МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Київський національний університет імені Тараса Шевченка
Навчально-науковий інститут філології
Кафедра української мови та прикладної лінгвістики

АВТОМАТИЧНА ДІАРИЗАЦІЯ МОВЦІВ НА МАТЕРІАЛАХ АУДІОЗАПИСІВ-ПЕРЕХОПЛЕНЬ РОСІЙСЬКИХ ЗАГАРБНИКІВ

Кваліфікаційна робота магістра
за спеціальністю 035 «Філологія»,
спеціалізацією 035.10 «Прикладна лінгвістика»,
галузі знань 03 «гуманітарні науки»
ОП "Прикладна лінгвістика (редакторсько-перекладацька та експертна діяльність)"
Валерії ШЕМЧУК

Наукові керівники:
д.філол.н., проф. Наталія ДАРЧУК,
Валентина РОБЕЙКО

Рецензент:
к.техн.н. Микола САЖОК

«Допущено до захисту»
Протокол № 10 засідання кафедри
української мови та прикладної лінгвістики
ННІФ від 04.06.2023
Завідувач кафедри_____**Сергій Різник**

Київ – 2023

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ГУР – Головне Управління Розвідки Міністерства оборони України

WAV – Waveform Audio File Format, формат аудіофайлу Waveform

RTTM – Rich Transcription Time Marked File Format, розширений формат файлу з часовими позначками транскрипцій

FRR – False Rejection Rate, коефіцієнт помилкових відмов

FAR – False Acceptance Rate, коефіцієнт помилкового прийняття

EER – Equal Error Rate – однаковий коефіцієнт помилок

MFCC – Mel-frequency cepstral coefficients, мел-частотні кепстральні коефіцієнти

VAD – Voice Activity Detection, виявлення голосової активності

DER – Diarization error rate, коефіцієнт помилок діаризації

ВСТУП

У зв'язку з повномасштабним вторгненням Росії на територію України 24 лютого 2022 року в нашому інформаційному просторі з'явилось безліч аудіозаписів (наприклад, перехоплення від ГУР МО України [1]) з участю російських військовослужбовців. Ці аудіозаписи мають важливе значення для розслідування воєнних злочинів, вчинених проти українців та української державності, тому важливо їх ретельно аналізувати та досліджувати. Записи можуть містити докази про військових злочинців, а також злочини, які були скоєні під час повномасштабного вторгнення. Ідентифікація мовців у цих записах може розкрити деталі щодо *планування, організації та виконання злочинів*, а також *причетності окремих осіб чи груп до цих подій.* Це може стати цінною інформацією для правосуддя, міжнародних судів та розслідувань. Розголос, який супроводжує розкриття аудіозаписів та їх використання як доказів, безумовно сприятиме створенню міжнародного тиску на злочинців і тих, хто відповідає за окупацію. Це може привести до засудження та санкцій проти відповідальних осіб і держав, що підтримують агресію. Дослідження про мовців на аудіозаписах може викликати більшу увагу міжнародної спільноти до російсько-української війни та потреби в допомозі Україні.

Технології автоматичного розпізнавання мовлення та мовців можуть відіграти важливу роль у розслідуванні злочинів, скоєних окупантами або зрадниками, які погодились на співпрацю з країною-агресором.

Ось декілька прикладів, як можна використати ці технології:

1. Ідентифікація мовців. Звукові записи, такі як телефонні розмови, аудіозаписи або відеозаписи, можуть містити голосові сліди окупантів. Технологія розпізнавання мовця може допомогти в ідентифікації осіб за їх

унікальними голосовими характеристиками. Це може бути важливим доказом при розслідуванні злочинів [53].

2. Верифікація записів. Якщо є підозра, що голосові записи, надані окупантами, можуть бути сфальсифікованими або зміненими, технологія розпізнавання мовця може використовуватись для перевірки автентичності цих записів. Вона може допомогти визначити, чи належать записи певній особі, або виявити будь-які зміни в голосових характеристиках, що можуть свідчити про фальсифікацію [25]. Для виявлення правдивих біометричних даних застосовують антиспуфінг.

Антиспуфінг (англ. “anti-spoofing”) — це процес виявлення та запобігання спробам обходу або підробки біометричних систем аутентифікації. Антиспуфінг-технології використовуються для виявлення живих біометричних ознак, що дозволяє відрізняти їх від підроблених чи статичних зразків. У контексті розпізнавання голосу антиспуфінг використовується для розрізнення живого мовлення від підроблених записів голосу, таких як відтворені аудіозаписи або синтезований голос. Техніки антиспуфінгу можуть включати аналіз фізичних характеристик голосу, використання додаткових сенсорів (наприклад, камери для виявлення руху губ або обличчя), а також аналіз активності та інших динамічних параметрів мовлення [74].

3. Аналіз змісту розмов. Крім голосових характеристик, технології розпізнавання мовлення та мовців також можуть використовуватись для аналізу змісту розмов (розпізнавання мовлення та автоматична генерація стенограм розмов, виявлення емоцій, аналіз стильових особливостей мовлення тощо). Ці аспекти можуть бути використані для отримання додаткової інформації про особу, її наміри або стан під час злочину [40].

4. Системи моніторингу. Технології розпізнавання мовлення та мовців можуть бути використані для розробки систем моніторингу, які автоматично аналізують аудіо- та відеозаписи, щоб виявляти підозрілу

активність або специфічні ключові слова, пов'язані зі злочинами. Наприклад, система може виявити загрозливу мову або ключові фрази, які вказують на злочинну діяльність, що допомагає вчасно реагувати на потенційні загрози [47].

5. Виявлення специфічних характеристик мовлення. Технології розпізнавання мовлення та мовців можуть також допомогти виявити специфічні характеристики мовлення, пов'язані з фізіологічними, нервовими чи психологічними станами особи. Наприклад, можуть бути виявлені ознаки стресу, паніки або брехні, що можуть свідчити про участь особи в злочині. Це може служити додатковим доказом при розслідуванні. [11, 12, 26]

6. Порівняння з архівом голосів. Якщо є архів голосів підозрюваних осіб, технологія розпізнавання мовців може використовуватись для порівняння голосів із архівом. Це може допомогти в ідентифікації осіб, які були раніше пов'язані зі злочинами або незаконною діяльністю [29].

7. Відновлення аудіозаписів. У разі, якщо аудіозаписи, пов'язані зі злочином, мають низьку якість або пошкоджені, технології розпізнавання мовлення та мовців можуть бути використані для відновлення цих записів. За допомогою алгоритмів обробки сигналу та покращення якості звуку, можна знизити рівень шумів, фільтрувати перешкоди та покращити зрозумілість мовлення на аудіозаписах [42].

8. Аналіз акцентів та мовних особливостей. Технології розпізнавання мовлення та мовців можуть розпізнавати акценти та мовні особливості особи, які можуть бути важливими при розслідуванні злочинів, особливо у випадках, коли злочинці мають специфічні мовні риси або акценти, пов'язані з певним регіоном або мовною групою [61].

9. Виявлення голосів відомих осіб. Якщо в архіві наявні голоси відомих осіб, таких як публічні діячі або популярні особистості, технологія розпізнавання мовців може використовуватись для виявлення їхніх голосів

у звукових записах, що може бути корисним для розслідування, аналізу чи відстеження їхньої участі в певних ситуаціях або злочинів [29].

Актуальність роботи зумовлена потребою в автоматичному аналізі великої кількості аудіозаписів, зокрема в автоматичному розпізнаванні мовців на цих записах, що може значно спростити та підвищити точність розслідування воєнних злочинів. Технологія створення автоматичної системи діаризації мовців допоможе отримати об'єктивні докази, ідентифікувати осіб, проводити аналіз мовлення та забезпечувати більш ефективний процес розслідування.

Мета роботи – розробити програмне забезпечення для діаризації звукових записів, які були перехоплені в ході російського повномасштабного вторгнення в Україну (з лютого 2022 року), щоб співвіднести певні висловлення в розмовах із конкретними мовцями.

Досягнення мети передбачає виконання таких **завдань**:

1. Зробити огляд наукових праць та описати системи розпізнавання мовців і їх різновиди;
2. Зробити огляд історії розвитку технології розпізнавання мовців;
3. Проаналізувати та описати сучасні системи розпізнавання мовців, їх структуру та основні компоненти;
4. Описати корпус усного мовлення для подальшого його використання у створенні програмного забезпечення для діаризації;
5. Підготувати дані для діаризації. Створити парсер-конвертатор анотаційних файлів TextGrid у файли RTTM;
6. Провести діаризацію звукових записів: використати моделі розпізнавання мовлення NeMo;
7. Протестувати та оцінити різні параметри моделі NeMo, визначити найкращі параметри.

Об’єктом дослідження є аудіозаписи із розмовами військовослужбовців країни-агресора, перехоплені в ході російського повномасштабного вторгнення в Україну (починаючи з лютого 2022 року).

Предметом дослідження є процес діаризації мовців.

Матеріалом дослідження є попередньо зібраний та проанотований корпус аудіозаписів-перехоплень російською мовою, що налічує 151 аудіозапис та має загальну тривалість 12 годин 8 хвилин, а також анотації до кожного аудіозапису у форматі TextGrid.

Практичне значення цієї роботи полягає у тому, що наразі отримано безліч аудіозаписів-перехоплень із розмовами російських військовослужбовців. Якщо записи будуть оброблятись фахівцями без автоматизації, то це буде дуже тривалим та дорогим процесом. Отже, усі записи потрібно автоматизовано обробити, зокрема: провести автоматичну ідентифікацію та автоматичну верифікацію мовців.

Теоретичне значення – розвиток технології розпізнавання мовців в умовах дефіциту подібних інструментів для української та російської мов. Використання та тестування математичної моделі для ідентифікації мовців, яка не залежить від тієї чи іншої мови, може змінити стан справ і відкрити шлях до використання цього інструменту для досліджень матеріалів українською та російською мовами.

Новизна роботи полягає в тому, що вперше буде створено та протестовано програмне забезпечення для автоматизованої ідентифікації мовців на основі записів-перехоплень в ході російсько-української війни.

Методи дослідження: метод аналізу, метод порівняння, експериментальний метод, метод машинного навчання, метод статистичного аналізу.

Робота складається зі вступу, трьох розділів, висновків до розділів, загальних висновків, списку використаних джерел із 73 позицій та 5 додатків. Загальний обсяг роботи – 80 сторінок.

У вступі зазначено актуальність та новизну, мету, об'єкт і предмет, матеріали дослідження, завдання роботи, методи та структуру, теоретичне і практичне значення.

У першому розділі описано класифікацію систем розпізнавання мовців, визначено поняття верифікації та ідентифікації мовців, надано довідку з історії розвитку цих технологій та визначено поняття діаризації. Висвітлено недоліки та труднощі, які існують у цій технології, а також відзначено її широкі можливості та перспективи в різних сферах застосування.

У другому розділі описано характеристики акустичного корпусу, який був завчасно зібраний та проанотований, визначено кроки для підготовки даних для подальшої діаризації, а також підготовано дані. Створено парсер-конвертатор анотаційних файлів.

У третьому розділі описано модель NeMo для подальшого використання, створено програмне забезпечення, яке використовує модель NeMo для діаризації зі стандартними та експериментальними параметрами, протестовано та оцінено результати за метрикою DER (Diarization error rate – коефіцієнт помилок діаризації).

У висновках стисло описано основні теоретичні поняття, необхідні для забезпечення експерименту, та результати практичної частини роботи.

РОЗДІЛ 1. СИСТЕМИ РОЗПІЗНАВАННЯ МОВЦІВ

Розпізнавання мовців (англ. "*speaker recognition*") **або розпізнавання голосу** (англ. "*voice recognition*") — процес автоматичного розпізнавання того, хто говорить у відео чи звукозаписі, на основі інформації, отриманої з аналізу акустичного сигналу [30, 31].

Ця технологія може використовуватись не лише у криміналістичній експертизі, про що йшлося у Вступі, але й для інших важливих задач, пов'язаних з автоматичним обробленням звукових сигналів.

Системи біометричного розпізнавання все частіше використовуються як більш «природні» засоби для розпізнавання людей. Замість того, щоб запам'ятовувати паролі та PIN-коди (які можна вкрасти або забути) чи письмові підписи (які можна підробити), біометричні ознаки, такі як відбитки пальців, голос і обличчя, є специфічними для окремої людини (і, отже, їх неможливо легко вкрасти чи підробити) і характеризують цю особу (і, отже, не можуть бути забутими) [48]. Тож технологія розпізнавання мовців може служити для надання доступу до систем авторизованим користувачам [15].

Також технологія розпізнавання мовців може слугувати *для контролю доступу голосом у різних системах* (наприклад, голосовий набір, покупки через телефон, послуги бронювання, голосова пошта, контроль доступу до зон конфіденційних даних, доступ до віддалених комп'ютерів, вхід до хмарних сховищ, вхід у приміщення з обмеженим доступом та ін.) [13].

Система відвідування студентів та система відвідування працівників можуть бути розроблені з використанням методів розпізнавання мовців [9].

У банківському секторі для автентифікації можна використовувати системи розпізнавання мовців. Користувачів можна легко перевірити через телефон та належним чином поводитися з ними під час телеметричних

банківських операцій. Кол-центри також можуть успадкувати повторну ідентифікацію мовця для персоналізованих послуг і запитів [38].

Розпізнавання мовців має значний вплив на телекомунікаційні системи. Розпізнавання мовців може ідентифікувати невідомих абонентів на основі їхніх наявних голосових профілів у певній базі даних. Крім того, система розпізнавання може бути реалізована для автоматичного блокування невідомих абонентів за допомогою наявної бази даних відомих голосових профілів [36].

Цифровим помічникам і смартфонам часто потрібна система верифікації користувачів. Використовуючи верифікацію мовця, цифрові помічники можуть легко перевірити фактичного користувача. Смартфони також можуть успадкувати системи верифікації мовців, щоб розблокувати пристрій, навіть не торкаючись і не дивлячись у камеру. У випадку багатокористувацької платформи пристрій може розпізнавати кількох користувачів і надавати персоналізовані функції кожному за допомогою ідентифікації мовців [38].

Системи розпізнавання мовців покликані спростити побутове життя людей та зробити дослідження усного мовлення точнішими та простішими. Наразі можна передбачити ще поширеніше використання цієї технології у різноманітних сервісах.

У цьому розділі ми опишемо класифікацію систем розпізнавання мовців, зробимо огляд історії розвитку технології автоматичного розпізнавання мовців, а також схарактеризуємо роль машинного навчання у розвитку технології. Розглянемо різницю між поняттями ідентифікації мовців та верифікації мовців, а також різницю між системами розпізнавання відкритого набору та закритого набору.

Це дасть нам змогу визначити хід роботи у практичній частині та обрати інструментальні засоби для реалізації програмного забезпечення для ідентифікації мовців.

1.1 Класифікація систем розпізнавання мовців

Розпізнавання *мовлення*, розпізнавання *мови*, розпізнавання *мовців – не* тотожні процеси та технології. Розпізнавання *мовлення* означає отримання текстового контенту, автоматичну побудову стенограми сказаного мовцем. Розпізнавання *мови* означає визначення мови, якою говорить мовець. За Садаокі Фуруї, розпізнавання мовця (*speaker recognition*) поділяється на ідентифікацію мовця (*speaker identification*) та верифікацію мовця (*speaker verification) (Рис. 1.1.1)* [30].

Принципова відмінність між ідентифікацією та верифікацією полягає в кількості альтернативних рішень: при ідентифікації кількість альтернативних рішень дорівнює розміру сукупності, тоді як при верифікації є два альтернативні рішення, прийняти або відхилити, незалежно від розміру сукупності [30]. Це означає, що *ідентифікація* мовців – задача *мультикласової* класифікації, а *верифікація* мовців – задача *бінарної* класифікації [9]. Як правило, вхідними даними для ідентифікації мовців є N мовців та уривок мовлення. Система повинна видати результат, який називає одного мовця, авторству якого найімовірніше належить певний уривок на аудіозаписі. При задачі верифікації, вхідними даними є уривок мовлення та один мовець. Програма повинна повернути булеве значення (“так” або “ні”), яке нас інформує про те, чи належать слова запропонованому мовцю, чи ні.

Головна мета автоматичної діаризації полягає в ідентифікації того, коли кожен мовець говорить у записі. Результатом автоматичної діаризації є визначення розмовного контексту та відокремлення різних голосів у записі. Автоматична діаризація фокусується на розподілі різних голосів у записі, тоді як автоматична ідентифікація спрямована на визначення конкретної особи, що говорить.

Окрім поділу систем розпізнавання мовлення на системи ідентифікації, верифікації, та діаризації, існує також поділ за критерієм закритості/відкритості [38] (*Рис. 1.1.1)*.

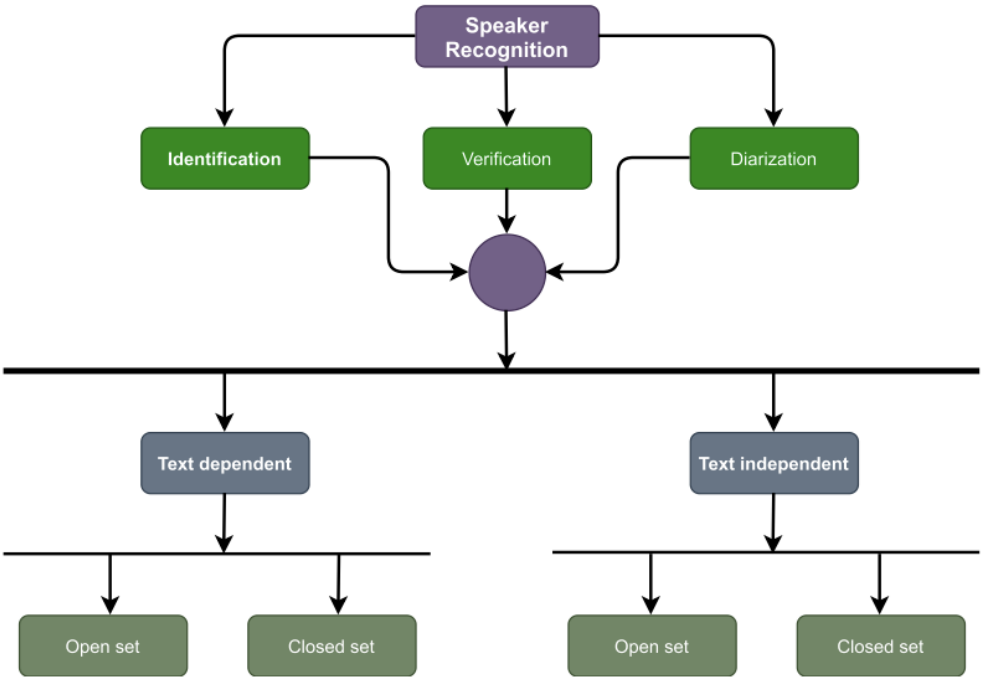


Рис. 1.1.1. Схема класифікації систем розпізнавання мовців

У розпізнаванні *закритого набору (англ. "closed set")* класифікатор призначений для розпізнавання лише фіксованого набору попередньо визначених класів або категорій. Класифікатор навчається на позначеному наборі даних, де кожен зразок належить до одного з відомих класів. На етапі тестування класифікатор призначає мітку класу кожному вхідному зразку, припускаючи, що зразок належить до одного з відомих класів. У розпізнаванні *закритого набору* припускається, що всі можливі класи відомі та визначені заздалегідь, а мета класифікатора полягає в тому, щоб правильно класифікувати зразки в ці відомі класи.

Розпізнавання *відкритого набору (англ. "open set")*, також відоме як розпізнавання відкритого світу *(англ. "open world")*, розширює концепцію розпізнавання закритого набору для обробки сценарію, де під час

тестування можуть бути присутні невідомі або нові класи, які не були частиною навчального набору. У розпізнаванні відкритого набору класифікатор навчається на позначених зразках із відомих класів, але також розроблений для усвідомлення можливості зустрічі з невідомими класами під час тестування. Класифікатор повинен мати можливість розрізняти відомі класи та відхиляти або призначати ярлик «невідомий» зразкам, які не відповідають жодному з відомих класів. Розпізнавання відкритого набору є більш гнучким і надійним у сценаріях реального світу, де класи можуть розвиватися або з часом можуть з'являтися нові класи [38].

Зазвичай системи розпізнавання мовців складаються з трьох основних етапів [34]:

1. *Feature extraction* – добування основних рис усіх мовців;
2. *Speaker modelling* – створення математичної моделі кожного з мовців;
3. *Matching* – зіставлення поточного уривка мовлення з математичною моделлю кожного мовця та генерування результату ідентифікації/верифікації/діаризації мовців.

В реальних системах кількість етапів може дещо варіюватись (найчастіше етапів більше, ніж описано вище), адже поле для експериментів є досить великим, і науковці все ще намагаються розробити оптимальне етапування і створити допоміжні модулі для технології розпізнавання мовців.

1.1.1 Ідентифікація мовців

Ідентифікація мовців покликана визначити одного мовця з числа *N* запропонованих мовців [33]. Наприклад, результат ідентифікації мовців дасть відповідь на запитання:

- чиї слова звучали на уривку аудіозапису з 1 по 10 секунду?;
- хто вимовив фразу “завтра на Беларусь”?

Щоб досягти результатів в задачі ідентифікації мовців, у нас повинні бути наявні такі дані: наприклад, у цій розмові брали участь Іван, Дмитро, Петро, Василь. Як результат, система ідентифікації мовців видасть відповідь, назвавши одного учасника розмови, наприклад, Петра.

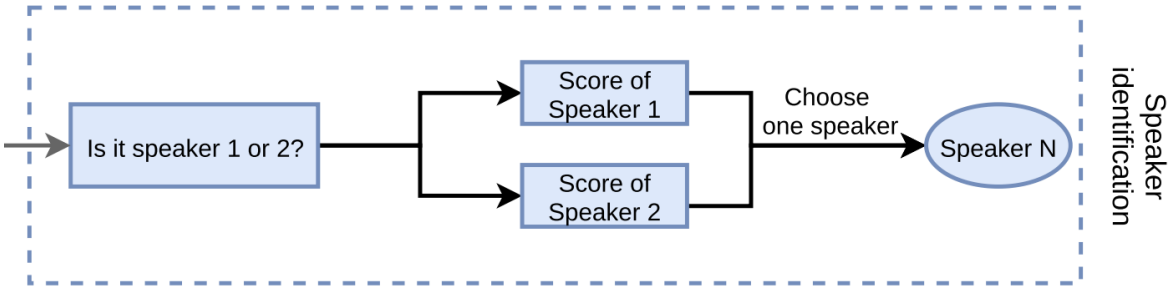


Рис. 1.1.1.1. Примітивна блок-схема ідентифікації мовця [9, с. 4]

Системи ідентифікації мовців також можуть бути впроваджені таємно без відома користувача, щоб ідентифікувати учасників дискусії, сповіщати автоматизовані системи про зміни мовців, перевіряти, чи користувач уже зареєстрований у системі тощо. Ці системи працюють зі знаннями користувачів і зазвичай вимагають їхньої співпраці [27].

Під час ідентифікації мовців використовується процес визначення особистості людини на основі її мовлення. Процес включає *два основних етапи (Рис. 1.1.1.2)*.

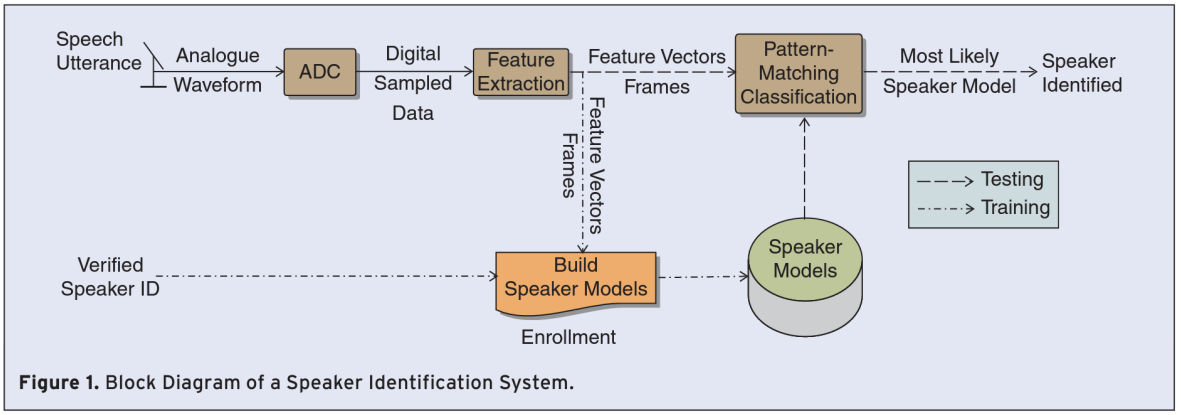


Figure 1. Block Diagram of a Speaker Identification System.

*Рис. 1.1.1.2. Блок-схема процесу ідентифікації мов*ців [48, с. 24]

Під час етапу *навчання (англ. "training")*, також відомого як *залучення (англ. "enrolment"), зб*ираються зразки мовлення відомих і перевірених осіб, щоб створити модель, специфічну для кожного мовця. Цей етап навчання зазвичай відбувається в автономному режимі, як частина конфігурації системи, перед розгортанням. На етапі *тестування (англ. "testing")* система запускається, і невідомий зразок мовлення порівнюється з навченими моделями мовців. При ідентифікації закритого типу очікується, що невідома особа належатиме до заздалегідь визначеного пулу або бази даних мовців, і завдання полягає в тому, щоб визначити, від якого мовця з пулу походить мовлення. Пул – це група об'єднаних мовців. Основним показником ефективності для таких систем є коефіцієнт ідентифікації, який представляє відсоток правильних ідентифікацій, середній для всіх мовців у пулі [48].

Ідентифікація закритого типу зазвичай використовується в організаціях, де члени групи відомі, їхні профілі мовців можна отримати та зберегти в базі даних, а процес ідентифікації обмежується внутрішнім використанням без залучення зовнішніх користувачів. З іншого боку, відкрита ідентифікація стосується випадків, коли невідома особа може походити ззовні. Однак, оскільки ідентифікація завжди виконується щодо обмеженої та відомої кількості осіб, неможливо ідентифікувати довільних

людей. Тому початкове завдання системи ідентифікації відкритого набору полягає в тому, щоб визначити, чи належить доповідач до групи зареєстрованих користувачів чи бази даних відомих доповідачів. Якщо ні, доповідач відхиляється. Якщо профіль мовця знайдено в пулі, тоді виконується ідентифікація закритого набору. Для цих систем вкрай важливо точно визначити, чи належить мовець до відомого пулу; інакше існує ризик постійної неправильної ідентифікації випадкової особи з пулу [48].

1.1.2 *Верифікація мовця*

Верифікація мовця покликана підтвердити або заперечити, що мовцем є запропонована кандидатура. Тоді перед системою верифікації мовця ставиться запитання “Чи справді певні слова сказав користувач Василь?”. Система надасть результат “так” або “ні”, але насправді ці системи не оперують булевими значеннями, а надають числові показники, які містять ймовірність, яка вказує на те, що автором слів є або не є Василь.

До прикладу, верифікацію мовця використовує “ПриватБанк”. Поки користувач розмовляє телефоном з оператором або чат-ботом, система автоматично розпізнає, чи є користувач клієнтом банку [6] (Рис. 1.1.2.1).

Рис. 1.1.2.1. Використання “ПриватБанком” верифікації користувача під час розмови з оператором кол-центру

Верифікація мовця поділяється на *залежну* від тексту (*text-dependent*) та *незалежну* від тексту (*text-independent*). Незалежна від тексту верифікація є актуальною темою в галузі обробки мовних сигналів [3, 9, 20, 32]. При залежній від тексту верифікації система верифікації мовця має попередні знання про текст, який буде вимовлено, і очікує, що користувач промовить цей текст. Однак у текстово-незалежній системі немає

попередніх знань про текст, який буде вимовлено, і від користувача не очікується співпраця [67].

Досвід науковців показує, що залежна від тексту верифікація дає точніші та швидші результати, однак, щоб створити якісну систему, нам потрібно зібрати велику кількість даних у домені, що на практиці дуже дорого [71]. Системи, що залежать від тексту, досягають високої продуктивності верифікації мовця відносно коротких висловлювань, тоді як системи, що не залежать від тексту, вимагають довгих висловлювань для навчання надійних моделей і досягнення хорошої продуктивності [63, 67]. У криміналістичній експертизі зазвичай спочатку виконується процес ідентифікації мовців, щоб створити список «найкращих збігів», а потім проводиться серія перевірок для визначення переконливого збігу. Працюючи над тим, щоб зіставити зразки від мовця зі списком найкращих збігів, можна визначити, чи є це та сама особа на основі кількості подібностей чи відмінностей. Сторони обвинувачення та захисту можуть використовувати це як доказ, щоб вирішити, чи справді підозрюваний є злочинцем, чи ні [58]. Під час верифікації мовця (Рис. 1.1.2.2) мовлення людини використовується для підтвердження чи відхилення твердження, що на записі говорить саме ця особа. Як і у випадку ідентифікації мовців, початкове налаштування системи виконується під час навчання або реєстрації, коли кожен мовець, який перевіряється системою, має надати зразки мовлення, які потім використовуються для навчання моделі для цього мовця. Під час тестування верифікація відбувається, коли особа має заявити про те, ким вона є, а потім система перевіряє, чи є ця заява правдивою, чи хибною. Під час верифікації мовця мовлення невідомої особи порівнюється як із заявленою особою, так і з усіма іншими мовцями. Потім береться співвідношення двох показників і порівнюється з пороговим значенням; якщо значення вище порогового значення,

твердження вважається істинним, якщо нижче, твердження відхиляється як хибне.

У системах верифікації популярні два ключові показники ефективності:

- *коефіцієнт помилкової відмови (FRR)* – кількість разів, коли справжній мовець був неправильно відхилений,
- *коефіцієнт помилкового прийняття (FAR)* – кількість разів, коли помилково прийнятий "мовець-самозванець".

Змінюючи порогове значення рішення, FAR і FRR змінюватимуться в протилежних напрямках. Наприклад, підвищення порогового значення знизить FAR, але збільшить FRR, оскільки правдиві випадки почнуть відхилятися, оскільки «планка» піднята, і навпаки, якщо порогове значення знижено, FRR зменшується, але FAR збільшиться, оскільки тепер не тільки всі правдиві випадки прийнято, але буде й більше фальшивих. Збалансованою пропорцією для вибору порогу є FAR = FRR, що називається *рівною частотою помилок (EER)*.

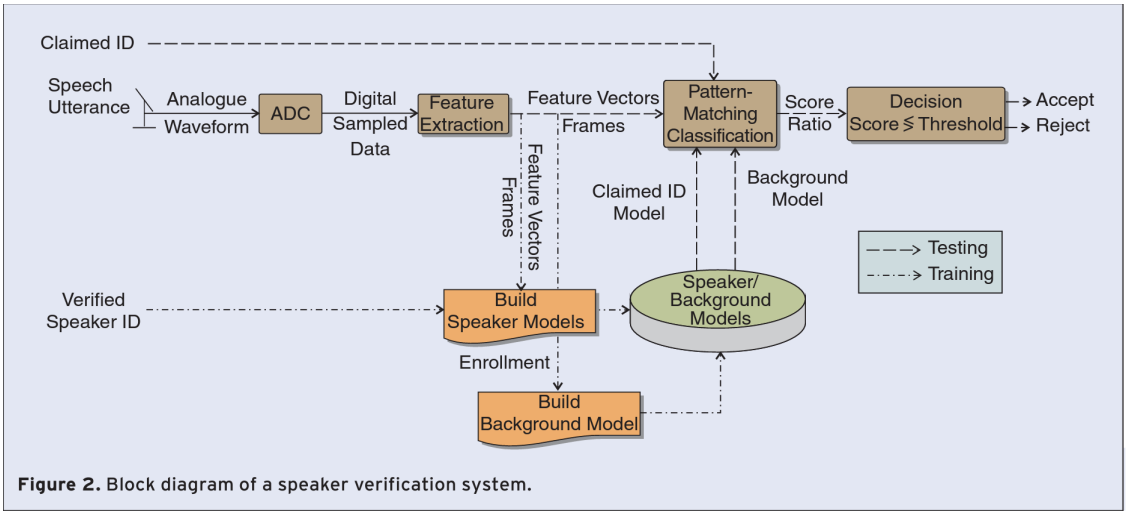


Figure 2. Block diagram of a speaker verification system.

Рис. 1.1.2.2. Блок-схема процесу верифікації мовця [48, с. 25]

1.1.3 Діаризація

Діаризація *(англ. "speaker diarizarion")* — це завдання позначити аудіо- чи відеозаписи класами, які відповідають ідентичності мовця. Діаризація дає нам відповідь на запитання *"Хто говорить коли?" (Рис. 1.1.3.1)*.

За словником Мерріам-Вебстер, "to diarize" означає робити запис у щоденник [23].

Аудіодіаризація — це процес анотування вхідного аудіоканалу інформацією, яка приписує (можливо, перекриваючи) області сигналу їхнім конкретним джерелам. Ці джерела можуть бути певними мовцями, музикою та фоновим шумом. Діаризація може бути використана для сприяння розпізнаванню мовлення, полегшення пошуку та індексування аудіоархівів, а також збільшення насиченості автоматичних транскрипцій, що робить їх більш читабельними. Діаризація має застосування в багатьох сферах, таких як субтитри до відео, розуміння змісту будь-яких розмов тощо [38, 51].

Аудіодіаризація грає важливу роль у системах розпізнавання мовців, оскільки дозволяє відокремити голоси різних учасників розмови та визначити, коли кожен мовець говорить. Вона забезпечує розрізнення між голосами різних осіб і створює контекст для подальшого аналізу та ідентифікації мовців. Діаризація мовців поділяє аудіозапис із кількома людьми на однорідні сегменти, пов'язані з кожною особою. Це – невіддільна частина системи розпізнавання мовців.

Раніше були розроблені алгоритми діаризації мовців для розпізнавання мовлення на аудіозаписах із кількома мовцями, щоб забезпечити адаптивну обробку мовців. Зовсім недавно, з появою технології глибокого навчання, яка спричинила революційні зміни в

дослідженнях і практиках у різних сферах обробітку мовлення, було досягнуто швидкого прогресу в діаризації мовців [8].

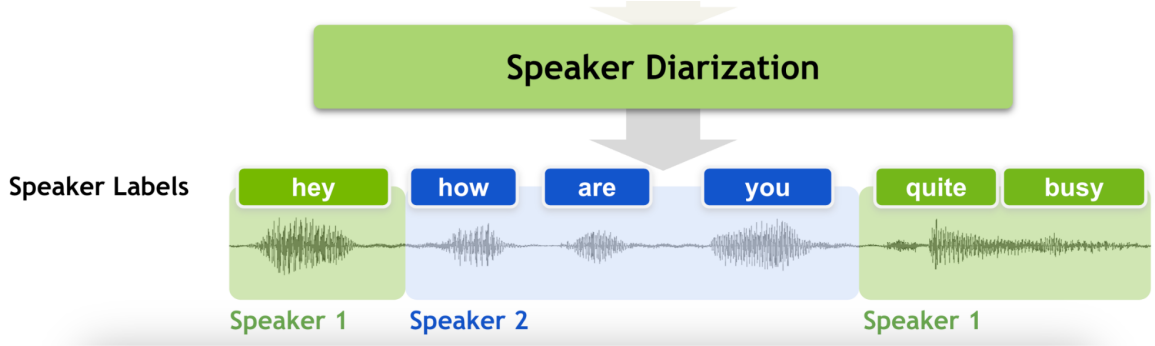


Рис. 1.1.3.1. Приклад результату діаризації уривку мовлення [56]

Діаризація мовця з розпізнаванням мовлення може служити для створення специфічних транскрипцій для конкретної особи. Такі системи мають значення для судів, адвокатів, суддів. Такі системи допомагають навіть створювати автоматизовані стенограми корпоративних зустрічей. Часто такі системи допомагають розробляти автоматизовані програми для нотаток будь-якої конкретної розмови. Крім того, можна розробити додатки, які можуть активувати сигнал тривоги на основі будь-якого критичного стану людини шляхом розпізнавання стану мовця та ідентифікації мовців [9].

1.2 Історія технології автоматичного розпізнавання мовців

Середина 20 століття, зокрема період між 1940-ми та 1960-ми роками, була важливим часом для розвитку технології розпізнавання мовців. У цю епоху науковці почали досліджувати різні підходи до аналізу та ідентифікації людей на основі їхніх голосових характеристик.

У 1940-х роках дослідники почали використовувати *спектрограми*, які являють собою візуальні зображення звуків, для вивчення акустичних властивостей мови. Дослідження, які стосувалися задачі розпізнавання голосу, вперше були проведені у 1937 році Франческою Макгі. В 1942 в лабораторії Белла відкрили поняття спектрограми, застосування якої є дуже широким у задачах аудіоаналізу. Спектрограми дали зрозуміти частотний вміст і часові моделі мовленнєвих сигналів, служачи основою для ранніх спроб розпізнавання мовців [37].

Наприкінці 1940-х років доктор Лоуренс Г. Керста ввів поняття *«відбитку голосу»* [37]. Він припустив, що кожна людина має унікальну голосову анатомію та мовні звички, які можна візуалізувати як характерні візерунки на спектрограмах. Відбитки голосу порівнювали та аналізували вручну, щоб визначити ідентифікацію мовця.

Цікавим є те, що навіть зараз, у 20-х роках 21 століття, технологія відбитка голосу є досі актуальною. Ця технологія є потрібною та водночас нелегкою для впровадження в інші інтелектуальні системи. Процитуємо автора статті "*Alexa, What Happened to My Car?*" в цифровому виданні *New York Times,* де він обурюється на відсутність біометричних методів безпеки голосового помічника *Alexa* від *Amazon (*тут і далі переклад наш. — В. Шемчук*)*:

"І навіть попри те, що голосових ботів, таких як Alexa та Google Assistant, можна навчити розпізнавати різні голоси — достатньо добре, щоб, наприклад, обслуговувати улюблені станції Pandora кожного члена

сім'ї — вони не пропонують жодного виду біометричної безпеки, як-от аналіз голосових відбитків. Як наслідок, за словами Amazon, можливості розпізнавання голосу Alexa недостатні для цілей безпеки." [46].

Протягом 1950-х років дослідники зосередилися на виявленні акустичних особливостей, за якими можна було розрізнити мовців. Вони вивчали такі характеристики, як частота основного тону, форманти, тривалість фонем і спектральні властивості. Ці особливості були перевірені вручну, щоб знайти шаблони, характерні для окремих людей [2].

У 1960 році Гуннар Фант розробив фізіологічну модель системи продукування голосу людини, яка закладає основу аналізу мовлення [38].

У 1960-х роках, з розвитком обчислювальних технологій, дослідники почали експериментувати з автоматизованими системами розпізнавання мовців. Початкові підходи передбачали створення шаблонів або моделей окремих мовців на основі вибраних акустичних характеристик. Потім ці шаблони порівнювали з новими зразками мовлення за допомогою методів зіставлення шаблонів. Спроби створення системи розпізнавання мовців з використанням кореляції двох цифрових спектрограм були проведені також у 1960 році Прузнаським, також у лабораторії Белла [2].

У 1960-х і 1970-х роках статистичні методи набули популярності в дослідженнях розпізнавання мовців. Науковці досліджували такі методи, як статистичне розпізнавання образів, дискримінантний аналіз і кластерний аналіз, щоб визначити унікальні характеристики мовця. Ці методи мали на меті зафіксувати статистичні варіації мовних сигналів для різних мовців [2].

У 1980-х роках приховані марківські моделі (HMM) зробили революцію в розпізнаванні мовців. Ланцюг Маркова містить усі можливі стани системи та ймовірність переходу з одного стану в інший [60].

Ланцюги Маркова корисні для розпізнавання мовців, оскільки вони дозволяють моделювати стилістичні особливості мовлення, розрізняти

мовців на основі їх мовлення, адаптуватися до конкретного мовця та класифікувати мовців за різними критеріями, використовуючи статистичні властивості та інформацію про них [5, 60].

Успішні спроби верифікації мовця були здійснені у Навчальному центрі й телекомунікаційній лабораторії (CSELT) в Італії (м. Турин). У 1983 році Мікеле Кавацца та Альберто Чіарамелла подали перший глобальний патент на основі телекомунікаційного дослідження [52]. Цей патент послужив основою для майбутніх телекомунікаційних послуг, що надаються кінцевим користувачам, і для покращення методів зменшення шуму в усій мережі. Пристрій отримує кілька характерних параметрів зі стандартного висловлювання мовця і порівнює їх із середніми параметрами того ж мовця, які зберігаються у внутрішній пам'яті та попередньо обраховані. Відповідно до результатів порівняння він отримує значення ймовірності того, що вимовлене речення належить цьому мовцеві, і порівнює значення з пороговим значенням, нормалізованим до середньої дисперсії параметра схемою обчислення порогового значення. Якщо поріг подолано, пристрій вважатиме мовця верифікованим [21].

У 1987 році компанія Worlds of Wonder випустила ляльку Джулі, яка була одним з перших комерційних прикладів використання технології навчання [52]. Відповідно до опису ляльки, вона могла навчитись розпізнавати голос конкретної дитини та реагувати на нього. Попри це, в рекламі використовувалась фраза "Нарешті лялька, яка тебе розуміє".

У 1996-1998 роках технологія розпізнавання мовців була уведена в експлуатацію в прикордонному переході Скобі-Коронач. Це дозволило зареєстрованим місцевим жителям, яким не було що декларувати, перетинати канадсько-американський кордон після того, як станції спостереження зачинялись на ночівлю [52]. Програмне забезпечення, яке використовувалось у переході через кордон, було створене компанією-бізнесом з розпізнавання голосу *Nuance* [43], яка також

розробляє програмне забезпечення для *Siri* – голосового помічника в продуктах *Apple* [4]. Наразі компанією *Nuance* володіє технологічний гігант *Microsoft* [44].

У 2000-х роках спостерігався значний розвиток систем розпізнавання мовців. Зокрема, зростала точність та надійність таких систем завдяки впровадженню нових технологій і підходів.

Одним з важливих напрямків у розвитку систем розпізнавання мовців було використання глибокого навчання. Глибокі нейронні мережі стали популярним інструментом у сфері розпізнавання мовців. Вони дозволили враховувати складні шаблони й особливості у вимові людей, покращуючи точність розпізнавання [9].

Також у 2000-х роках з’явилися нові методи та алгоритми для вилучення корисних ознак зі звукових сигналів мовлення.

Наприклад, використання мел-частотних кепстральних коефіцієнтів (MFCC) стало популярним методом для представлення звуків мовлення [62]. Ці коефіцієнти використовуються для виявлення важливих акустичних характеристик зі звукових сигналів, що допомагає покращити якість розпізнавання мовлення. MFCC (Mel Frequency Cepstral Coefficients) — це метод аналізу і представлення звукових сигналів мовлення, який широко використовується в області розпізнавання мовців та обробки мовлення. Він дозволяє ефективно виділити корисні акустичні характеристики з амплітудного спектра звукового сигналу. MFCC базується на сприйнятті звуку людиною. Людському слуху властива неоднакова чутливість до різних частот звуку. Ідея MFCC полягає в тому, щоб перетворити амплітудний спектр звукового сигналу на шкалу, яка краще відповідає сприйняттю звуку людиною. MFCC дає змогу представити складні акустичні характеристики звукового сигналу мовлення у вигляді компактного набору числових ознак. Ці ознаки можна використовувати для розпізнавання мовця, ідентифікації мовленнєвих

команд, виявлення емоцій тощо. Після обчислення MFCC можна також застосовувати додаткові операції для покращення якості ознак і зменшення впливу шуму або некорисної інформації [62].

Окрім цього, у 2000-х роках з'явилися нові бази даних для навчання та тестування систем розпізнавання мовців. Ці бази даних включали записи мовлення великої кількості людей з різними акцентами, вимовами та мовами. Вони дозволили розширити можливості систем розпізнавання мовців і зробити їх більш універсальними та точними.

Зростання обчислювальних можливостей також сильно позитивно вплинуло на розвиток систем розпізнавання мовців. Поява потужних процесорів і високошвидкісних обчислювальних систем дозволила знизити час обробітку аудіо та підвищити точність розпізнавання.

1.2.1 Роль сучасного машинного навчання у розпізнаванні мовців

Ідентифікація мовця – це класифікаційне завдання, метою якого є ідентифікація суб'єкта з певного часового ряду послідовних даних. Оскільки мовленнєвий сигнал є безперервним одновимірним часовим рядом, більшість сучасних методів дослідження базуються на згортковій нейронній мережі (CNN) або рекурентній нейронній мережі (RNN) [55, 72, 39].

Сучасний стан машинного навчання дозволяє дотренувати модель відповідно до певних даних, причому очікується, що точність дотренованої моделі буде вищою, ніж точність вихідної моделі. Наразі машинне навчання використовується в усіх етапах розпізнавання мовців:

- в анотації даних (якщо для мови існують окремі моделі, що роблять розпізнавання мовлення та якщо модуль розпізнавання підтримує результати цієї моделі),
- для відсіювання шуму в аудіозаписах,
- для виділення основних рис мовця (feature extraction),
- для кластеризації (розрахунок кількості мовців на аудіозаписі),
- порівняння результатів виконання моделі з результатом-еталоном.

Попри те, що розпізнавання мовців на основі глибокого навчання досягло великого успіху, багато проблем ще належить вирішити [73].

Для навчання нейронної мережі більшість методів виділення рис мовця потребують вхідних даних ручної роботи, що може бути неоптимальним. Сучасні моделі глибокого навчання мають велику кількість параметрів, які важко застосувати до портативних пристроїв. Навчання мережі також потребує великої кількості анотованих навчальних даних і значних обчислювальних ресурсів.

У задачі розпізнавання мовців використовуються різноманітні функції втрат (англ. "*loss functions"*), які оцінюють різницю між

прогнозованими та фактичними значеннями. Основна мета полягає в тому, щоб зменшити цю різницю і покращити точність розпізнавання мовця. Попри те, що запропоновано так багато функцій втрат, немає міцної теоретичної бази для успіху функцій втрат, ані теоретичних вказівок, які могли б привести до кращих функцій втрат. Хоча втрати верифікації для наскрізної перевірки мовців добре відповідають процесу верифікації, їхній потенціал ще не повністю розроблений [73].

Що стосується діаризації в реальному світі, можна побачити, що діаризація мовця все ще є складною проблемою. Під час розмови в реальному світі запис мовлення може бути забруднений серйозним накладенням мовлення та сильним фоновим шумом. Деякі технічно складні проблеми, такі як невідома кількість мовців і швидка зміна мовців, також серйозно перешкоджають виконанню діаризації мовців у реальних програмах. Нарешті, хоча було запропоновано багато алгоритмів адаптації до домену, особливо тих, що базуються на змагальному навчанні, вони не досягли значного прогресу порівняно з традиційними методами неглибокої адаптації [73].

ВИСНОВКИ ДО РОЗДІЛУ 1

У цьому розділі було окреслено класифікацію систем розпізнавання мовців, визначено поняття верифікації та ідентифікації мовця та надано довідку з історії розвитку цих технологій. Також ми дали визначення поняття діаризації. Діаризація – це важливий складник у технології розпізнаванні мовців, тому ми будемо проводити саме діаризацію у наступних розділах, оскільки ми повинні визначити межі кожного уривку, сказаного певним мовцем.

Крім цього, наведено аргументи на підтвердження того, що ця технологія, попри значні наукові прориви, все ще має недоліки та труднощі, які варті уваги та майбутніх досліджень.

Технологія розпізнавання мовців знайшла безліч сфер застосування і безумовно має великі перспективи у майбутньому.

РОЗДІЛ 2. ПІДГОТОВКА ДАНИХ ДО ДІАРИЗАЦІЇ МОВЦІВ

У цьому розділі ми опишемо і підготуємо вихідні дані для подальшого використання моделі автоматичної діаризації мовців. Потрібно визначити, які кроки є необхідними для приведення наявних даних до такого формату, щоб їх було зручно використовувати для автоматичної діаризації мовців.

Зробимо інформаційну довідку про математичну модель, що використовуватиметься та визначимо аргументи, які визначили наш вибір на користь моделі NeMo.

2.1 Вихідні дані. Акустичний корпус з анотаціями

Вихідні дані для задачі діаризації мовців за голосом – це дані, що будуть використані як вхідний параметр та дані, що пройдуть обробку при використанні математичної моделі для діаризації мовців.

Нашими вихідними даними є попередньо розмічений акустичний корпус, що містить *151 аудіофайл у форматі .wav* (загальна тривалість записів — 12 годин 08 хвилин). Аудіофайли в цьому корпусі – це перехоплені розмови військовослужбовців-окупантів Збройних Сил Російської Федерації під час повномасштабного вторгнення в Україну (починаючи з лютого 2022 року).

Якщо говорити про статистичні дані акустичного корпусу записів-перехоплень, то маємо таку статистику:

- загальна тривалість записів у корпусі (.WAV files): 12 годин 08 хвилин 50 секунд;
- середня тривалість записів у корпусі (.WAV files): 04 хвилини 49 секунд;
- мінімальна тривалість записів у корпусі (.WAV files): 11 секунд;

- мінімальна тривалість записів у корпусі (.WAV files): 17 хвилин 38 секунд.

Кожен аудіофайл у корпусі має відповідний файл (*.TextGrid*) з анотацією.

Анотація TextGrid – це спеціальний формат файлу, який використовується для структурованого анотування та опису акустичних подій, зокрема для аналізу мовлення. Формат *TextGrid* широко використовується в комп'ютерній лінгвістиці для представлення часових міток і маркерів, які вказують на початок та кінець фонем, слів, речень або інших лінгвістичних одиниць у звуковому сигналі.

У файлі-анотації TextGrid зазвичай містяться такі елементи:

1. Таблиця шарів (рівнів) (*англ. "tier")*: Кожен шар відповідає певному виду анотації, наприклад, фонетичному шару, словниковому шару, шару речень тощо. Кожен шар містить часові мітки та відповідні маркери або текстові описи. *У наших файлах-анотаціях кожен шар (рівень) відповідає за конкретного мовця.*

2. Часові мітки: Це значення часу, яке вказує на момент відтворення або початок і кінець певного події у звуковому сигналі. Часові мітки зазвичай представлені у мілісекундах або секундах.

3. Маркери або текстові описи: Це мітки або текстові описи, які прикріплені до відповідних часових міток. Вони можуть включати фонетичні символи, слова, речення або будь-яку іншу лінгвістичну інформацію, яка анотує звуковий сигнал. У наших анотаціях містяться фрази, вимовлені мовцем, що відповідають певному уривку звукозапису.

Анотація TextGrid дозволяє зручно візуалізувати та редагувати анотаційні дані, спрощує аналіз мовлення та дослідження акустичних подій. Формат TextGrid широко використовується в програмах та інструментах для обробки та аналізу мовленнєвих даних [64].

Анотація наших даних здійснювалась вручну експертами-лінгвістами за допомогою комп'ютерної програми Praat [16].

Наприклад, ось так виглядає пара з корпусу – звуковий файл (.wav) та Praat анотація до нього (.TextGrid) (*Рис. 2.1.1)*.

Файли обов'язково повинні мати однакові імена. Для зручності пошуку по файлах та для автоматизації обробки усіх файлів у директорії була проведена уніфікація назв файлів. Якщо назва файлу містила і дефіс, і нижнє підкреслення, то обидва знаки замінялись на нижнє підкреслення. Наприклад, якщо звуковий файл називався "535530_2022_03_12_17_**45-19-1-34**.wav", а його анотація називалась "535530_2022_03_12_17_**45_19_1_34**.rttm", то перший файл ми перейменовуємо на "535530_2022_03_12_17_**45_19_1_34**.wav". Також уніфіковані назви для кожної пари "звукозапис-анотація" нам будуть потрібні для подальшої обробки файлів за допомогою моделі NeMo.

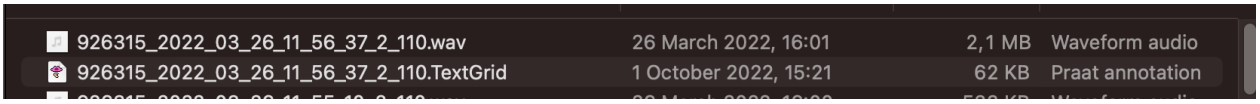


Рис. 2.1.1. Уніфіковані назви для .wav та .TextGrid файлів

За допомогою цієї ж програми ми можемо переглянути анотацію, завантаживши пару файлів (*Рис. 2.1.2)*.

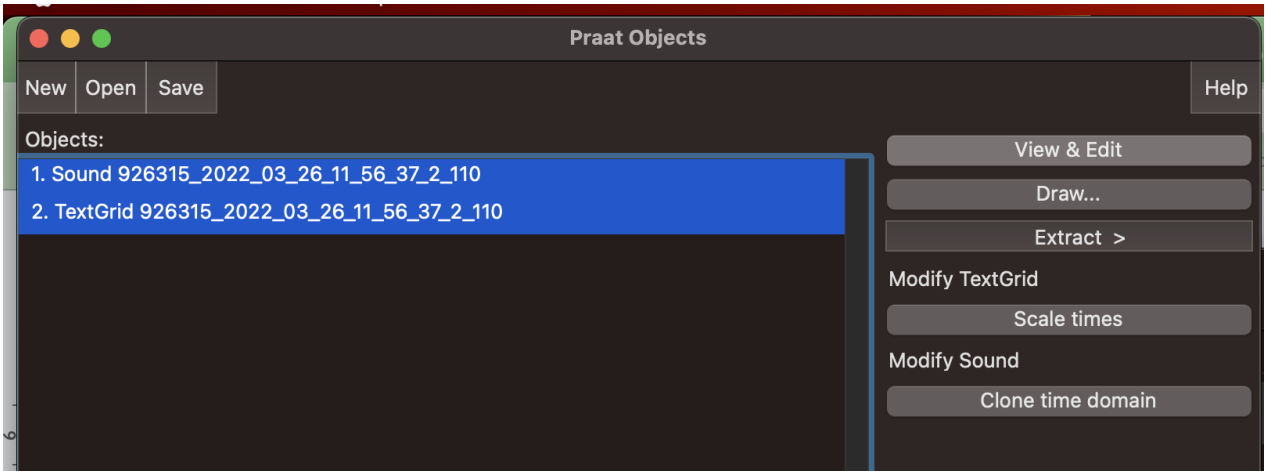


Рис. 2.1.2. .wav та .TextGrid файли, відкриті в програмі Praat, яка створює анотацію формату TextGrid

Так виглядає анотація звукового файлу (*Рис. 2.1.3*):

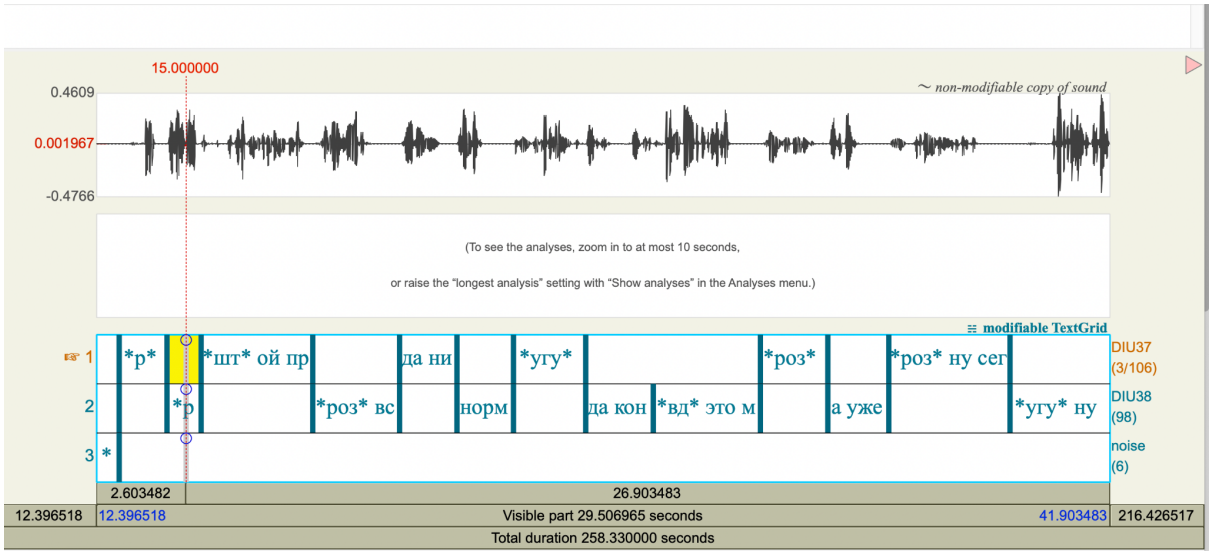


Рис. 2.1.3. Анотація у програмі Praat

Ми можемо бачити осцилограму (перше поле зверху) та три рівні анотації (розмітки). Якщо сфокусуватись на діапазоні 10 секунд, то можемо спостерігати ще й спектрограму (*Рис. 2.1.4)*.

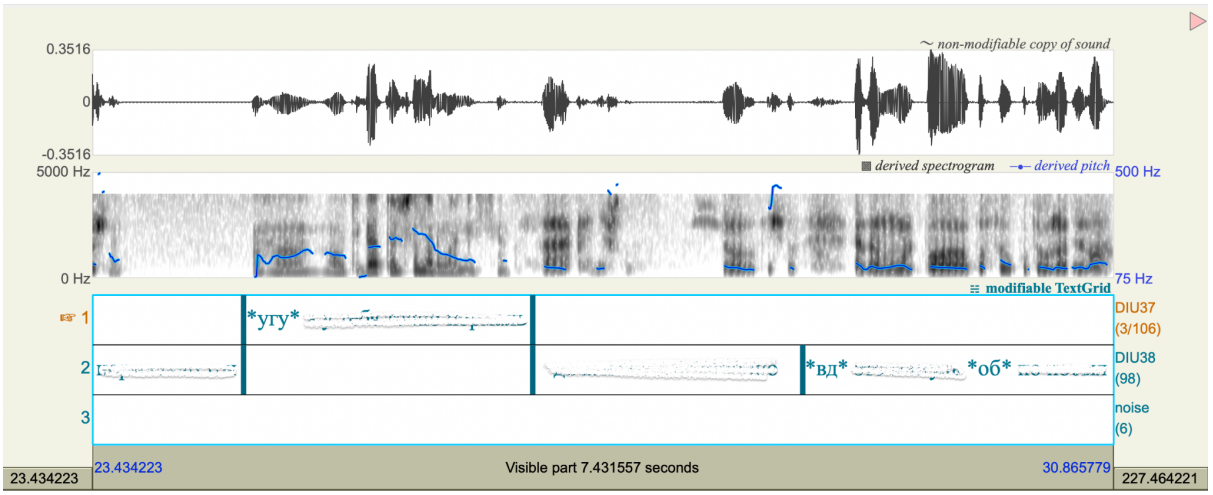


Рис. 2.1.4. Анотація десятисекундного діапазону в програмі Praat із зображенням спектрограми

Ліворуч від рівнів розмітки присутня нумерація рівнів (1, 2, 3). Кожен рівень має свою назву праворуч. Наприклад, рівень 1 має назву “DIU37”, рівень 2 має назву “DIU38” – це коди, якими позначаються диктори. Рівень 3 має назву “noise” (укр. “шум”).

Кожен рівень відповідає за мовлення окремого мовця або за шум.

У нашому корпусі в одному звуковому записі одночасно розмовляли:

- мінімум 1 мовець,
- максимум 4 мовці.

Шумом вважається, наприклад, гавкіт собак, перешкоди на телефонному зв'язку, стрілянина тощо. Якщо в розмові брали участь, наприклад, троє людей, то рівнів анотації буде чотири: по одному рівню для кожного мовця плюс один рівень для шуму.

Рівень “noise” (укр. “шум”) містить метапозначки, які записуються скорочено, між астерисками: **гуд** *(Рис. 2.1.5)*. Це – гудок. Ця метапозначка присутня майже в кожній анотації, адже майже кожен дзвінок починається з гудка.

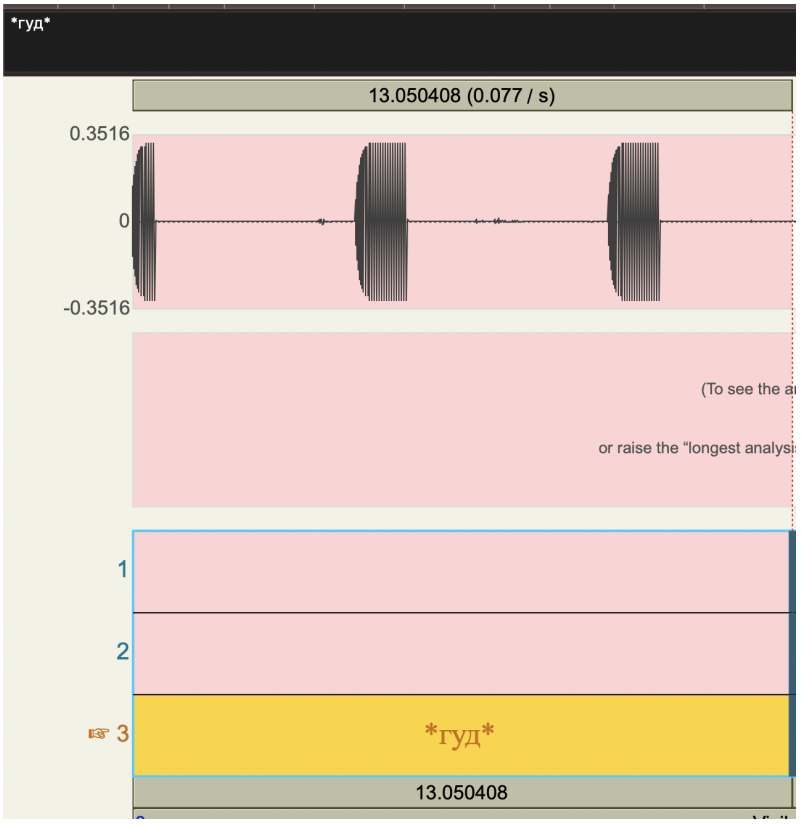


Рис. 2.1.5. Приклад метапозначки шуму

Ось приклади інших метапозначок, які характеризують шум:

шум – невизначений шум, **пік** – “запікування” обсценної лексики, **гов** – говоріння, **віт** – вітер, **виб** – вибух, **постр** –

постріл, **стук** – стукіт, **птах** – спів птахів, **пів** – спів півня, **тел** – дзвінок телефону, **спов** – звук сповіщення на телефоні. Так зване "*запікування*" пояснюється тим, що мовці в аудіозаписах послуговувались вживанням лайки та оскільки ці записи були викладені у публічний простір Головним Управлінням Розвідки, лайку замінили спеціальним сигналом.

Було розроблено скрипт, який автоматично перевіряє кожен .TextGrid файл з директорії на наявність шуму (Див. Додаток 1). Цей скрипт порівнево перевіряє вміст файлів на наявність рівня з назвою "noise". Отже, було проведено перевірку, чи кожен файл .TextGrid містить проанотований рівень шуму (*Рис. 2.1.6)*:

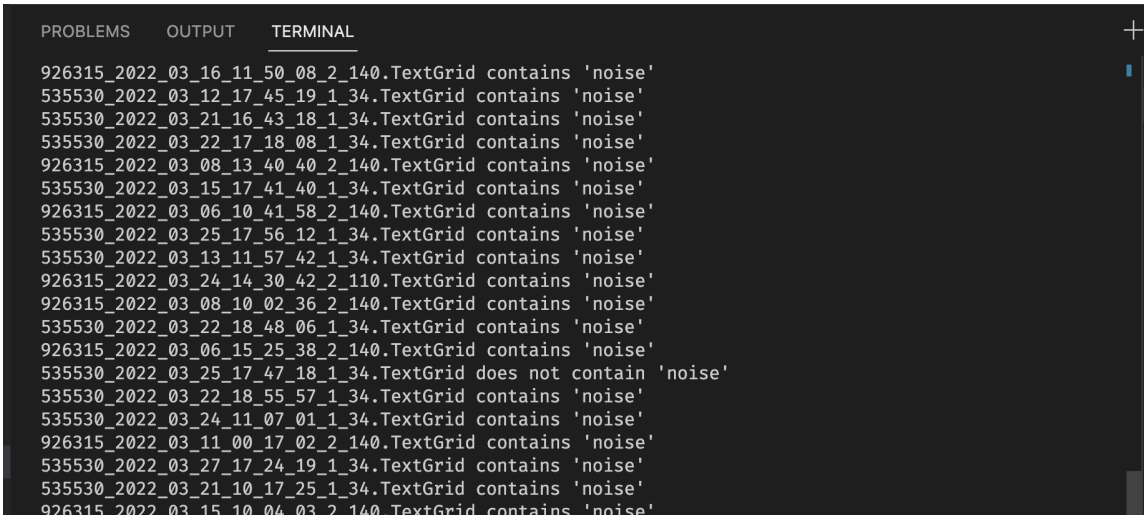


Рис. 2.1.6 Усі зі 151 файлу, крім одного, містять рівень шуму.

Виявилось, що всі файли містять рівень шуму, окрім одного файлу.

Рівень 1 та 2 теж містять метапозначки, але вони позначають мову, спосіб вимови слів, фон, паузи хезитації та стиль мовлення.

Наприклад, позначки мови (*Табл. 2.1.1)*:

****у****	Українська мова
****р****	Російська мова
****р-укр****	Російська мова з українським акцентом
****англ****	Англійська мова

нім	Німецька мова
фр	Французька мова
с	Суржик

Табл. 2.1.1 Метапозначки мови для анотацій

Позначки стилю мовлення/хезитацій (див. Додаток 2): **см** – сміх, **плач** – плач, **зітх** – зітхання, **к** – кашель, **кд** – кашель дитини, **вд** –вдих/видих, **пл** – плямкання, **шм** – шмигання носом, **ім** – імітація звуку, **йой** – йойкання, **виг** – вигук, **цьом** – звук умовного поцілунку , **плю** – плювання, **хлип** – хлипання.

2.2 Конвертація *TextGrid* файлів в *RTTM* формат

З урахуванням того, що анотація звукових записів проводилась в програмі Praat, файли з розміткою мають формат *TextGrid*. *Анотація TextGrid* – це спеціальний формат файлу, який використовується для структурованого анотування та опису акустичних подій, зокрема для аналізу мовлення.

Оскільки більшість програмних модулів з розпізнавання мовців працюють з файлами *RTTM*, необхідно провести конвертацію наявних *TextGrid* файлів у формат *RTTM*, щоб використати їх у майбутній програмі [45, 65].

RTTM (Rich Transcription Time Marked) файл — це стандарт текстового файлу, де елементи розмітки розділені пробілами, який містить по одному проанотованому уривку на рядок. Стандарт *RTTM* файлу визначений *NIST* — Національним інститутом стандартів і технологій у 2002 році [33, 35].

За стандартом, кожен рядок містить десять полів, розділених пробільними символами. Так виглядає файл RTTM [33] (*Рис. 2.2.1*):

Field 1	2	3	4	5	6	7	8	9	10
Type	file	chnl	tbeg	tdur	ortho	stype	name	conf	Slat

Рис. 2.2.1 Структура-стандарт RTTM файлу [35]

Так визначаються значення полів:

- ***type*** – тип сегмента; якщо файл використовується для задач діаризації мовця, то тип завжди має значення "*SPEAKER" (укр. "мовець");*
- ***file*** – ідентифікатор файлу; зазвичай це назва файлу без його розширення;
- ***channel*** – індекс увімкненого каналу, завжди має значення "1";

- ***tbeg*** – числове значення в секундах, час початку конкретного сегмента розмітки на звуковому записі;
- ***tdur –*** числове значення в секундах, тривалість конкретного сегмента розмітки на звуковому записі;
- ***ortho*** – поле для орфографічних поміток, завжди має значення "*<NA>*";
- ***stype*** (speaker type) – тип мовця, завжди має значення "*<NA>*";
- ***name*** (speaker name) – ім'я мовця, яке повинне бути унікальним в межах одного файлу;
- ***conf*** (Confidence Score) – числове значення того, наскільки система впевнена в правильності результату; завжди має значення "*<NA>*";
- ***slat*** (Signal Lookahead Time) – час очікування сигналу, завжди має значення "*<NA>*".

Структура *RTTM* файлу може різнитися в залежності від наявних даних, від задачі, яку потрібно виконати з використанням цього файлу та від вимог програмного забезпечення. Найнеобхіднішими елементами розмітки, які без яких RTTM не використовують, є часове значення початку уривка, тривалості уривка, поле "SPEAKER" та ідентифікатор файлу. Варто зауважити, що діаризація мовця не може бути проведена без наявності поля ідентифікатора мовця в RTTM файлах.

Для конвертації *TextGrid* в *RTTM* потрібен парсер, але таких парсерів немає у вільному доступі. Зазвичай вони вбудовані в моделі, або створюються індивідуально під задачу. Саме тому було створено парсер, який приймає *TextGrid і* повертає відповідний *RTTM* (див. Додаток 3).

Парсер використовує бібліотеку *TGT (TextGridTools*). *TextGridTools* — це безкоштовний модуль *Python* для обробки, запитів і маніпулювання файлами *Praat TextGrid* [14]. *TextGridTools* усуває багато недоліків

вбудованої мови сценаріїв Praat, надаючи чисту модель даних для об’єктів *TextGrid* та їхніх атрибутів, а також пропонуючи функціональні можливості для поширених завдань, пов’язаних з анотаціями. Завдяки повній інтеграції з іншими інструментами Python, такими як бібліотеки аналізу даних та інтерактивні інтерпретатори, користувачі отримують доступ до універсального та потужного обчислювального середовища без необхідності повторного перетворення форматів [14].

Цей парсер є програмою, яка перетворює файл формату *TextGrid* на файл формату *RTTM (Rich Transcription Time Marked)*.

Основні функції цього парсеру:

1. *textgrid2rttm(textgrid)*: ця функція приймає шлях до файлу TextGrid як вхідний параметр. Вона використовує бібліотеку *tgt* для читання файлу TextGrid. Потім вона проходиться по різних рівнях розмітки (tiers) у файлі TextGrid, отримує інформацію про кожен рівень (ідентифікатор, мітки й часові межі) і зберігає цю інформацію в словнику *rttm_out*, де ключем є ім'я рівня, а значенням є список кортежів зі значеннями початку, тривалості, фрази та ідентифікатора мовця. У Python *словник (dictionary)* є вбудованим типом даних, який використовується для зберігання колекції пар {*ключ: значення}* [24]. Він також відомий як асоціативний масив або хеш-таблиця в інших мовах програмування.
2. *write_rttm(rttm_out, full_file_name):* ця функція приймає словник *rttm_out* і повне ім'я вихідного файлу як вхідні параметри. Вона відкриває файл з розширенням ".rttm" для запису та проходиться по кожному рівню зі словника *rttm_out*. Для кожного рівня вона записує рядок у форматі RTTM, який містить інформацію про початок, тривалість, ідентифікатор мовця та шлях до вхідного файлу.
3. Остання частина коду виконується, якщо скрипт викликається напряму з командного рядка. Вона використовує модуль `argparse`

для обробки аргументів командного рядка. Скрипт очікує два аргументи: `input_file` (шлях до вхідного файлу TextGrid) і `output_file` (ім'я вихідного файлу RTTM). Після отримання аргументів викликаються функції `textgrid2rttm` і `write_rttm` для обробки файлів.

Для більшої автоматизації також створено скрипт, який приймає шлях до директорії, в якій зберігаються TextGrid файли, та шлях для зберігання файлів-результатів. Цей скрипт запускає парсер для кожного файлу в директорії.

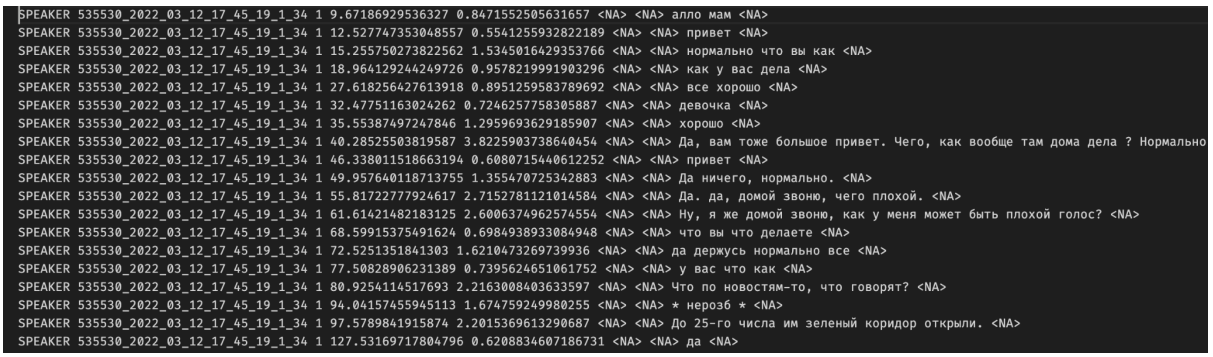
Так виглядає *TextGrid* файл, якщо ми відкриємо його у текстовому редакторі (*Рис. 2.2.2)*:

```
File type = "ooTextFile"
Object class = "TextGrid"

xmin = 0
xmax = 619.915
tiers? <exists>
size = 4
item []:
    item [1]:
        class = "IntervalTier"
        name = "KAZ49"
        xmin = 0
        xmax = 619.915
        intervals: size = 157
        intervals [1]:
            xmin = 0
            xmax = 9.67186929536327
            text = ""
        intervals [2]:
            xmin = 9.67186929536327
            xmax = 10.519024545926436
            text = "алло мам"
        intervals [3]:
            xmin = 10.519024545926436
            xmax = 12.527747353048557
            text = ""
        intervals [4]:
            xmin = 12.527747353048557
            xmax = 13.081872946330776
            text = "привет"
        intervals [5]:
            xmin = 13.081872946330776
            xmax = 15.255750273822562
            text = ""
        intervals [6]:
            xmin = 15.255750273822562
            xmax = 16.79025191675794
```

Рис. 2.2.2 Вигляд TextGrid файлу у текстовому редакторі

Ось так виглядає відповідний файл-результат формату RTTM, який ми згенерували за допомогою парсера:



```
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 9.67186929536327 0.8471552505631657 <NA> <NA> алло мам <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 12.527747353048557 0.5541255932822189 <NA> <NA> привет <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 15.255750273822562 1.5345016429353766 <NA> <NA> нормально что вы как <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 18.964129244249726 0.9578219991903296 <NA> <NA> как у вас дела <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 27.618256427613918 0.8951259583789692 <NA> <NA> все хорошо <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 32.47751163024262 0.7246257758305887 <NA> <NA> девочка <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 35.55387497247846 1.2959693629185907 <NA> <NA> хорошо <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 40.28525503819587 3.8225903738640454 <NA> <NA> Да, вам тоже большое привет. Чего, как вообще там дома дела ? Нормально
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 46.338011518663194 0.6080715440612252 <NA> <NA> привет <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 49.957640118713755 1.355470725342883 <NA> <NA> Да ничего, нормально. <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 55.81722777924617 2.7152781121014584 <NA> <NA> Да. да, домой звоню, чего плохой. <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 61.61421482183125 2.6006374962574554 <NA> <NA> Ну, я же домой звоню, как у меня может быть плохой голос? <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 68.59915375491624 0.6984938933084948 <NA> <NA> что вы что делаете <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 72.5251351841303 1.6210473269739936 <NA> <NA> да держусь нормально все <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 77.50828906231389 0.7395624651061752 <NA> <NA> у вас что как <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 80.9254114517693 2.2163008403633597 <NA> <NA> Что по новостям-то, что говорят? <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 94.04157455945113 1.674759249980255 <NA> <NA> * нерозб * <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 97.5789841915874 2.2015369613290687 <NA> <NA> До 25-го числа им зеленый коридор открыли. <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 127.53169717804796 0.6208834607186731 <NA> <NA> да <NA>
```

Рис. 2.2.3 Файл-результат формату RTTM після парсингу-конвертації файлу TextGrid

Як результат роботи коду ми отримали *151 файл-анотацію формату RTTM*.

ВИСНОВКИ ДО РОЗДІЛУ 2

У Розділі 2 описано характеристики акустичного корпусу, який був завчасно зібраний та проанотований. Акустичний корпус буде безпосередньо використовуватись у Розділі 3 – у проведенні діаризації та тестуванні результатів діаризації записів моделі NeMo.

Зважаючи на те, що для подальшої роботи нам потрібні будуть файли RTTM, ми:

- уніфікували назви файлів .wav та .TextGrid;
- перевірили усі .TextGrid файли на наявність рівня шуму;
- написали парсер для конвертації файлів TextGrid у файли RTTM. Парсер написаний мовою програмування Python. Основний функціонал коду використовує модуль TGT(TextGridTools);
- для кожного зі 151 TextGrid файлів ми отримали відповідний файл-результат формату RTTM.

РОЗДІЛ 3. ПРОВЕДЕННЯ ДІАРИЗАЦІЇ ЗАПИСІВ

У цьому розділі ми розглянемо можливості моделі *NeMo* від NVIDIA – моделі, яка має багато можливостей у сфері технології розпізнавання мовлення і мовців. Протестуємо модель з різними параметрами, визначимо найефективніші параметри для наших даних. Вихідними даними є попередньо розмічений акустичний корпус, що містить 151 аудіофайл у форматі .wav (загальна тривалість записів — 12 годин 08 хвилин) та RTTM файли, які ми згенерували у Розділі 2.

3.1. Технічні можливості моделі NeMo

Сучасний стан інформаційних технологій зумовлює великий вибір інструментів, за допомогою яких можна проводити автоматичну ідентифікацію, верифікацію та діаризацію. Для мови програмування Python розробляється багато модулів, які використовують машинне навчання та мають гарні результати при тестуванні.

Однак, ми зупинились на виборі моделі NeMo саме через наступні фактори:

- модель є безкоштовною у використанні;
- модель створена технологічним гігантом – компанією NVIDIA – це означає, що цей продукт є протестованим, надійним та стабільним, а також це означає, що продукт має свою інтернет-спільноту, в якої при потребі можна запитати поради або попросити про допомогу. На відміну від NeMo, існує багато проєктів не такого високого рівня розробки, які були створені, але не підтримуються уже тривалий час через відсутність фінансування тощо. Сюди відносяться проєкти (наприклад, *Unispeech* [68]), які були створені розробниками в

межах стажування на базі компаній-техногігантів (Microsoft). На перший погляд, такі інструменти виглядають надійними, але при користуванні ними виявляється, що версії пакетів конфліктують, певні частини код-бази виявляються застарілими тощо;

- для NeMo не є принциповою мова мовців, а це є гарною умовою, враховуючи, що російська та українська мови мають певний дефіцит даних та інструментів для їх обробки порівняно з англійською, китайською, іспанською мовами тощо;
- розробники NeMo надають гарні навчальні ресурси, які допомагають розібратись в широкому арсеналі можливостей моделі;
- модель легко та зручно застосувати в таких хмарних середовищах як Google Colab, за допомогою яких можна зекономити час обчислень.

Отже, для автоматичної діаризації записів ми будемо використовувати модель *NeMo*. *NVIDIA NeMo™* — це хмарна корпоративна структура повного циклу для розробників, яка дозволяє створювати, налаштовувати та розгортати генеративні моделі ШІ з мільярдами параметрів. Для завдань обробки мовлення ця структура пропонує сервіси з розпізнавання мовлення, класифікації мовлення та діаризації мовців [41].

Так виглядає схема процесу діаризації в моделі NeMo (*Рис. 3.1.1.)*:

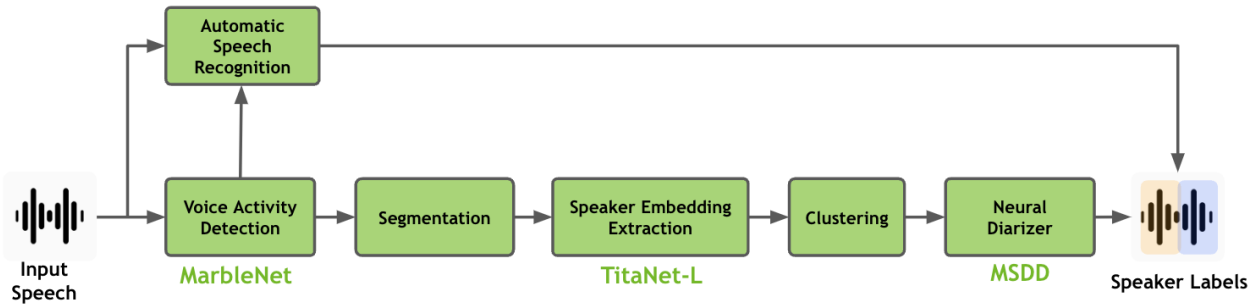


Рис. 3.1.1. Схема процесу діаризації в моделі NeMo [56]

Система діаризації мовців NeMo складається з наступних модулів:

- Voice Activity Detector (VAD) (укр. “Детектор голосової активності“) – модель, що піддається донавчанню та яка виявляє наявність або відсутність мовлення для генерування позначок часу для мовленнєвої активності з даного аудіозапису.
- Speaker Embedding Extractor (укр. “екстрактор векторів мовця”) – модель, що піддається донавчанню і яка витягує вектори мовця, що містять характеристики голосу, із необробленого аудіосигналу.
- Clustering Module (укр. “модуль кластеризації”) – модуль, що не піддається навчанню та групує вектори мовця у декілька кластерів.
- Neural Diarizer (укр.“модуль діаризації на основі нейронних мереж”) – модель, що піддається навчанню, яка оцінює мітки мовців за заданими функціями.

3.2 Використання моделі NEMO з різними параметрами

Обробка файлів моделлю NeMo відбувалась у Google Colab [19]. Google Colab (скорочено від "Colaboratory") — це хмарна платформа для виконання коду Python. Вона надає середовище Jupyter Notebook, де можна створювати, редагувати та виконувати код Python у веб-браузері без необхідності встановлювати та налаштовувати середовище на своєму комп'ютері. Безкоштовна версія платформи дає змогу ефективно обробляти дані. Оскільки обробка звукових WAV файлів займає значно більше часу, ніж обробка текстових файлів, було використано платну версію платформи – Colab Pro. Для порівняння, у безкоштовній версії в середньому один файл оброблявся десять хвилин, а у платній версії приблизно дві хвилини. Ще однією перевагою Colab Pro було те, що ця версія надає також збільшену оперативну пам'ять, що давало змогу працювати коду довгий час без переривань, на відміну від безкоштовної версії.

Вихідними даними для роботи з NeMo були пари файлів: звукозапис (.wav) та відповідна розмітка (.rttm) (*Рис. 3.2.1)*.

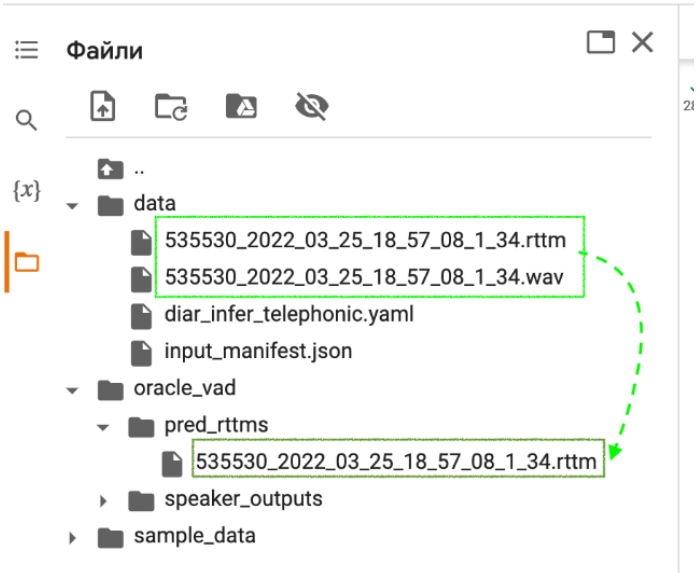


Рис. 3.2.1. Файловий менеджер у хмарному середовищі Google Colab, який містить вихідні файли та файл-результат

Так виглядає вміст вихідного RTTM файлу (*Рис. 3.2.2.*).

```
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 9.67187 0.8471499999999992 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 12.52775 0.5541200000000011 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 15.25575 1.5344999999999995 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 18.96413 0.9578199999999981 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 27.61826 0.8951200000000021 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 32.47751 0.7246299999999977 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 35.55387 1.295969999999997 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 40.28526 3.822589999999998 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 46.33801 0.608070000000005 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 49.95764 1.355470000000004 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 55.81723 2.71528 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 61.61421 2.6006399999999985 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 68.59915 0.6985000000000099 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 72.52514 1.6210400000000078 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 77.50829 0.7395599999999973 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 80.92541 2.216300000000004 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 94.04157 1.6747600000000062 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 97.57898 2.2015399999999943 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 127.5317 0.6208799999999997 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 129.76134 2.8804700000000025 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
SPEAKER 535530_2022_03_12_17_45_19_1_34 1 145.95426 1.5522100000000023 <NA> <NA> 535530_2022_03_12_17_45_19_1_34-0 <NA>
```

Рис. 3.2.2. Вміст вихідного RTTM файлу

У цьому RTTM файлі можемо спостерігати:

- перше поле завжди має значення “SPEAKER”;
- друге поле має значення ідентифікатора файлу;
- третє поле завжди має значення “1”
- четверте поле має значення початку уривка в секундах;
- п'яте поле має значення тривалості уривка в секундах;
- шосте поле має значення “NA”
- сьоме поле має значення “NA”
- восьме поле має значення за формулою *{ідентифікатор файлу}-{ідентифікатора спікера (згідно з анотаціями від лінгвістів-експертів)};*
- дев’яте поле має значення “NA”.

Варто зауважити, що значення полів “NA” є певною мірою формальними. Кількість полів буде нам принциповою при тестуванні результатів лише тому, що модуль, який ми будемо використовувати, вертає помилку, якщо кількість полів є меншою за *9*.

Можемо зробити висновок, що за потреби конвертації файлів TextGrid в файли RTTM, дійсно потрібно зберігати кількість полів, близьку до кількості полів в стандартні RTTM – 10 полів.

Експеримент проходив у кілька етапів. На кожному етапі ми підбирали різні параметри, щоб протестувати, які з них спрацюють найкраще для наших даних.

Для коригування результатів ми можемо змінювати параметри у файлі з конфігураціями. Конфігураційний файл – це *diar_infer_telephonic.yaml (Рис. 3.2.3)*. До прикладу, можна змінити виділені значення на протилежні (*False* замінити на *True* та навпаки) і таким чином ми отримаємо інакші результати діаризації. В цьому й полягає суть зміни параметрів моделі. Цей файл наповнили параметрами творці моделі, а ми у своєму коді його просто завантажуємо та коригуємо.

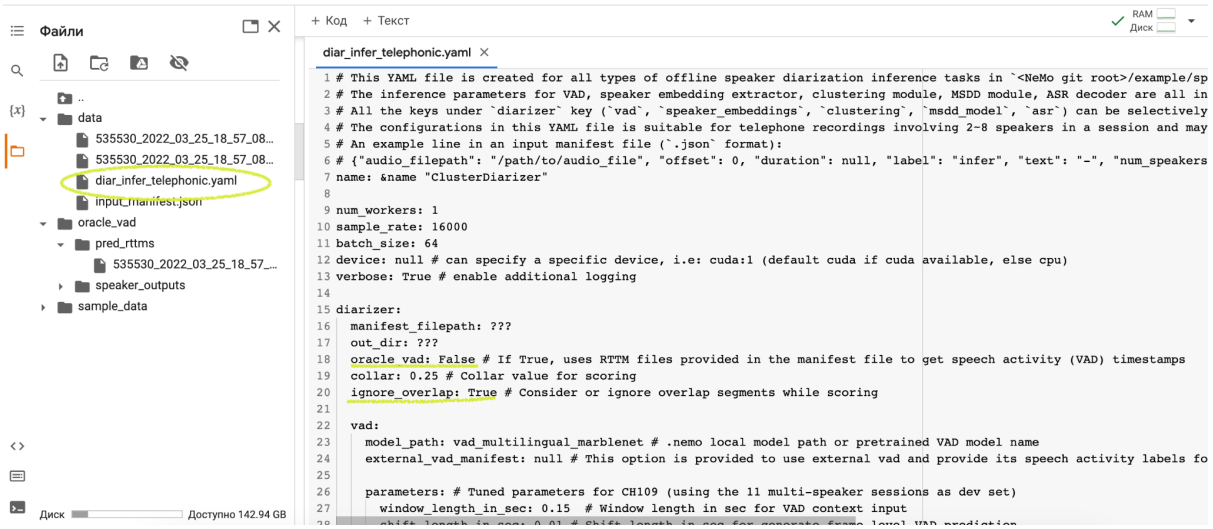


Рис. 3.2.3. Файл diar_infer_telephonic.yaml, який містить параметри, які модель NeMo використовує для своєї роботи. Саме ці значення ми можемо змінювати для коригування роботи моделі та тестування.

Параметри, які ми братимемо до уваги, щоб експериментувати, це:

1) стандартні параметри,
2) *oracle_vad*,
3) *ignore_overlap*,
4) *max_num_speakers*

Пояснимо значення параметрів.

1. *oracle_vad* – це параметр, який вмикає або вимикає роботу модуля VAD (*Voice Activity Detection, укр. "виявлення голосової активності"*). VAD модуль є результатом створення нейронної мережі MarbleNet, розробниками якої є теж NVIDIA [28]. NVIDIA позиціює цю модель як легку, тобто ту, що не вимагає багато обчислювальних ресурсів. Це означає, що технологія VAD підходить для використання на пристроях з обмеженими ресурсами.

   VAD використовує RTTM файли, які ми даємо як вхідні дані для моделі, щоб визначити уривки, де дійсно звучить голос людини. Модель має на меті ігнорувати шум та інші другорядні звуки, які не належать до мовлення людини. Якщо параметр *oracle_vad* має значення *True, то значить, що* модуль виявлення голосової активності працює.
2. *ignore_overlap* – це параметр, який ігнорує або бере до уваги перетин мовців в один момент часу. Наприклад, бувають ситуації, коли два чи три мовці говорять одночасно. Цей параметр вміє враховувати такі ситуації. Якщо параметр *ignore_overlap* має значення *True, то значить, що* перетини мовців не беруться до уваги.
3. *max_num_speakers – це максимальна кількість мовців для кожного запису.*

   Розглянемо етапи експерименту.

   Першим етапом було використання стандартних параметрів.

   Ті параметри, які нас цікавлять, мають таке значення в конфігураційному файлі:

   - *oracle_vad: True;*
   - *ignore_overlap: True;*
   - *max_num_speakers: 8.*

   Вихідними даними були аудіозаписи та RTTM файли, які ми

отримали як результат конвертації файлів TextGrid в формат RTTM.

На другому етапі ми змінили значення параметра *max_num_speakers з числа 8 на число 4*, адже за нашою статистикою, найбільшим числом мовців в одному записі є 4.

Вихідними даними були аудіозаписи та RTTM файли, які ми отримали як результат конвертації файлів TextGrid в формат RTTM.

На третьому етапі ми змінили значення параметра *oracle_vad* зі значення *True на значення False*, тобто вимкнули модуль виявлення голосової активності. Це означає, що на цьому етапі розмітка RTTM не береться до уваги моделі, а використовується суто звуковий запис у форматі WAV.

Вихідними даними були аудіозаписи та RTTM файли, які ми отримали як результат конвертації файлів TextGrid в формат RTTM.

На четвертому етапі ми змінили значення параметра *ignore_overlap* зі значення *True на значення False,* тобто прийняли рішення не ігнорувати перетини мовців в один момент часу, а брати їх до уваги.

Вихідними даними були аудіозаписи та RTTM файли, які ми отримали як результат конвертації файлів TextGrid в формат RTTM.

Підсумовуючи, з першого по четвертий етап ми використовували наш акустичний корпус з RTTM розміткою, але щоразу змінювали по одному параметру в конфігураційному файлі моделі NeMo.

Наступними етапами було використання розміток RTTM без шуму, але з такими ж параметрами. Оскільки початково файли TextGrid містили шум, то після конвертації в RTTM, вони цей шум зберегли. Для створення набору даних без шуму було створено програму, який видаляє “шумний” канал з файлів TextGrid. На практиці це означає видалити рівень з назвою “Noise” з файлу (*Рис. 3.2.4)* і конвертувати його в RTTM за допомогою парсера, який ми описували в Розділі 2:

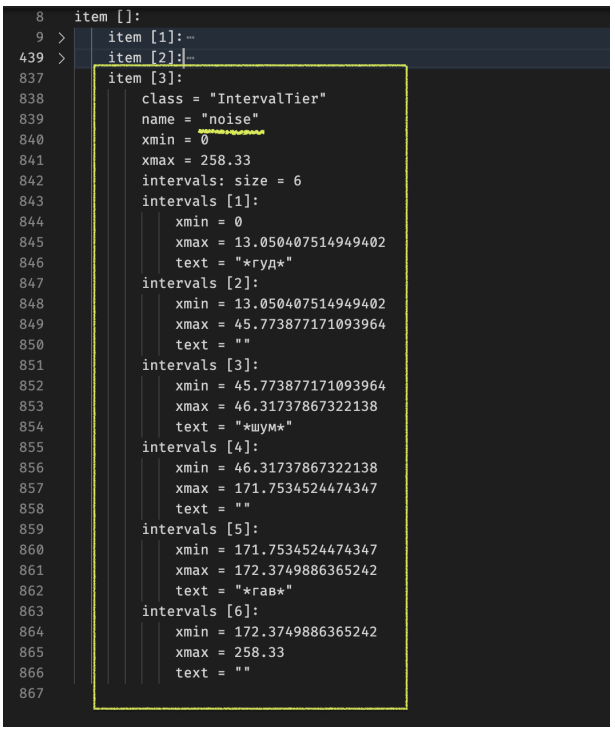


Рис. 3.2.4. Як рівень шуму (“noise”) виглядає в текстовому редакторі

На п'ятому етапі ми використали стандартні параметри.

Вихідними даними були аудіозаписи та RTTM файли *без шумного рівня*, які ми отримали як результат конвертації файлів TextGrid, очищених від шуму, у формат RTTM.

На шостому етапі ми змінили значення параметра *max_num_speakers з числа 8 на число 4*, адже за нашою статистикою, найбільшим числом мовців в одному записі є 4.

Вихідними даними були аудіозаписи та RTTM файли *без шумного рівня*, які ми отримали як результат конвертації файлів TextGrid, очищених від шуму, у формат RTTM.

На сьомому етапі ми змінили значення параметра *oracle_vad* зі значення *True на значення False*, тобто вимкнули модуль виявлення голосової активності. Це означає, що на цьому етапі розмітка RTTM не береться до уваги моделі, а використовується суто звуковий запис у форматі WAV.

Вихідними даними були аудіозаписи та RTTM файли *без шумного рівня*, які ми отримали як результат конвертації файлів TextGrid, очищених від шуму, у формат RTTM.

На восьмому етапі ми змінили значення параметра *ignore_overlap* зі значення *True на значення False,* тобто прийняли рішення не ігнорувати перетини мовців в один момент часу, а брати їх до уваги.

Вихідними даними були аудіозаписи та RTTM файли *без шумного рівня*, які ми отримали як результат конвертації файлів TextGrid, очищених від шуму, у формат RTTM.

В результаті використання моделі на кожному етапі ми отримали по одному RTTM файлу з однієї пари. RTTM файл-результат тепер містить поле з ідентифікованим мовцем (*Рис. 3.2.5*):

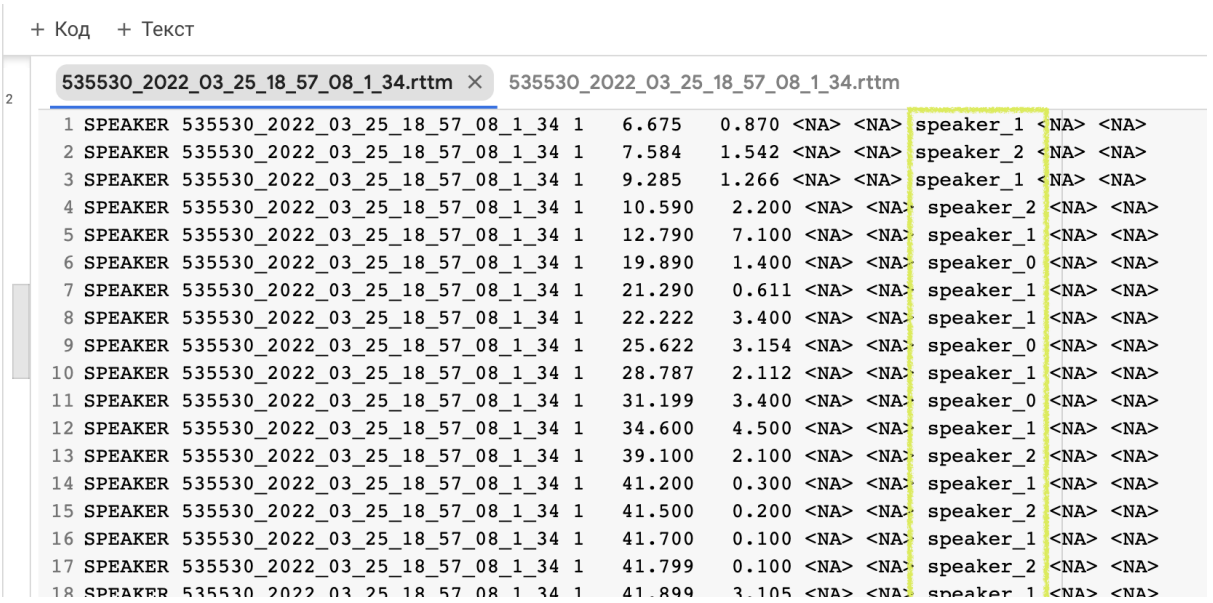


Рис. 3.2.5. RTTM-файл як результат роботи NeMo. Файл містить поле з ідентифікатором мовця

Програмне забезпечення з використанням моделі, знаходиться в Google Colab за посиланням (Див. Додаток 4).

3.3 Оцінка результатів

Оскільки ми отримали багато файлів-результатів в ході багатоетапного експерименту, їх оцінка потребуватиме багато зусиль та часу. З цього випливає рішення провести автоматичне тестування результатів.

Для оцінки результатів ми обрали Python бібліотеку *dscore* [22]. Ця бібліотека є зручною для використання, а також пропонує багато метрик, за якими можна провести оцінку результатів.

Для того, щоб провести тестування, ми повинні створити еталонний RTTM. Це означає, що ми будемо порівнювати результати, які видала модель NeMo, з тими, які ми вважаємо ідеальними. Ідеальними результатами в нашому випадку є розмічені вручну TextGrid файли, які ми конвертували в RTTM, та об'єднали в один великий RTTM файл.

При парсингу TextGrid файлів ми відкоригували код парсера відповідно до нової задачі – для того, щоб в межах великого еталонного файлу ідентифікатори мовців залишались унікальними, ми додали поле за формулою *{файл id}-{мовець id} (Рис. 3.3.1)*. Таким чином, бібліотека dscore буде використовувати це поле як на поле з ідентифікаторами мовців.

Наш еталонний RTTM містить усю розмітку з кожного 151 RTTM файлу.

gold_standart.rttm

```
1414  SPEAKER 535530_2022_03_14_13_12_25_1_34 1 96.15964 2.151130000000009 <NA> <NA> 535530_2022_03_14_13_12_25_1_34-0 <NA>
1415  SPEAKER 535530_2022_03_14_13_12_25_1_34 1 98.31077 1.9699799999999925 <NA> <NA> 535530_2022_03_14_13_12_25_1_34-1 <NA>
1416  SPEAKER 535530_2022_03_14_13_23_39_1_34 1 104.80244 3.187259999999995 <NA> <NA> 535530_2022_03_14_13_23_39_1_34-0 <NA>
1417  SPEAKER 535530_2022_03_14_13_23_39_1_34 1 11.67944 0.5795000000000012 <NA> <NA> 535530_2022_03_14_13_23_39_1_34-0 <NA>
1418  SPEAKER 535530_2022_03_14_13_23_39_1_34 1 110.03606 0.8511400000000009 <NA> <NA> 535530_2022_03_14_13_23_39_1_34-1 <NA>
1419  SPEAKER 535530_2022_03_14_13_23_39_1_34 1 110.50161 7.4429700000000025 <NA> <NA> 535530_2022_03_14_13_23_39_1_34-0 <NA>
1420  SPEAKER 535530_2022_03_14_13_23_39_1_34 1 118.90438 5.320499999999996 <NA> <NA> 535530_2022_03_14_13_23_39_1_34-1 <NA>
1421  SPEAKER 535530_2022_03_14_13_23_39_1_34 1 12.45814 1.1227900000000002 <NA> <NA> 535530_2022_03_14_13_23_39_1_34-1 <NA>
1422  SPEAKER 535530_2022_03_14_13_23_39_1_34 1 125.14397 0.525180000000006 <NA> <NA> 535530_2022_03_14_13_23_39_1_34-0 <NA>
1423  SPEAKER 535530_2022_03_14_13_23_39_1_34 1 128.29501 0.5432900000000132 <NA> <NA> 535530_2022_03_14_13_23_39_1_34-0 <NA>
1424  SPEAKER 535530_2022_03_14_13_23_39_1_34 1 133.52497 0.6519399999999962 <NA> <NA> 535530_2022_03_14_13_23_39_1_34-0 <NA>
```

Рис. 3.3.1. Еталонний файл RTTM, який містить вміст усіх RTTM файлів та містить унікальні ідентифікатори мовців

Використовуючи бібліотеку dscore, запускаючи скрипт для порівняння еталонного файлу з результатами кожного етапу, ми отримали такі результати (Див. Додаток 5):

Результати **першого етапу** (*Рис. 3.3.2)*:

Scoring...

File	DER	JER	B3-Precision	B3-Recall	B3-F1	GKT(ref, sys)	GKT(sys, ref)	H(ref|sys)	H(sys|ref)	MI	NMI
*** OVERALL ***	22.55	28.06	0.78	0.76	0.77	0.76	0.78	0.64	0.63	7.37	0.92

Рис. 3.3.2. Результати першого етапу

Результати **другого етапу** є такими ж, як результати першого.

Результати **третього етапу (***Рис. 3.3.3)*:

Scoring...

File	DER	JER	B3-Precision	B3-Recall	B3-F1	GKT(ref, sys)	GKT(sys, ref)	H(ref|sys)	H(sys|ref)	MI	NMI
535530_2022_03_12_17_45_19_1_34	16.20	19.47	0.74	0.82	0.78	0.73	0.63	0.84	0.53	1.25	0.65
535530_2022_03_12_18_01_37_1_34	39.32	35.20	0.75	0.58	0.66	0.43	0.61	0.71	1.08	0.90	0.50
926315_2022_03_24_14_52_02_2_110	23.76	25.35	0.69	0.72	0.70	0.54	0.49	0.89	0.74	0.72	0.47
926315_2022_03_24_15_29_22_2_110	26.23	30.39	0.67	0.65	0.66	0.42	0.42	0.96	0.93	0.50	0.35
926315_2022_03_26_11_55_10_2_110	25.49	27.64	0.63	0.70	0.66	0.48	0.42	1.05	0.76	0.65	0.42
926315_2022_03_26_11_56_37_2_110	26.47	28.44	0.72	0.63	0.67	0.43	0.52	0.82	0.91	0.62	0.42
*** OVERALL ***	29.98	37.70	0.68	0.64	0.66	0.64	0.68	0.90	0.91	7.08	0.89

Рис. 3.3.3. Результати третього етапу

Результати **четвертого етапу** (*Рис. 3.3.4*):

Scoring...

File	DER	JER	B3-Precision	B3-Recall	B3-F1	GKT(ref, sys)	GKT(sys, ref)	H(ref|sys)	H(sys|ref)	MI	NMI
*** OVERALL ***	21.72	28.15	0.78	0.77	0.77	0.77	0.78	0.65	0.60	7.33	0.92

Рис. 3.3.4. Результати четвертого етапу

Результати **п'ятого етапу** (*Рис. 3.3.5*):

File	DER	JER	B3-Precision	B3-Recall	B3-F1	GKT(ref, sys)	GKT(sys, ref)	H(ref|sys)	H(sys|ref)	MI	NMI
*** OVERALL ***	17.30	26.60	0.80	0.81	0.81	0.81	0.80	0.55	0.50	7.37	0.93

Рис. 3.3.5. Результати п'ятого етапу

Результати **шостого етапу** (*Рис. 3.3.6*):

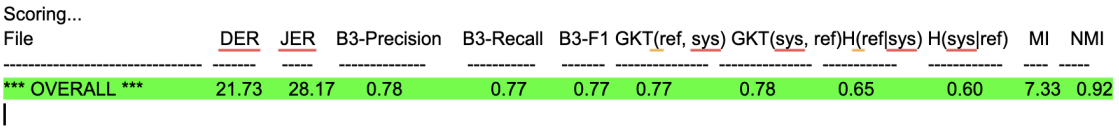


Рис. 3.3.6. Результати шостого етапу

Результати **сьомого етапу** (*Рис. 3.3.7*):

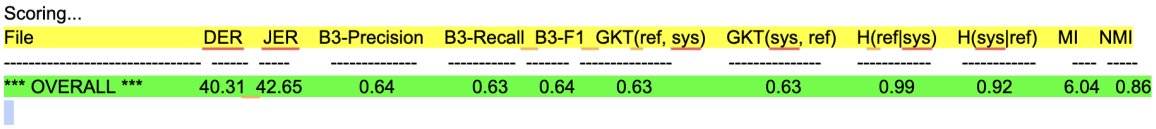


Рис. 3.3.7. Результати сьомого етапу

Результати **восьмого етапу** (*Рис. 3.3.8*):

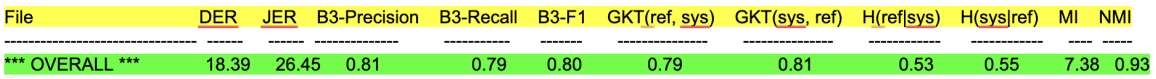


Рис. 3.3.8. Результати восьмого етапу

Отже, за результатами тестування можемо укласти таблицю (Табл. 3.3.1), яка містить параметри, опис даних та показники тестувань. Ми будемо оцінювати результати за метрикою *Diarization Error Rate* (DER) (*укр. "коефіцієнт помилок діаризації"*) (*Рис. 3.3.9)*. Коефіцієнт помилок діаризації (DER) є найбільш часто використовуваним показником для діаризації мовців.

$$DER = \frac{\text{False Alarm} + \text{Miss} + \text{Overlap} + \text{Confusion}}{\text{Reference Length}}$$

Рис. 3.3.9. Формула, за якою вираховується коефіцієнт помилок діаризації (DER) [7]

Визначення змінних [7]:

- *Reference Length* – загальна довжина запису в еталонному файлі.

- *False Alarm* – тривалість сегментів, які розглядаються як мовлення в тестованому файлі, але не в еталонному файлі.
- *Miss* – тривалість сегментів, які розглядаються як мовлення в еталонному файлі, але не в тестованому файлі.
- *Overlap* – тривалість сегментів, які розглядаються як перетин мовців в тестованому файлі, але не в еталонному файлі.
- *Confusion* – тривалість сегментів, які призначаються різним мовцям у тестованому файлі та еталонному файлі.

Тривалість вимірюється секундами.

Параметри моделі NeMo	**% DER, файли з шумом**	**% DER, файли *БЕЗ* шуму**
стандартні параметри (oracle_vad: True, ignore_overlap: True, max_num_speakers: 8)	22.55	17.30
max_num_speakers: 4	21.73	17.32
oracle_vad: False	29.98	40.31
ignore_overlap: False	21.72	18.39

Таблиця 3.3.1.Результати тестування моделі NEMO за метрикою DER

Чим менший відсоток DER, тим краще модель впоралась із застосованими параметрами.

Якщо порівнювати результати за стандартними параметрами, то бачимо, що коефіцієнт помилок діаризації в файлах без шуму є на 5.25% нижчим, ніж у файлах з шумом. З цього випливає теза, що, попри роботу

модуля VAD, дійсно є сенс очищати шум у файлах (Див. Розділ 3.2) перед діаризацією.

Застосування параметра *max_num_speakers: 4 додало* певну точність діризації файлів з шумом, але при цьому підвищило DER для файлів без шуму.

Застосування параметра *oracle_vad: False* підвищило коефіцієнт помилки до майже 30% у файлах з шумом і до 40% у файлах без шуму. З цього можемо зробити висновок, що увімкнений VAD дає дійсно хороші результати, а вимкнений VAD суттєво (в 1.5-2 рази) підвищує коефіцієнт помилки.

Використання параметра *ignore_overlap: False* дещо пішло на користь точності діаризації для файлів з шумом, зменшивши DER на майже 1%, тоді як для файлів без шуму DER підвищився на 1%.

ВИСНОВКИ ДО РОЗДІЛУ 3

У Розділі 3 ми описали технічні можливості моделі NeMo. Ми також обґрунтували переваги цієї моделі та пояснили, чому вибрали саме цю модель для проведення діаризації мовців:

- модель NeMo є безкоштовною;
- модель NeMo є зручною у використанні;
- модель NeMo є надійною та підтримуваною в довгостроковій перспективі;
- модель NeMo підтримує будь-яку мову;
- модель NeMo має гарні навчальні матеріали.

Ми створили програмне забезпечення, яке використовує модель NeMo для діаризації зі стандартними та зміненими параметрами. Ми проводили експеримент у 8 етапів, змінюючи вхідні параметри та дані. Нашими вхідними даними були RTTM файли з шумом та без шуму. В ході експерименту ми отримали по 8 варіантів RTTM розмітки для кожного WAV файлу.

Для оцінки результатів роботи моделі NeMo ми обрали Python бібліотеку *dscore*. Ця бібліотека є зручною для використання, а також пропонує багато метрик, за якими можна провести оцінку результатів. Для тестування ми створили еталонний RTTM файл, аби порівнювати з ним кожен результат.

Підсумуємо ефективність зміни параметрів:

- стандартні параметри моделі NeMo показали доволі гарні результати – коефіцієнт помилки діаризації становить від 17.3% до 22.5%, і кращий результат ми отримали на даних з очищеним шумом;
- для файлів з очищеним шумом варто застосувати стандартні параметри, адже вони дали найкращий результат;

- параметри *max_num_speakers* та *ignore_overlap: False* дозволяють знизити коефіцієнт помилок діаризації для файлів з шумом на декілька відсотків;
- модуль VAD чудово впорався із завданнями виявлення голосової активності, адже його робота відчутно знижує коефіцієнт помилки.

ВИСНОВКИ

Технологія розпізнавання мовців знайшла безліч сфер застосування і безумовно має великі перспективи у майбутньому. Ця технологія є розвиненою, проте не має достатніх інструментів та досліджень для усіх задач, які постають перед людством.

Після вторгнення Російської Федерації на територію Україну є потреба автоматично обробити усі аудіозаписи-перехоплення, зроблені в ході війни, щоб ідентифікувати мовців, які з високою ймовірністю є воєнними злочинцями. Проблема розпізнавання мовців є малодослідженою для російської та української мов, порівнюючи з англійською та китайською мовами тощо. В ході роботи ми зробили огляд наукових праць, щоб описати класифікацію систем розпізнавання мовців. Ми дали визначення таких понять: *розпізнавання мовців, ідентифікація, верифікація, діаризація.* Було описано історію технології автоматичного розпізнавання мовців від перших спроб її реалізації до сьогодення та окреслили роль машинного навчання у сучасних системах автоматичного розпізнавання мовців. Попри те, що розпізнавання мовців на основі глибокого навчання досягло великого успіху, багато проблем ще належить вирішити. Для навчання нейронної мережі більшість методів виділення рис мовця потребують вхідних даних ручної роботи, що може бути дорогим та займати багато часу. Сучасні моделі глибокого навчання мають велику кількість параметрів, які важко застосувати до портативних пристроїв. Навчання мережі також потребує великої кількості анотованих навчальних даних і значних обчислювальних ресурсів.

Діаризація грає важливу роль у системах розпізнавання мовців, оскільки дозволяє відокремити голоси різних учасників розмови та визначити, коли кожен мовець говорить. На основі попередньо створеного

акустичного корпусу з анотаціями у форматі TextGrid, що містить *151 аудіофайл у форматі .wav* (загальна тривалість записів — 12 годин 08 хвилин), ми підготували дані для подальшої діаризації. RTTM файли були згенеровані шляхом конвертації файлів TextGrid за допомогою парсера, який ми власноруч створили. Тож було описано корпус усного мовлення для подальшого його використання у створенні програмного забезпечення для діаризації, а також створено парсер-конвертатор анотаційних файлів з формату TextGrid у формат RTTM.

Ми обрали для нашої роботи, проаналізували та описали сучасну систему діаризації аудіозаписів NeMo від NVIDIA – визначили модулі системи та окреслили функціонал кожного модуля. За допомогою цієї моделі ми провели діаризацію, експериментуючи з вхідними даними та параметрами моделі. Після діаризації ми провели тестування результатів: обрали Python бібліотеку для тестування, створили еталонний файл RTTM, провели тестування та зібрали показники DER (коефіцієнт помилки діаризації), визначили найкращі параметри.

В результаті експерименту можна визначити такі тези, спираючись на статистичні показники DER:

- стандартні параметри моделі NeMo показали доволі гарні результати – коефіцієнт помилки діаризації становить від 17.3% до 22.5%, і кращий результат ми отримали на даних з очищеним шумом;
- для файлів з очищеним шумом варто застосувати стандартні параметри, адже вони дали найкращий результат;
- параметри *max_num_speakers* та *ignore_overlap: False* дозволяють знизити коефіцієнт помилок діаризації для файлів з шумом на декілька відсотків;

- модуль VAD чудово впорався із завданнями виявлення голосової активності, адже його робота відчутно знижує коефіцієнт помилки.

У майбутньому варто провести донавчання модулів NeMo для підвищення точності діаризації та потестувати результати, враховуючи усі можливі метрики.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Головне управління розвідки МО України [Електронний ресурс] – URL: https://t.me/DIUkraine.
2. Зарецька Д. Аудіо-аутентифікація користувача за голосом / Д. Зарецька, М. Баранов, С. Іванов. // Всн. Львів. ун-ту. Сер. прикл. матем. та інф.. – 2022. – №30.
3. Корнієнко О. Метод відображення мовних сигналів у задачі розпізнавання мовця / О. Корнієнко // Технічні науки та технології. - 2017. - № 3. - С. 129-137. - Режим доступу: http://nbuv.gov.ua/UJRN/Vcndtn_2017_3_17
4. Перегляд можливостей Siri на iPhone [Електронний ресурс] // Apple – URL: https://support.apple.com/uk-ua/guide/iphone/ipha48873ed6/ios.
5. Пустовіт, Д. Т. Приховані Марківські ланцюги в задачах розпізнавання кризових явищ : магістерська дис. : 124 Системний аналіз / Пустовіт Дмитро Тарасович. – Київ, 2018. – 98 с.
6. Що таке голосова біометрія [Електронний ресурс] – URL: https://privatbank.ua/voice-biometrics.
7. A lightweight library to compute Diarization Error Rate (DER). [Електронний ресурс] – URL: https://pypi.org/project/simpleder/.
8. A Review of Speaker Diarization: Recent Advances with Deep Learning [Електронний ресурс] / [J. P. Te, K. Naoyuki, D. Dimitriadisb та ін.]. – 2021. – URL: https://arxiv.org/pdf/2101.09624.
9. A. Q. Ohi, M. F. Mridha, M. A. Hamid and M. M. Monowar, "Deep Speaker Recognition: Process, Progress, and Challenges," in IEEE Access, vol. 9, pp. 89619-89643, 2021, doi: 10.1109/ACCESS.2021.3090109.

10. Alameda-Pineda, X., Ricci, E., & Sebe, N. (2019). Multimodal behavior analysis in the wild: An introduction. Multimodal Behavior Analysis in the Wild
11. Ali Bou Nassif, Ismail Shahin, Ashraf Elnagar, Divya Velayudhan, Adi Alhudhaif, Kemal Polat, Emotional speaker identification using a novel capsule nets model, Expert Systems with Applications, Volume 193, 2022, 116469, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2021.116469.
12. Alluhaidan, A.S.; Saidani, O.; Jahangir, R.; Nauman, M.A.; Neffati, O.S. Speech Emotion Recognition through Hybrid Features and Convolutional Neural Network. Appl. Sci. 2023, 13, 4750. https://doi.org/10.3390/app13084750
13. Andrea T. G. An Access Control System with Speaker Verification [Електронний ресурс] / Tiziano Galizia Andrea. – 2019. – URL: https://www.hackster.io/at-galizia/an-access-control-system-with-speaker-verification-b401d0.
14. API — TextGridTools 1.4.4 documentation [Електронний ресурс] – URL: https://textgridtools.readthedocs.io/en/stable/api.html.
15. Automatic Speaker Recognition for Authenticating Users in the Internet of Things [Електронний ресурс]. – 2016. – URL: https://www.sri.com/blog-archive/automatic-speaker-recognition-for-authenticating-users-in-the-internet-of-things/.
16. Boersma P. Praat: doing phonetics by computer [Електронний ресурс] / P. Boersma, D. Weenink – URL: https://www.fon.hum.uva.nl/praat/.
17. Chaudhuri, Sourish, Joseph Roth, Daniel PW Ellis, Andrew Gallagher, Liat Kaver, Radhika Marvin, Caroline Pantofaru et al. "Ava-speech: A densely labeled dataset of speech activity in movies." arXiv preprint arXiv:1808.00606 (2018).

18. Cognitive Computing: Theory and Applications 1st Edition - September 1, 2016, Authors: Vijay Raghavan, Venkat Gudivada, Venu Govindaraju, C.R. Rao
19. Colab [Електронний ресурс] – URL: https://colab.research.google.com/#scrollTo=SKQ4bH7qMGrA.
20. Daqrouq K. Wavelet entropy and neural network for text-independent speaker identification, Engineering Applications of Artificial Intelligence / Khaled Daqrouq. // ScienceDirect. – 2011. – №24. – С. 796–802.
21. Device for speaker's verification [Електронний ресурс] // 1988 – URL: https://patents.google.com/patent/US4752958/en.
22. Diarization scoring tools. [Електронний ресурс] // 2019 – URL: https://github.com/nryant/dscore.
23. diarize [Електронний ресурс] // Merriam-Webster – URL: merriam-webster.com/dictionary/diarize.
24. Dictionaries [Електронний ресурс] – URL: https://docs.python.org/3/tutorial/datastructures.html#dictionaries.
25. DRĂGHICESCU1 D. Forensic application of speaker identification [Електронний ресурс] / Dragoş DRĂGHICESCU1 // U.P.B. Sci. Bull. – 2015. – URL: https://www.scientificbulletin.upb.ro/rev_docs_arhiva/full861_950357.pdf.
26. Emotional speaker identification using a novel capsule nets model [Електронний ресурс] / [B. N. Ali, I. Shanin, V. Divya та ін.] // Expert Systems with Applications. – 2022. – URL: https://www.sciencedirect.com/science/article/abs/pii/S0957417421017498.
27. Falabella A. Speaker Verification: Architecture and Results (Part 2) [Електронний ресурс] / Antonio Falabella. – 2022. – URL:

https://medium.com/data-reply-it-datatech/speaker-verification-architecture-and-results-part-2-8f9997cdf323.

28. Fei J. Marblenet: Deep 1d Time-channel Separable Convolutional Neural Network For Voice Activity Detection [Електронний ресурс] / J. Fei, S. Majumdar, B. Ginsburg. – 2020. – URL: https://arxiv.org/abs/2010.13886.
29. Forensic Voice Comparison: The Essential Guide [Електронний ресурс] – URL: https://www.phonexia.com/knowledge-base/forensic-voice-comparison-essential-guide/.
30. Furui, S. (1996). An Overview of Speaker Recognition Technology. In: Lee, CH., Soong, F.K., Paliwal, K.K. (eds) Automatic Speech and Speaker Recognition. The Kluwer International Series in Engineering and Computer Science, vol 355. Springer, Boston, MA
31. Furui, S. (1997) “Recent Advances in Speaker Recognition”, Proc. First Int. Conf. Audio- and Video-based Biometric Person Authentication, Crans-Montana, Switzerland, pp. 237-252.
32. Hamidreza B. K. End-to-end deep speaker embedding learning using multi-scale attentional fusion and graph neural networks [Електронний ресурс] / B. K. Hamidreza, J. Siyavash // Expert Systems with Application. – 2023. – URL: https://www.sciencedirect.com/science/article/abs/pii/S0957417423003342.
33. Introduction to Speaker Diarization - ICSI (Berkeley) [Електронний ресурс] – URL: https://www1.icsi.berkeley.edu/eecs225d/spr12/slides/diarization.pdf.
34. Jin, Minho & Yoo, Chang. (2009). Speaker Verification and Identification. Behavioral Biometrics for Human Identification: Intelligent Applications. 264-289. 10.4018/978-1-60566-725-6.ch013.

35. KWS15 KEYWORD SEARCH EVALUATION PLAN [Електронний ресурс]. – 2015. – URL: https://www.nist.gov/system/files/documents/itl/iad/mig/KWS15-evalplan-v05.pdf.
36. L. Boves and E. den Os, ‘‘Speaker recognition in telecom applications,’’ in Proc. IEEE 4th Workshop Interact. Voice Technol. Telecommun. Appl. (IVTTA), Sep. 1998, pp. 203–208.
37. Liberman M. "Voiceprint" springs eternal [Електронний ресурс] / Mark Liberman // Language and the law, Language and the media, Words words words. – 2018. – URL: https://languagelog.ldc.upenn.edu/nll/?p=36412.
38. M. M. Kabir, M. F. Mridha, J. Shin, I. Jahan and A. Q. Ohi, "A Survey of Speaker Recognition: Fundamental Theories, Recognition Methods and Opportunities," in IEEE Access, vol. 9, pp. 79236-79263, 2021, doi: 10.1109/ACCESS.2021.3084299.
39. M. McLaren, Y. Lei and L. Ferrer, "Advances in deep neural network approaches to speaker recognition," 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), South Brisbane, QLD, Australia, 2015, pp. 4814-4818, doi: 10.1109/ICASSP.2015.7178885.
40. Moore S. Voice Analysis in Forensics [Електронний ресурс] / Sarah Moore – URL: https://www.azolifesciences.com/article/Voice-Analysis-in-Forensics.aspx
41. Nemo Framework for Generative AI - Get Started | NVIDIA Developer [Електронний ресурс] – URL: https://developer.nvidia.com/nemo.
42. Noise reduction using Spectral Gating in python [Електронний ресурс] – URL: https://pypi.org/project/noisereduce/.
43. Nuance [Електронний ресурс] – URL: https://www.nuance.com/index.html.

44. Owen M. Microsoft buys Siri speech recognition partner Nuance in $19.7B deal [Електронний ресурс] / Malcolm Owen // AppleInsider. – 2021. – URL: https://appleinsider.com/articles/21/04/12/microsoft-buys-siri-speech-recognition-partner-nuance-in-197b-deal.
45. Pierre-Alexandre Broux, Florent Desnous, Anthony Larcher, Simon Petitrenaud, Jean Carrive, et al.. S4D: Speaker Diarization Toolkit in Python. Interspeech, Sep 2018, Hyderabad, India. ⟨hal-02280162⟩
46. Quain J. R. Alexa, What Happened to My Car? [Електронний ресурс] / John R. Quain // The New York Times Company. – 2018. – URL: https://www.nytimes.com/2018/01/25/business/amazon-alexa-car.html.
47. QUINTIN C. The Catalog of Carceral Surveillance: Voice Recognition and Surveillance [Електронний ресурс] / C. QUINTIN, B. LIPTON // DEEPLINKS BLOG. – 2021. – URL: https://www.eff.org/deeplinks/2021/09/catalog-carceral-surveillance-voice-recognition-and-surveillance.
48. R. Togneri and D. Pullella, "An Overview of Speaker Identification: Accuracy and Robustness Issues," in IEEE Circuits and Systems Magazine, vol. 11, no. 2, pp. 23-61, Secondquarter 2011, doi: 10.1109/MCAS.2011.941079.
49. Ramage D. Hidden Markov Models Fundamentals [Електронний ресурс] / Daniel Ramage// College of Information and Computer Sciences. – 2007. URL: https://people.cs.umass.edu/~mccallum/courses/inlp2004a/lect10-hmm2.pdf.
50. S. E. Tranter and D. A. Reynolds, "An overview of automatic speaker diarization systems," in IEEE Transactions on Audio, Speech, and Language Processing, vol. 14, no. 5, pp. 1557-1565, Sept. 2006, doi: 10.1109/TASL.2006.878256.

51. S. E. Tranter and D. A. Reynolds, "An overview of automatic speaker diarization systems," in IEEE Transactions on Audio, Speech, and Language Processing, vol. 14, no. 5, pp. 1557-1565, Sept. 2006, doi: 10.1109/TASL.2006.878256.]
52. Sabry F. Multimodal Contactless Biometric Systems / Fouad Sabry., 2022.
53. Saferstein R. Criminalistics: An Introduction to Forensic Science / Saferstein., 1995. – 599 с. – (6).
54. Shomron G. Who's Talking? Speaker Diarization and Emotion Recognition [Електронний ресурс] / Gil Shomron // Medium. – 2023. – URL: https://medium.com/@gil.shomron/whos-talking-speaker-diarization-and-emotion-recognition-in-radio-3e9623baeb2c.
55. Snyder D, Garcia-Romero D, Povey D, et al. Deep Neural Network Embeddings for TextIndependent Speaker Verification[C]//Interspeech. 2017: 999-1003.
56. Speaker Diarization [Електронний ресурс] – URL: https://docs.nvidia.com/deeplearning/nemo/user-guide/docs/en/stable/asr/speaker_diarization/intro.html.
57. Speaker Recognition and Verification [Електронний ресурс] – URL: https://speechprocessingbook.aalto.fi/Recognition/Speaker_Recognition_and_Verification.html.
58. Speaker Verification: Architecture and Results (Part 2) | by Antonio Falabella | Data Reply IT | DataTech | Medium "Speaker Verification: Architecture and Results (Part 2)." Medium, 2022, https://medium.com/data-reply-it-datatech/speaker-verification-architecture-and-results-part-2-8f9997cdf323. Accessed 1 05 2023
59. Speaker_Diarization_Inference.ipynb [Електронний ресурс] // NVIDIA – URL:

https://colab.research.google.com/github/NVIDIA/NeMo/blob/stable/tutorials/speaker_tasks/Speaker_Diarization_Inference.ipynb#scrollTo=fFl_mTPYVLCe.

60. Speech Recognition — GMM, HMM [Електронний ресурс] // Medium. – 2019. – URL: https://jonathan-hui.medium.com/speech-recognition-gmm-hmm-8bb5eff8b196.
61. Speech-Accent-Recognition [Електронний ресурс] // 2019 – URL: https://github.com/yatharthgarg/Speech-Accent-Recognition.
62. Sunitha, C. & Chandra, Evania. (2015). Speaker Recognition using MFCC and Improved Weighted Vector Quantization Algorithm. International Journal of Engineering and Technology. 7. 1685-1692.
63. T. Hasan, R. Saeidi, J. Hansen and D. van Leeuwen, "Duration mismatch compensation for I-vector based speaker recognition systems", Proc. Int. Conf. Acoust. Speech Signal Process., pp. 7663-7667, 2013.], [цит W. Chen, J. Huang and T. Bocklet, "Length- and noise-aware training techniques for short-utterance speaker recognition", Proc. Annu. Conf. Int. Speech Commun. Assoc., pp. 3835-3839, 2020.
64. TextGridTools Overview [Електронний ресурс] – URL: https://textgridtools.readthedocs.io/en/stable/overview.html.
65. The ACLEW DiViMe: An Easy-to-use Diarization Tool. [Електронний ресурс] / [A. Le Franc, E. Riebling, J. Karadayi та ін.] – URL: https://www.cs.cmu.edu/~fmetze/interACT/Publications_files/publications/aclew-divime-easy.pdf.
66. The inference parameters for VAD, speaker embedding extractor, clustering module, MSDD module, ASR decoder are all included in this YAML file. [Електронний ресурс] – URL: https://raw.githubusercontent.com/NVIDIA/NeMo/main/examples/speaker_tasks/diarization/conf/inference/diar_infer_telephonic.yaml.

67. Types of Speech Recognition [Електронний ресурс] – URL: https://www.lumenvox.com/espanol/resources/tips/types-of-speech-recognition.aspx.
68. UniSpeech - Large Scale Self-Supervised Learning for Speech [Електронний ресурс] – URL: https://github.com/microsoft/UniSpeech.
69. Wiebke Toussaint Hutiri and Aaron Yi Ding. 2022. Bias in Automated Speaker Recognition. In 2022 ACM Conference on Fairness, Accountability, and Transparency (FAccT '22). Association for Computing Machinery, New York, NY, USA, 230–247. https://doi.org/10.1145/3531146.3533089].
70. Xavier Anguera, Simon Bozonnet, Nicholas Evans, Corinne Fredouille, Gerald Friedland, et al. Speaker diarization: A review of recent research. IEEE transactions on acoustics, speech, and signal processing, 2010, pp. 1. hal-00733397
71. Y. Tu, W. Lin and M. -W. Mak, "A Survey on Text-Dependent and Text-Independent Speaker Verification," in IEEE Access, vol. 10, pp. 99038-99049, 2022, doi: 10.1109/ACCESS.2022.3206541.
72. Ye, F.; Yang, J. A Deep Neural Network Model for Speaker Identification. Appl. Sci. 2021, 11, 3603. https://doi.org/10.3390/app11083603
73. Z. Bai and X.-L. Zhang, ‘‘Speaker recognition based on deep learning: An overview,’’ Neural Netw., vol. 140, pp. 65–99, Aug. 2022
74. Voice Antispoofing: Origin, Types and Preventive Techniques [Електронний ресурс] – URL: https://antispoofing.org/voice-antispoofing-origin-types-and-preventive-techniques/.

ДОДАТКИ

Додаток 1

Скрипт, який автоматично перевіряє кожен .TextGrid файл з корпусу на наявність шуму.

```
from textgrid import TextGrid

def remove_item_with_name(textgrid_file, item_name, output_file):
    tg = TextGrid()
    tg.read(textgrid_file)

    item_index_to_remove = None
    for i, item in enumerate(tg):
        if item.name.lower() == item_name:
            item_index_to_remove = i
            break

    if item_index_to_remove is None:
        print(f"No item named '{item_name}' found in the TextGrid file.")
        return

    # Remove the item by reconstructing the TextGrid object
    new_tg = TextGrid()
    for i, item in enumerate(tg):
        if i != item_index_to_remove:
            new_tg.append(item)
```

```
    # Save the modified TextGrid file
    new_tg.write(output_file)


#
remove_item_with_name('/Users/valeriia/stuff/STUDYING/YEAR_6/masters_
degree/paired_plain_data/535530_2022_03_12_18_01_37_1_34.TextGrid',
#                         'noise',
#
'535530_2022_03_12_18_01_37_1_34_WITHOUT_NOISE.TextGrid')


import os

def process_textgrid_files(folder_path, item_name):
    # Iterate over all files in the folder
    for filename in os.listdir(folder_path):
        if filename.endswith(".TextGrid"):
            # Create the full file path
            file_path = os.path.join(folder_path, filename)

            # Generate the output file path

            # Remove the item with the specified name from the TextGrid file
            remove_item_with_name(file_path, item_name, filename)

# Example usage
```

```
folder_path =
'/Users/valeriia/stuff/STUDYING/YEAR_6/masters_degree/TG_without_NOISE/'
item_name = 'noise'
process_textgrid_files(folder_path, item_name)
```

Додаток 2

Посилання на метапозначки анотацій з акустичного корпусу.

https://docs.google.com/document/d/1DFoWucQ0oo8IM4JfxqMHSsDGt4MKJHHrUIqviNplpjc/edit?usp=sharing

Додаток 3

Скрипт, який конвертує TextGrid файл в RTTM.

```
import argparse
import tgt

def textgrid2rttm(textgrid):
    rttm_out = dict()
    tg = tgt.read_textgrid(textgrid, encoding="utf-8")

    for spkr in tg.get_tier_names():
        speaker_id = tg.get_tier_names().index(spkr)
        spkr_timestamps = []

        for _interval in tg.get_tiers_by_name(spkr):
            for interval in _interval:

                start, end, phrase = interval.start_time,\
                        interval.end_time,\
                        interval.text

                spkr_timestamps.append((start, end-start, phrase, speaker_id))

        rttm_out[spkr] = spkr_timestamps
    return rttm_out



def write_rttm(rttm_out, full_file_name):
    with open(full_file_name + '.rttm', 'w') as fout:
```

```
        for spkr in rttm_out:
            for start, dur, phrase, speaker_id in rttm_out[spkr]:
                fout.write(u'SPEAKER {} 1 {} {} <NA> <NA> {} <NA>\n'
                           .format(
                    full_file_name.split('/')[-1],
                    start,
                    dur,
                    full_file_name + '-' + str(speaker_id)
                    ))


if __name__ == '__main__':
    command_example = "python textgrid2rttm.py /folder/"
    parser = argparse.ArgumentParser(epilog=command_example)
    parser.add_argument('input_file',
                        help='' Input File '')
    parser.add_argument('output_file',
                        help=''Name of the output file in which to write'')

    args = parser.parse_args()

    rttm_out = textgrid2rttm(args.input_file)
    write_rttm(rttm_out, args.output_file)
```

Додаток 4

Посилання на середовище Colab з програмним забезпеченням, яке використовує модель NeMo з заданими параметрами.

https://colab.research.google.com/drive/1VnaEasVq5dQlx2DsJpeadVUB5pGY_F7M?usp=sharing

Додаток 5

Посилання на показники тестування результатів моделі NeMo.

https://drive.google.com/drive/folders/1AL4tdZxwoWLG0RKY8jeYRGVWrr0TWIgU?usp=sharing