також залежить від фізичних особливостей пристрою та оптимізації з мультипроцесорною обробкою команд, що не є критичним для невеликих наборів даних чи спрощеною послідовністю кроків, але може відігравати критичну роль у порівняно комплексних запитах та налаштуваннях.

ВИСНОВКИ

У процесі виконання науково-дослідницької роботи на тему «Дослідження методів та моделей автоматичного розпізнавання учасників аудіо розмови» було проведено дослідження галузі з існуючими моделями та сервісами і розроблена високорівнена схема потенційно можливої системи розпізнавання промовців, проведено експериментування з параметрами машинного навчання щодо покращення цього процесу.

У ході дослідження було визначено що покращення процесу автоматичного розпізнавання учасників аудіо-розмови є актуальною та важливою проблемою в сучасному цифровому світі, оскільки воно може бути застосоване в різних сферах, включаючи телефонну комунікацію, відеоконференції, медіа, а також критичних напрямах, як забезпечення охорони здоров’я та збереження миру.

В цьому дослідженні машинне навчання та нейронні мережі використовуються як основні потужні інструментами для розпізнавання учасників розмови. Використання глибоких нейронних мереж, таких як згорткові нейронні мережі (CNN) та рекурентні нейронні мережі (RNN), дозволяє досягти високої точності в розпізнаванні, а побудова системи навчання на основі прогресивних бібліотек та фреймворків дасть напрям на отримання позитивних результатів.

Під час проведення дослідження було проведено порівняльну характеристику результатів роботи існуючих моделей на базі Reseblyzer та Pyannote audio. Існуючі рішення продемонстрували доволі точні результати та дали поштовх у напрямку побудови власної моделі. У свою чергу практичне дослідження з побудови та тестування власної моделі було спрямовано на використання різних напрямів удосконалення процесів її навчання та перевірки. Це включало використання різних наборів параметрів між ітераціями, розширення набору аудіо даних або зміна самих характеристик вже існуючих файлів, і звичайно, експериментування з можливими налаштуваннями фізичних аспектів машини, таких як використання багатопроцесорного виконання програми або переходом між CPU та GPU процесорами.

Незважаючи на вже існуючі результати у цьому напрямку, є ще багато можливостей для подальшого вдосконалення системи розпізнавання учасників розмови. До них можна віднести використання більш великих та різноманітних даних для навчання та перевірки моделі, розширення функціональності за рахунок оптимізації трансформерів характеристик промовців, для цього можуть бути використанні більш прогресивні ресурси PyTorch чи альтернативний поставник бібліотек, такий як TensorFlow від Google або SageMaker від AWS. Також вибір більш ефективних параметрів для навчання, а саме конфігурація прихованих проміжних шарів або кількості кроків з обробки аудіо проміжків чи вже опрацьованих характеристик.

Реалізація цих аспектів можуть стати наступними цілями щодо подальших досліджень та вдосконалення у галузі аудіо обробки та розпізнавання аудіо сигналів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Бридінський, В. Побудова системи ідентифікації мовців на основі бібліотеки аудіообробки PyAnnote. Information Technology: Computer Science, Software Engineering and Cyber Security. 2022. №2. С. 3-11.

2. Каук В. І. Безпека даних при використанні хмарних обчислень для розробки програмних систем / В. І. Каук, А. О. Трибух // Поліграфічні, мультимедійні та web-технології : матеріали Молодіжної школи-семінару IX Міжнар. наук.-техн. конф., 14-28 травня 2024 р. – Харків : ТОВ «Друкарня Мадрид», 2024. – Т. 2. – С. 86-87.

3. Корнієнко О. Метод відображення мовних сигналів у задачі розпізнавання мовця. Технічні науки та технології. 2017. №3. С. 129-137.

4. Bai Z., Zhang X.-L. Speaker recognition based on deep learning: An overview. Neural Networks. 2021. Vol. 140. P. 65-99.

5. Bredin H., Yin R., Coria J.M., Gelly G, Korshunov P., Lavechin M., Fustes D., Titeux H., Bouaziz W., Gill M. Pyanote.audio: Neural building blocks for speaker diarization. IEEE International Conference on Acoustics, Speech, and Signal Processing. 2019.

6. Comparing state-of-the-art speaker diarization frameworks: Pyannote vs Nemo. URL: https://lajavaness.medium.com/comparing-state-of-the-art-speaker-diarization-frameworks-pyannote-vs-nemo-31a191c6300 (дата звернення: 13.03.2024).

7. D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur. Deep neural network-based speaker embeddings for end-to-end speaker verification, Proceedings of the IEEE Spoken Language Technology work- shop (SLT), 2016.

8. Dawalatabad N., Madikeri S., Sekhar C., Murthy H. Novel Architectures for Unsupervised Information Bottleneck based Speaker Diarization of Meetings. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2020. Vol. 29. P. 14-29.

9. Introduction to Speech Processing. Speaker Diarization. URL:

https://speechprocessingbook.aalto.fi (дата звернення: 20.11.2023).

10. Hernawan S. Speaker Diarization: Its Developments, Applications, And Challenges, Proceedings of The 1 st International Conference on Information Systems For Business Competitiveness, 2023.

11. Khovrat, A., Kobziev, V. Using Recurrent and Convulation Neural Networks to Indentify the Fake Audio Messages, IEEE 7th International Conference on Methods and Systems of Navigation and Motion Control, MSNMC 2023 - Proceedings, 2023, P. 174–177.

12. Kizitskyi,M., Turuta, O., Turuta, O. Improving Speaker Verification Model for Low-Resources Languages, CEUR Workshop Proceedings, 2023, 3403, P. 99–113.

13. Li C., Ma X. , Jiang B., Li X., Zhang X., Liu X., Cao Y., Kannan A., Zhu Z. Deep Speaker: an End-to-End Neural Speaker Embedding System. Computation and Language. 2017.

14. Lim D. K., Rashid N.U., Oliva J.B., Ibrahim J.G. Deeply-Learned Generalized Linear Models with Missing Data. Journal of Computational and Graphical Statistics, 2022.

15. Lu H. LSTM-Based stock price predictions. Frontiers in Computing and Intelligent Systems. 2023. Vol. 4. P. 68-71.

16. Lyashenko, V., Rabotiahov, A., Kobylin, O., Kolesnykov, D. Analysis of Human Speech as a Protection Tool in Infocommunication Systems, International Scientific-Practical Conference on Problems of Infocommunications Science and Technology, PIC S and T 2018 - Proceedings, 2019, P. 79–83.

17. Ming C. Experiments in speaker diarization using speaker vectors. Degre Project in computer science and engineering. Sweden, 2021.

18. P. Swathi. Advanced machine learning algorithms based image recognition. International Journal of Innovative Research in Computer and Communication Engineering. 2019.

19. Shirshendu R., Pratyay M. Air quality index forecasting using hybrid neural network model with LSTM on AQI. Proceedings on Engineering Sciences 2(4). 2020.

20. Smelyakov, K., Chupryna, A., Darahan, D., Midina, S. Effectiveness of modern text recognition solutions and tools for common data sources, CEUR Workshop Proceedings, 2021, 2870, P. 154–165

21. Yakovlev S., Khovrat, A., Kobziev, V. Using Parallelized Neural Networks to Detect Falsified Audio Information in Socially Oriented Systems, CEUR Workshop Proceedings, 2023, 3624, P. 220–238

ДОДАТОК А

Перелік джерел посилання за науковими напрями керівника та науковців кафедри програмної інженерії

2. Каук В. І. Безпека даних при використанні хмарних обчислень для розробки програмних систем / В. І. Каук, А. О. Трибух // Поліграфічні, мультимедійні та web-технології : матеріали Молодіжної школи-семінару IX Міжнар. наук.-техн. конф., 14-28 травня 2024 р. – Харків : ТОВ «Друкарня Мадрид», 2024. – Т. 2. – С. 86-87.

11. Khovrat, A., Kobziev, V. Using Recurrent and Convulation Neural Networks to Indentify the Fake Audio Messages, IEEE 7th International Conference on Methods and Systems of Navigation and Motion Control, MSNMC 2023 - Proceedings, 2023, P. 174–177.

12. Kizitskyi,M., Turuta, O., Turuta, O. Improving Speaker Verification Model for Low-Resources Languages, CEUR Workshop Proceedings, 2023, 3403, P. 99–113.

16. Lyashenko, V., Rabotiahov, A., Kobylin, O., Kolesnykov, D. Analysis of Human Speech as a Protection Tool in Infocommunication Systems, International Scientific-Practical Conference on Problems of Infocommunications Science and Technology, PIC S and T 2018 - Proceedings, 2019, P. 79–83.

20. Smelyakov, K., Chupryna, A., Darahan, D., Midina, S. Effectiveness of modern text recognition solutions and tools for common data sources, CEUR Workshop Proceedings, 2021, 2870, P. 154–165

21. Yakovlev S., Khovrat, A., Kobziev, V. Using Parallelized Neural Networks to Detect Falsified Audio Information in Socially Oriented Systems, CEUR Workshop Proceedings, 2023, 3624, P. 220–238

ДОДАТОК Б

Апробація результатів роботи

УДК: 004.89 DOI: https://doi.org/10.30837/IYF.IIS.2024.428

ДОСЛІДЖЕННЯ МЕТОДІВ ТА МОДЕЛЕЙ АВТОМАТИЧНОГО РОЗПІЗНАВАННЯ УЧАСНИКІВ АУДІО РОЗМОВИ

Андрєєв І. Г.
Науковий керівник – к.т.н., доц. каф. ПІ Каук В. І.
Харківський національний університет радіоелектроніки, каф. ПІ,
м. Харків, Україна
e-mail: ivan.andrieiev@nure.ua

The object of the study is the process of automatic speech recognition (ASR) and one of the branch of it – speaker recognition or diarization (SD).

The purpose of the work is to study the stages of the process of speech and speakers recognition, the analysis of methods and models of machine learning and neural networks, as well as modern frameworks for training speech recognition models and diarization. Based on the received results of the study it would be selected the existing models, systems or products that would play a role of starting point in the creation of a custom ASR and SD models for enhancing those processes and bringing more values and benefits in different areas of human being.

У сучасному цифровому світі, де аудіо розмови і мовлення є невід'ємною частиною нашого повсякденного життя, проблема розпізнавання мовлення учасників розмови стає все більш актуальною і важливою. Ця проблема має великий потенціал застосування в різних сферах, включаючи телефонну комунікацію, відеоконференції, медіа і багато інших галузей.

Машинне навчання та нейронні мережі є ключовими компонентами сучасних процесів розпізнавання мовлення та визначення учасників розмови. Машинне навчання дозволяє моделі навчатися на основі великої кількості даних та робити прогнози, а нейронні мережі, зокрема глибокі нейронні мережі, є потужними інструментами для обробки аудіо даних та розпізнавання мовлення.

Покращення процесу розпізнавання учасників розмови та оптимізація використання цих технологій можуть дати позитивний вплив у таких напрямах:

– аутентифікація та безпека: питання безпеки стає особливо актуальним у контексті конфіденційності та захисту від несанкціонованого доступу;

– навчання: транскрипція запитань і відповідей студента для аналізу відповідей, наданих професором або студентами;

– здоров'я: окремі коментарі пацієнтів і лікарів як для особистих зустрічей, так і для телефонних консультацій;

Рисунок Б.1 – XXVIII Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті», Харків, 2024. Перша сторінка

– підтримка продажів: відстеження того, хто що сказав на зустрічі з продавцями, і навчання продавця, що говорити та коли мовчати;

– аналіз доповідача: відстежуйте поточні та попередні коментарі певного доповідача під час зустрічей або відстежуйте час розмови під час телефонної розмови, тощо.

Метою цього дослідження є аналіз методів та моделей автоматичного розпізнавання учасників аудіо розмови з використанням машинного навчання і нейронних мереж та встановленню шляхів покращення окремих етапів або всього процесу розпізнавання мовлення загалом. До основних підходів визначення мовлення виділені:

– підхід під наглядом (класифікація): модель навчена розпізнавати обмежену кількість промовців (тобто класів). Цей підхід менш гнучкий, але може бути дуже ефективним, особливо для великої кількості промовців (>5);

– підхід без нагляду (кластеризація): модель групує аудіо сегменти відповідно до промовця на основі виділених аудіо характеристик.

Об'єктом дослідження є процеси автоматичного розпізнавання мовлення (ASR) та розпізнавання учасників розмови (SD), а також існуючі системи та сервіси, що реалізують ці процеси. Найбільш популярні сервіси з розпізнавання мови та визначення учасників розмови базують навчання своїх моделей на основі згорткових (CNN) і рекурентних типів нейронних мережах (RNN).

Отримані результати цього дослідження стануть потенційно важливою складовою для розвитку технологій обробки аудіо даних та розпізнавання мовлення. Аналіз моделей та процесів, з одного боку, є важливою складовою на шляху до повного розуміння цієї галузі та її перспектив, а з іншого, є реальним шансом використати знання з області машинного навчання та нейронних мереж для вирішення реальних завдань сучасного світу аудіо відносин.

Список використаних джерел

1. Бридінський, В. Побудова системи ідентифікації мовців на основі бібліотеки аудіообробки PyAnnote. Information Technology: Computer Science, Software Engineering and Cyber Security. 2022. №2. С. 3-11.

2. Корнієнко О. Метод відображення мовних сигналів у задачі розпізнавання мовця. Технічні науки та технології. 2017. №3. С. 129-137.

3. Bai Z., Zhang X.-L. Speaker recognition based on deep learning: An overview. Neural Networks. 2021. Vol. 140. P. 65-99.

4. Introduction to Speech Processing. Speaker Diarization. URL: https://speechprocessingbook.aalto.fi (дата звернення: 20.11.2023).

Рисунок Б.2 – XXVIII Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті», Харків, 2024. Друга сторінка

ДОДАТОК В

Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ

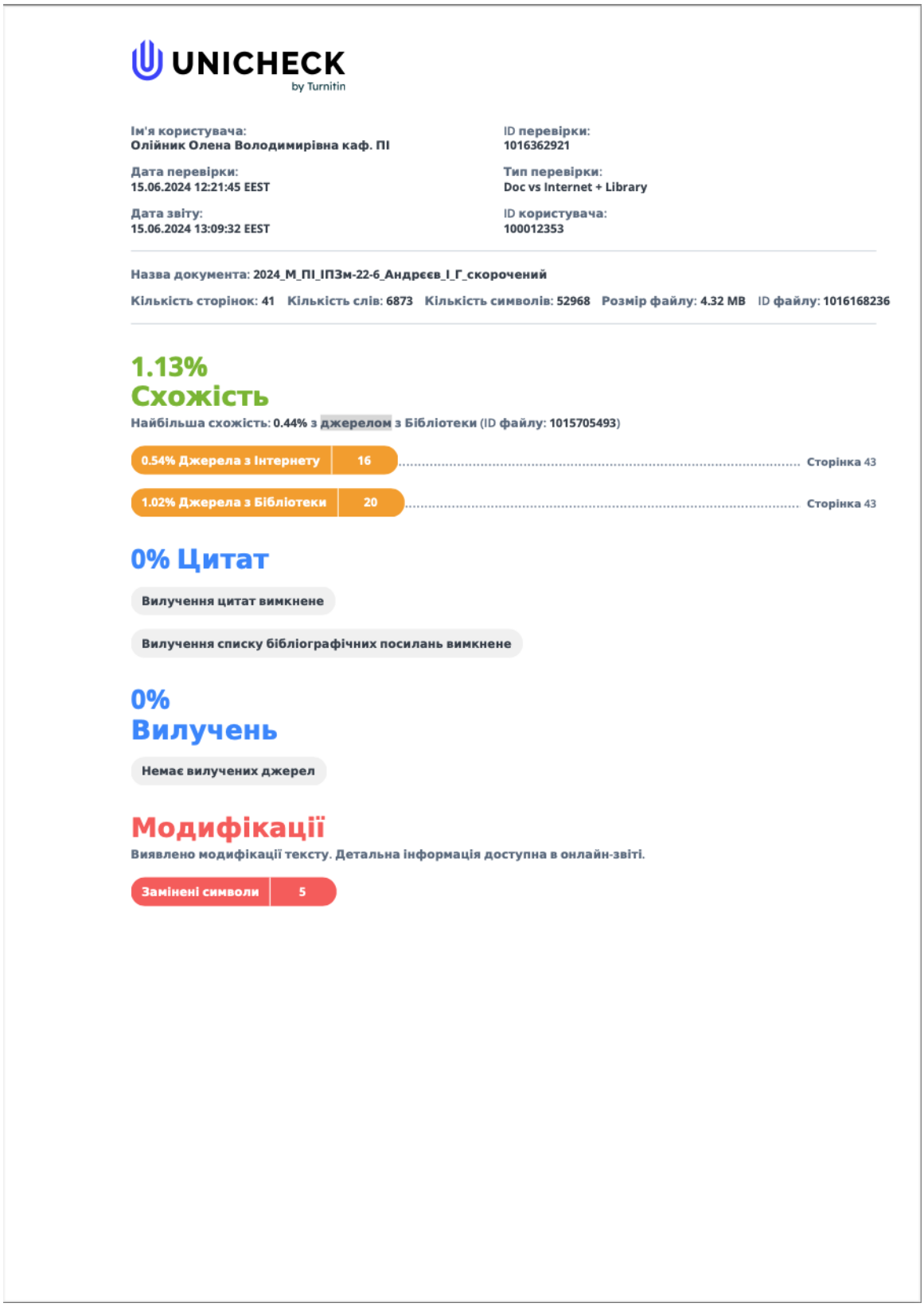


Рисунок В.1 – Звіт результатів перевірки на унікальність тексту в базі ХНУРЕ

ДОДАТОК Г

Слайди презентації

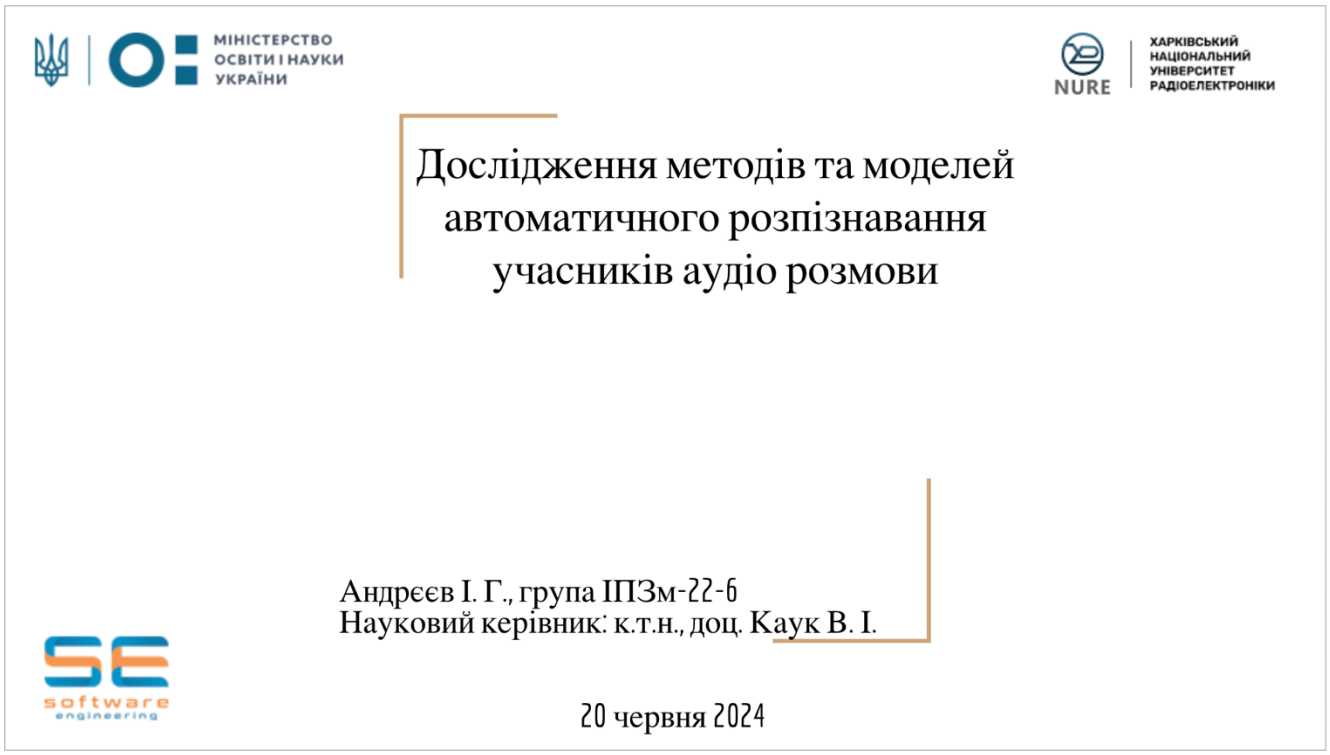


Рисунок Г.1 – Слайд №1 презентації

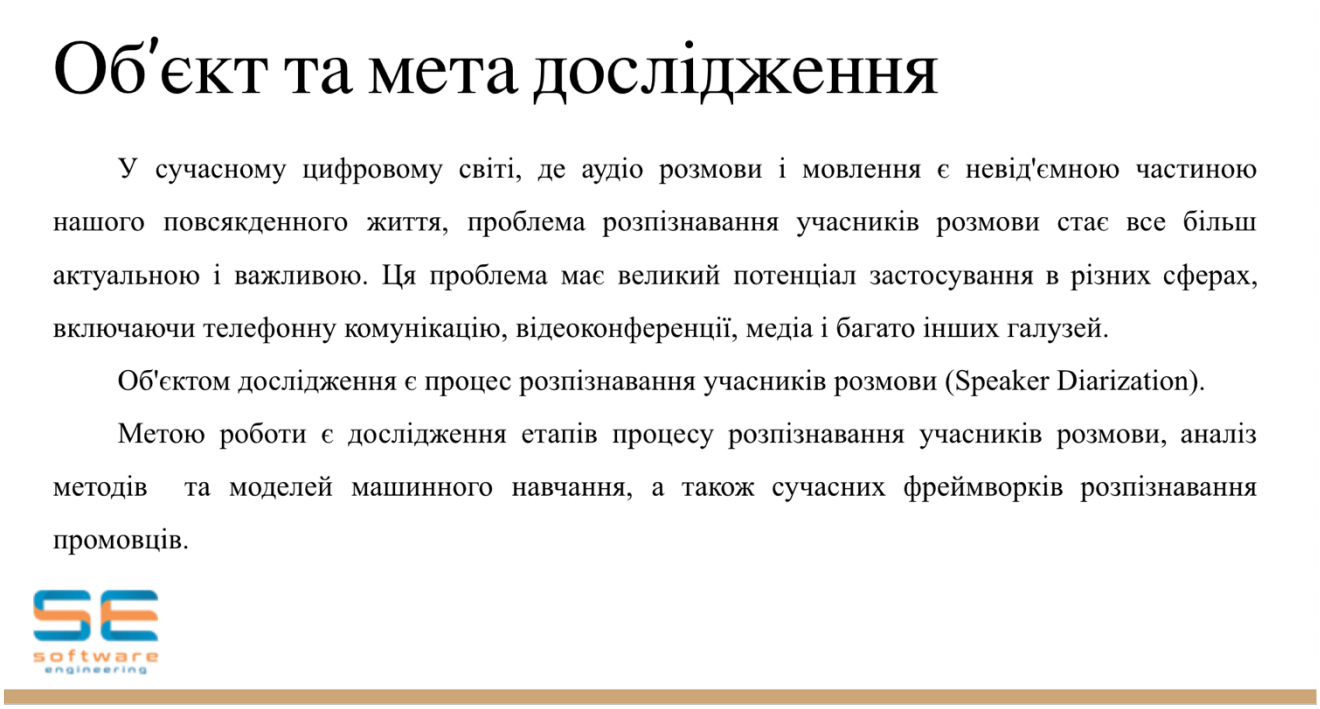


Рисунок Г.2 – Слайд №2 презентації

Огляд процесу розпізнавання мовців

У контексті розпізнавання мовця з підходом без нагляду, наразі є найбільш поширеним підходом, послідовність виконання включає кілька завдань для виявлення голосової активності та сегментації аудіо на мовні та немовні сегменти, а потім групування їх відповідно до промовця

Послідовність кроків розпізнавання промовців з використанням підходу без нагляду

Рисунок Г.3 – Слайд №3 презентації

Архітектура підходу до навчання

Сучасні популярні сервіси розпізнавання промовців базують навчання своїх моделей на основі RNN типі нейронної мережі. RNN використовують повторювані з'єднання, які дозволяють інформації зберігатися та перетікати від одного кроку до наступного. Це дозволяє їм фіксувати тимчасові залежності та послідовні шаблони в даних. RNN мають приховані стани, які зберігають пам'ять про минулі вхідні дані, що робить їх здатними навчатися з історичного контексту.

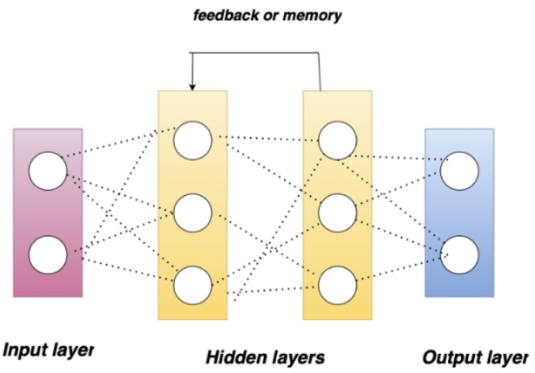


Рисунок Г.4 – Слайд №4 презентації

Огляд існуючих сервісів

Аспект	Amazon Transcribe	Speech Service MS Azure	Speech-to-Text Google Cloud	AssemblyAI	IBM Watson
Модифікація	Словник, пунктуація, ідентифікація мови	Використання власної моделі	Зміна голосу, адаптація мови	Аналіз настроїв, виявлення теми	Адаптація до специфічної лексики
Сфери використання	Контакт-центри, створення медіаконтенту	Різні напрями у підприємництві	Системи навігації, онлайн зустрічі	Віртуальні асистенти, чат боти	Бізнес-аналітика, маркетинг
Особливі властивості	Генеративне підсумовування дзвінків, рівень впевненості слів	Власні нейронні моделі голосу, переклад у реальному часі	Маркування аудіо даних	Використання аудіо інтелекту	Виявлення ключових слів, визначення емоцій
Легкість використання	Мінімальні зміни для існуючих клієнтів	Простота використання для розробників	Спрощена обробка та аналіз даних	Проста інтеграція з API	Зручний інтерфейс та інтеграція

Рисунок Г.5 – Слайд №5 презентації

Схема потенційної системи

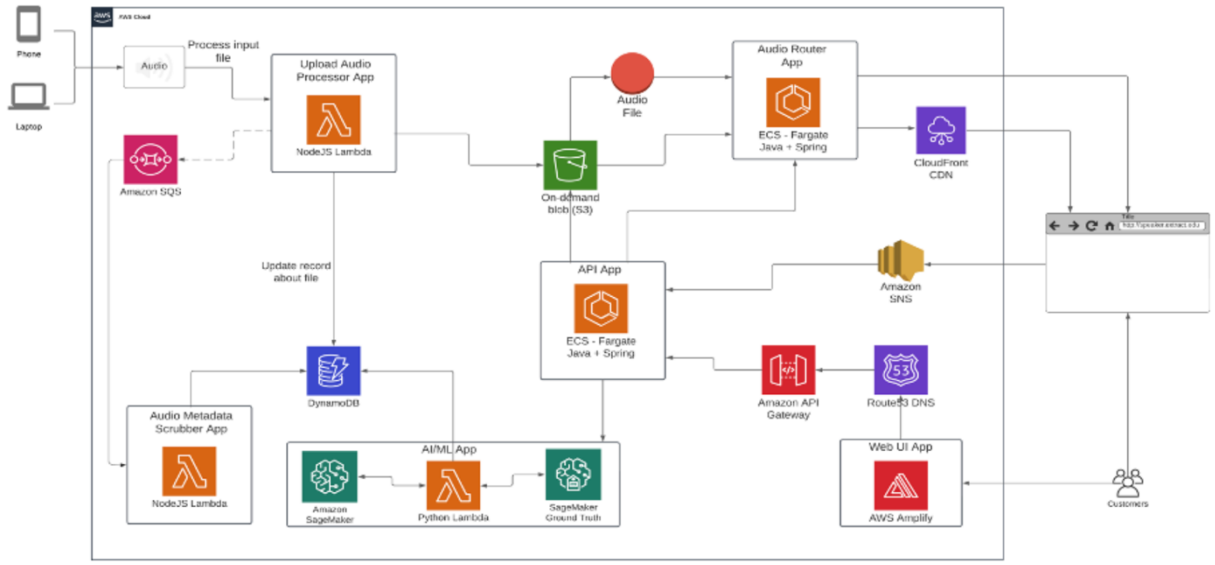


Рисунок Г.6 – Слайд №6 презентації

Методологія та план дослідження

План дослідження має наступні фази:

1. визначення цілі - визначення масштабу та контексту проведення розпізнавання мовця
2. збір та анотація даних - можливість використання анотовані набори даних
3. підготовка даних для навчання - визначення ознак звукових характеристик в числовому вигляді, щоб передати їх на вхід нейронній мережі
4. навчання моделі - використання фреймворків та налаштування параметрів
5. оцінка моделі - аналіз показника частоти помилок розпізнавання промовця, ERR, Loss Function
6. експериментування та ітерація
7. тестування в реальних умовах
8. формування висновків та оформлення результатів

Рисунок Г.7 – Слайд №7 презентації

Набори даних для дослідження

Характеристика	LibriSpeech	VoxCeleb	CN-Celeb
Розмір (години)	1000	VoxCeleb1: 352	1300
Джерело	Аудіокниги	Інтерв'ю, новини, ток-шоу, YouTube	Телевізійні шоу, інтерв'ю, новини, документальні фільми, фільми
Якість звуку	16 кГц	16 кГц	16 кГц
Анотації	Точні текстові транскрипції	Метадані про мовців	Метадані про мовців, часові мітки для мовленнєвих сегментів
Різноманітність	Професійні диктори	Широкий спектр акцентів, стилів мовлення, умов запису	Широкий спектр умов запису, емоційних станів, акцентів, шумів

Рисунок Г.8 – Слайд №8 презентації

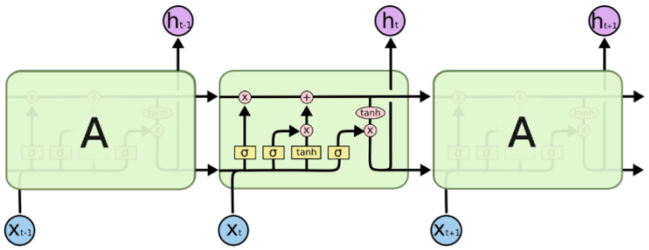
Методи аугментація аудіо записів

- додавання шуму;
- зміна швидкості відтворення;
- зміна висоти тону;
- реверберація;
- часова зміна;
- комбінація методів.

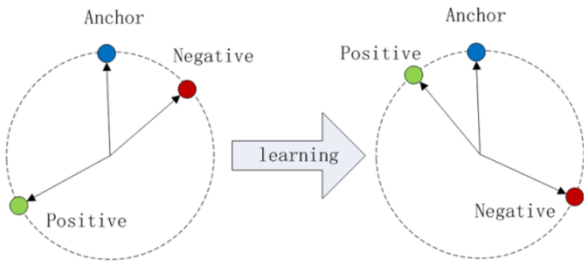
```python
import numpy as np
import librosa
import soundfile as sf
import os

# Завантаження аудіофайлу
directory_path = os.path.join(os.path.expanduser("~"), 'Documents/Nure/Mag_robota/DataForProcessing')
file_path = directory_path + '/Conference.wav'
y, sr = librosa.load(file_path, sr=None)

# Додавання білого шуму
noise = np.random.randn(len(y))
y_noisy = y + 0.005 * noise

# Зміна швидкості відтворення
y_fast = librosa.effects.time_stretch(y, rate=1.25)
y_slow = librosa.effects.time_stretch(y, rate=0.75)

# Зміна висоти тону
y_pitch_up = librosa.effects.pitch_shift(y, sr=sr, n_steps=4)
y_pitch_down = librosa.effects.pitch_shift(y, sr=sr, n_steps=-4)

# Збереження результатів
sf.write(directory_path + '/Conference_noisy.wav', y_noisy, sr)
sf.write(directory_path + '/Conference_fast.wav', y_fast, sr)
sf.write(directory_path + '/Conference_slow.wav', y_slow, sr)
sf.write(directory_path + '/Conference_pitch_up.wav', y_pitch_up, sr)
sf.write(directory_path + '/Conference_pitch_down.wav', y_pitch_down, sr)
```

Рисунок Г.9 – Слайд №9 презентації

Машинне навчання у програмній реалізації



Повторюваний LSTM модуль



Триплетна втрата з косинусною подібністю

Рисунок Г.10 – Слайд №10 презентації

Виконання процесу розпізнавання мовця

Приклад виконання кроків з розпізнавання промовця з використанням Resemblyzer

```python
encoder = VoiceEncoder()
_, cont_embeds, wav_splits = encoder.embed_utterance(wav, return_partials=True, rate=16)

print(f'cont_embeds = ${cont_embeds}')

clusterer = SpectralClusterer(
    min_clusters=3,
    max_clusters=6)

labels = clusterer.predict(cont_embeds)
print(f'labels = ${labels}')

labelling = create_labelling(labels, wav_splits)
labeled_speakers = build_speakers(labelling)

print(f'labelling = ${labeled_speakers}')

speaker_embeds = [encoder.embed_utterance(speaker_wav) for speaker_wav in speaker_wavs]
similarity_dict = {name: cont_embeds @ speaker_embed for name, speaker_embed in
                   zip(speaker_names, speaker_embeds)}

## Run the interactive demo
interactive_diarization(similarity_dict, wav, wav_splits)
```

Рисунок Г.11 – Слайд №11 презентації

Результат розпізнавання мовця

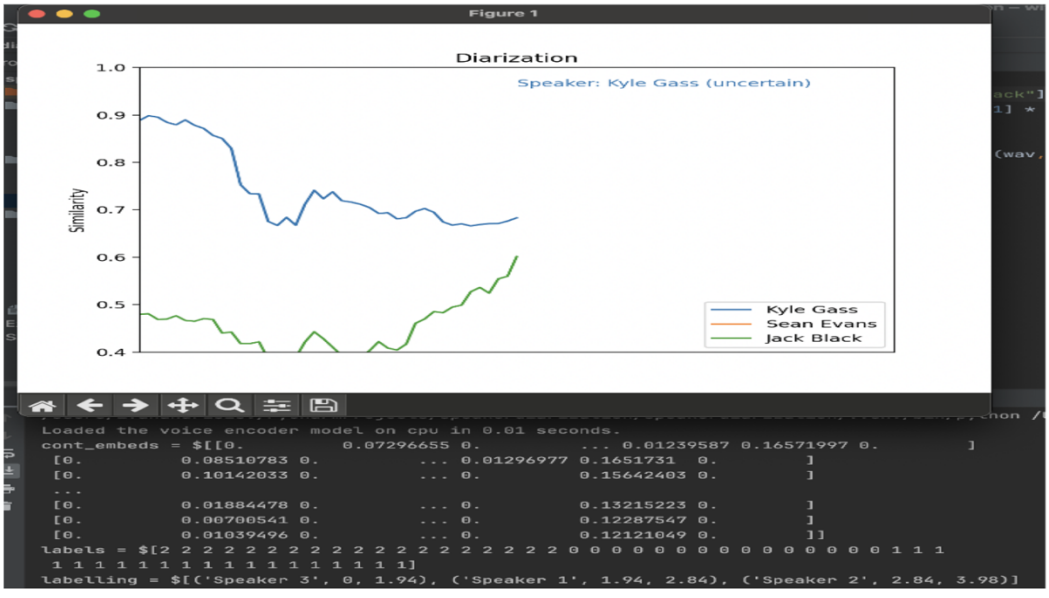
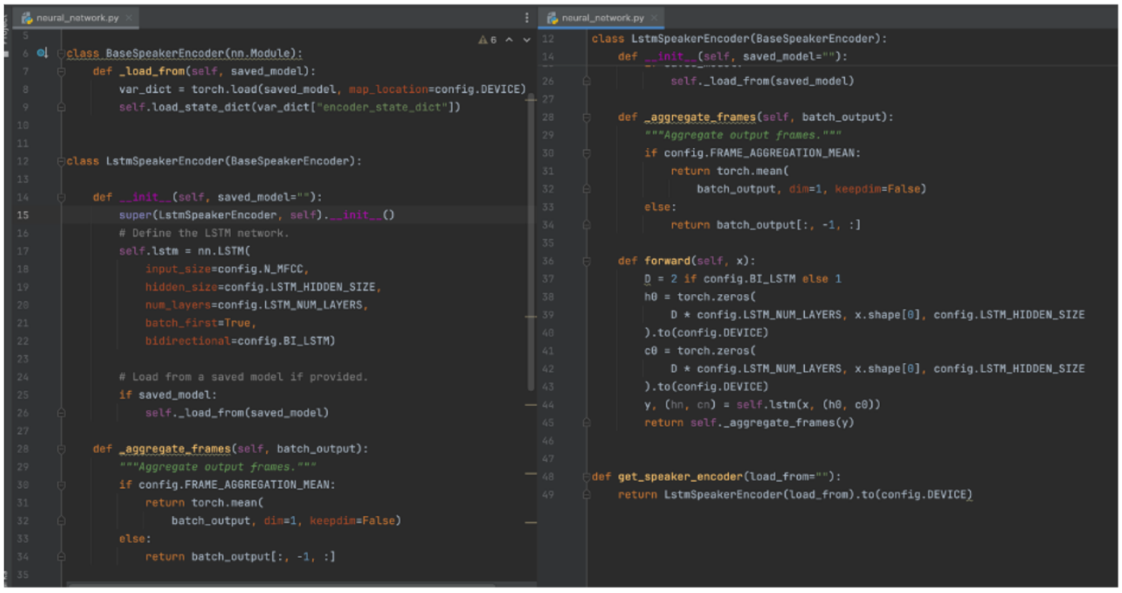


Рисунок Г.12 – Слайд №12 презентації

Приклад коду побудови власної моделі



```python
class BaseSpeakerEncoder(nn.Module):
    def _load_from(self, saved_model):
        var_dict = torch.load(saved_model, map_location=config.DEVICE)
        self.load_state_dict(var_dict["encoder_state_dict"])


class LstmSpeakerEncoder(BaseSpeakerEncoder):

    def __init__(self, saved_model=""):
        super(LstmSpeakerEncoder, self).__init__()
        # Define the LSTM network.
        self.lstm = nn.LSTM(
            input_size=config.N_MFCC,
            hidden_size=config.LSTM_HIDDEN_SIZE,
            num_layers=config.LSTM_NUM_LAYERS,
            batch_first=True,
            bidirectional=config.BI_LSTM)

        # Load from a saved model if provided.
        if saved_model:
            self._load_from(saved_model)

    def _aggregate_frames(self, batch_output):
        """Aggregate output frames."""
        if config.FRAME_AGGREGATION_MEAN:
            return torch.mean(
                batch_output, dim=1, keepdim=False)
        else:
            return batch_output[:, -1, :]
```

```python
class LstmSpeakerEncoder(BaseSpeakerEncoder):
    def __init__(self, saved_model=""):
            self._load_from(saved_model)

    def _aggregate_frames(self, batch_output):
        """Aggregate output frames."""
        if config.FRAME_AGGREGATION_MEAN:
            return torch.mean(
                batch_output, dim=1, keepdim=False)
        else:
            return batch_output[:, -1, :]

    def forward(self, x):
        D = 2 if config.BI_LSTM else 1
        h0 = torch.zeros(
            D * config.LSTM_NUM_LAYERS, x.shape[0], config.LSTM_HIDDEN_SIZE
        ).to(config.DEVICE)
        c0 = torch.zeros(
            D * config.LSTM_NUM_LAYERS, x.shape[0], config.LSTM_HIDDEN_SIZE
        ).to(config.DEVICE)
        y, (hn, cn) = self.lstm(x, (h0, c0))
        return self._aggregate_frames(y)


def get_speaker_encoder(load_from=""):
    return LstmSpeakerEncoder(load_from).to(config.DEVICE)
```

Рисунок Г.13 – Слайд №13 презентації

Результат навчання власної моделі

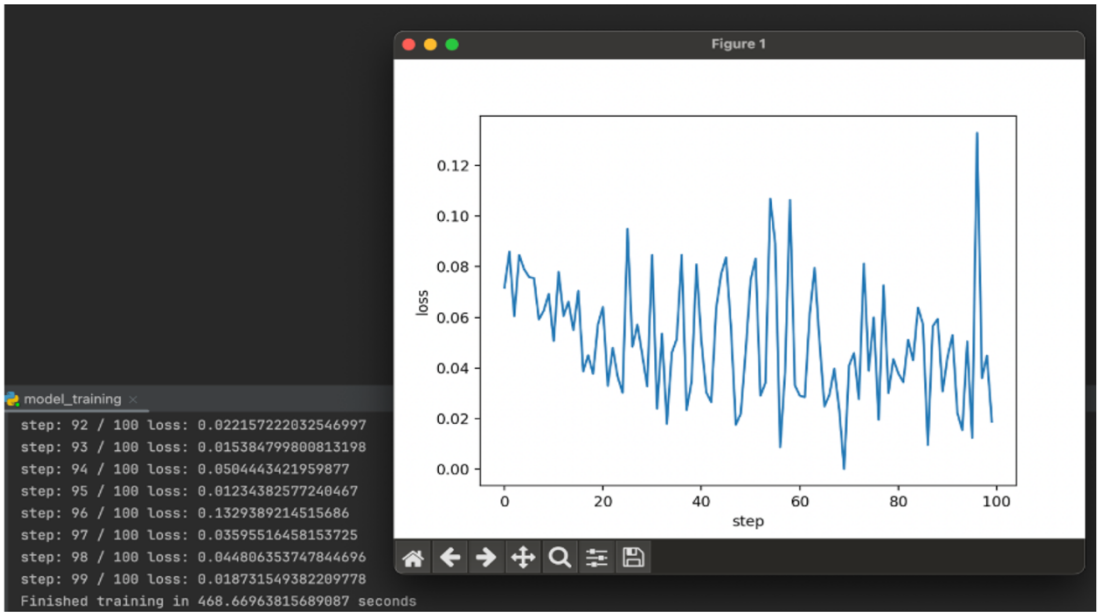


Рисунок Г.14 – Слайд №14 презентації

Результат тестування власної моделі

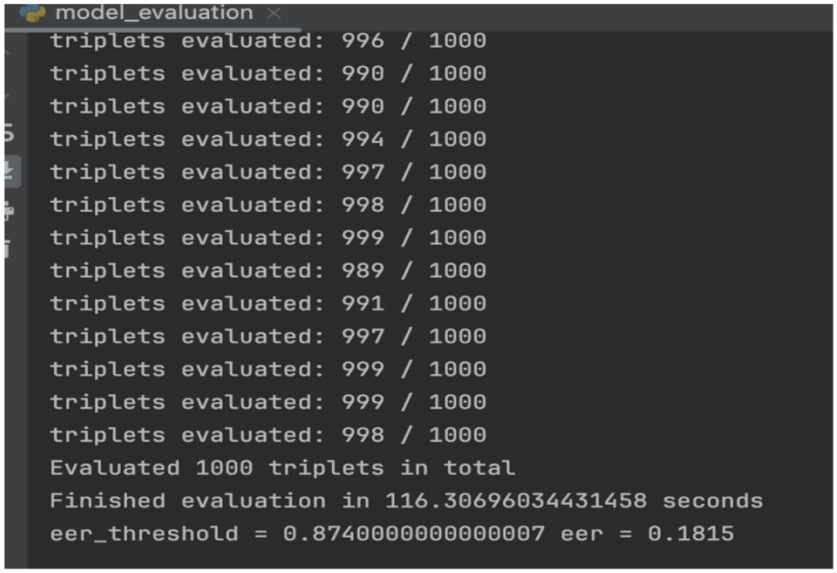


Рисунок Г.15 – Слайд №15 презентації

Результати експерименту, частина А

Кількість шарів / Кількість кроків	16	32	64	128	256
100	0.0247	0.0194	0.0187	0.0162	0.0151
500	0.0193	0.0173	0.0156	0.0148	0.0134
1000	0.0176	0.0159	0.0137	0.0122	0.0113
2000	0.0162	0.0144	0.0121	0.0105	0.0093
3500	0.0147	0.0128	0.0109	0.0097	0.0079
5000	0.0133	0.0116	0.0101	0.0088	0.0065

Рисунок Г.16 – Слайд №16 презентації

Результати експерименту, частина Б

Кількість шарів / Кількість кроків	16	32	64	128	256
100	0.0467	0.0414	0.0398	0.0365	0.0341
500	0.0432	0.0403	0.0387	0.0332	0.0315
1000	0.0416	0.0362	0.0331	0.0304	0.0293
2000	0.0401	0.0352	0.0329	0.0285	0.0261
3500	0.0387	0.0341	0.0305	0.0259	0.0221
5000	0.0323	0.0276	0.0235	0.0202	0.0179

Рисунок Г.17 – Слайд №17 презентації

Результати експерименту, частина В

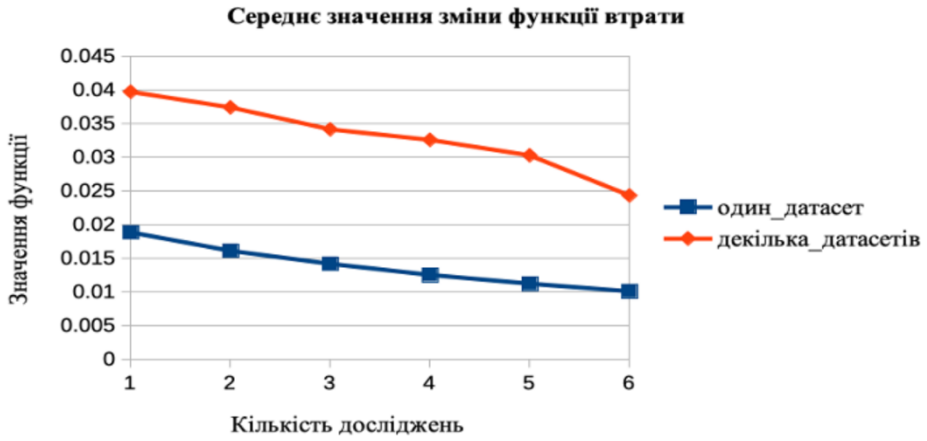


Рисунок Г.18 – Слайд №18 презентації

Результати експерименту, частина Г

Кількість триплетів	Результат ERR	
	З одним набором даних	З декількома наборами даних
50	0.3416	0.5256
100	0.2932	0.4137
500	0.2021	0.3328
1000	0.1815	0.2316
2000	0.1398	0.1412
3000	0.0743	0.0998

Рисунок Г.19 – Слайд №19 презентації

Аналіз отриманих результатів

Під час проведення дослідження було проведено порівняльну характеристику результатів роботи існуючих моделей на базі Reseblyzer та Pyannote audio. Існуючі рішення продемонстрували доволі точні результати та дали поштовх у напрямку побудови власної моделі.

Аналізуючи отримані значення власного навчання видно, що процес навчання може бути оптимізованим за рахунок модифікації параметрів обробки аудіо файлів, формування та трансформації проміжних даних, таких я к триплетів, а також через розширення набору даних через застосування різних тренувальних або перевірочних сетів.

Треба виділити, що ефективність та швидкість роботи з навчанням нейронної мережі також залежить від фізичних особливостей пристрою та оптимізації з мультипроцесорною обробкою команд, що не є критичним для невеликих наборів даних чи спрощеною послідовністю кроків, але може відігравати критичну роль у порівняно комплексних запитах та налаштуваннях.

Рисунок Г.20 – Слайд №20 презентації

Підсумки

Незважаючи на вже існуючі результати у цьому напрямку, є ще багато можливостей для подальшого вдосконалення системи розпізнавання учасників розмови. До них можна віднести використання більш великих та різноманітних даних для навчання та перевірки моделі, розширення функціональності за рахунок оптимізації трансформерів характеристик промовців, для цього можуть бути використанні більш прогресивні ресурси PyTorch чи альтернативний поставник бібліотек, такий як TensorFlow від Google або SageMaker від AWS. Також вибір більш ефективних параметрів для навчання, а саме конфігурація прихованих проміжних шарів або кількості кроків з обробки аудіо проміжків чи вже опрацьованих характеристик або експериментування з можливими налаштуваннями фізичних аспектів машини, таких як використання багатопроцесорного виконання програми або переходом між CPU та GPU процесорами.

Рисунок Г.21 – Слайд №21 презентації

ДОДАТОК Д

Експертний висновок результатів перевірки кваліфікаційної роботи на відповідність оформлення вимогам ДСТУ 3008: 2015

1

Експертний висновок результатів перевірки кваліфікаційної роботи

студент (посада) | програмної інженерії (кафедра) | ІПЗм-22-6 (група)

Андрєєв Іван Георгійович

(прізвище, ім'я, по батькові)

Зауваження

Пункт ДСТУ 3008-2015	Зміст пункту	Сторінка кваліфікаційної роботи
1	2	3
	7.1 Загальні положення	
	7.3 Нумерація сторінок звіту	
	7.4 Нумерація розділів, підрозділів, пунктів, підпунктів	
	7.5 Рисунки	
	7.6 Таблиці	
	7.7 Переліки	
	7.8 Примітки	
	7.9 Виноски	
	7.10 Формули та рівняння	
	7.11 Посилання	
	7.13 Список авторів	
	7.14 Скорочення та умовні позначки	
	7.15 Додатки	

зауважень немає

Експерт _____ (підпис) Олена ОЛІЙНИК (прізвище, ініціали)

16.06.2024

Рисунок Д.1 – Експертний висновок результатів перевірки кваліфікаційної роботи