МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет

імені Петра Могили

Факультет комп’ютерних наук

Кафедра комп'ютерної інженерії

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри,
д-р техн. наук, проф.

_____ І. М. Журавська

« __ » _____ 2024 р.

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

Система розпізнавання мовлення на базі Raspberry Pi та Whisper AI

Спеціальність 123 Комп’ютерна інженерія

123 – КБР.ПЗ.00 – 405.2010606

Студент _____ М. Ю. Жгарьов
підпис
«__» _____ 202__ р.

Керівник ст. викладач _____ Є. С. Дарнапук
підпис
«__» _____ 202__ р.

Миколаїв – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра комп'ютерної інженерії

ЗАТВЕРДЖУЮ

Зав. кафедри_____І.М Журавська

«_____»_____20___р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

Видано студенту групи 405 факультету комп'ютерних інженерії

Жгарьов Максим Юрійович

(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи
Система розпізнавання мовлення на базі Raspberry Pi та Whisper AI

Затверджена наказом по ЧНУ від «30»січня 2024р. № 17

2. Строк представлення кваліфікаційної роботи «_» червня 2024р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні
Розроблений прототип програмного комплексу на основі Whisper AI для розпізнавання мови.

4. Перелік питань, що підлягають розробці
Аналіз інформації стосовно технологій по навігації роботів в приміщені, особливості вибору, сфери застосування, переваги та недоліки. Порівняльний аналіз існуючих технічних рішень. Підбір необхідних для реалізації системи компонентів. Реалізація програмної частини апаратно-програмного комплексу

5. Перелік графічних матеріалів

Зображення компонентів

Схема архітектури Whisper AI

Таблиці характеристик компонентів

6. Завдання до спеціальної частини

перевірка освітленості у львівському офісі EPAM UKRAINE

7. Консультанти:

Консультант	Кафедра (організація)	Частина роботи
Макарова Олена Валеріївна	Старший викладач медичного інституту ЧНУ імені Петра Могили	Спеціальна частина з охорони праці

Керівник роботи старший викладач кафедри Дарнапук Євген Сергійович

(посада, прізвище, ім'я, по батькові)

_____

(підпис)

Завдання прийнято до виконання

Жгарьов Максим Юрійович

(прізвище, ім'я, по батькові студента)

_____

(підпис

Дата видачі завдання «31» січня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної роботи

Тема: Система розпізнавання мовлення на базі Raspberry Pi та Whisper AI_____
_____

№	Найменування роботи	Початок	Закінчення	Примітки
1	Розробка та затвердження завдання на виконання КР	01.02.24	03.02.24	Виконав
2	Огляд літератури за темою роботи	04.02.24	18.02.24	Виконав
3	Складання календарного плану БКР	19.02.24	24.02.24	Виконав
4	Аналіз предметної області	25.02.24	09.03.24	Виконав
5	Розробка проєктних рішень	10.03.24	20.04.24	Виконав
6	Перевірка працездатності, тестування та апробація розробленого АПЗ, аналіз результатів тестування	21.04.24	20.05.24	Виконав
7	Відгук керівника КР	13.06.24	13.06.24	Виконав
8	Оформлення БКР та презентації	15.05.24	04.06.24	Виконав
9	Попередній захист	28.05.24	28.05.24	Виконав
10	Рецензування	13.06.24	13.06.24	Виконав
11	Завершення оформлення КР та презентації	05.06.24	08.06.24	Виконав
12	Захист бакалаврської кваліфікаційної роботи			Виконав

Розробив здобувач ВО ___ Жгарьов Максим Юрійович _
(прізвище, ім'я, по батькові)

«__25__» ____травня_____ 2024_ р.
(підпис)

Керівник роботи _старший викладач кафедри Дарнапук Євген Сергійович__
(посада, прізвище, ім'я, по батькові)

«____» _____ 20__ р.

(підпис)

АНОТАЦІЯ

до кваліфікаційної бакалаврської роботи

«Система розпізнавання мовлення на базі Raspberry Pi та Whisper AI»

Студент 405гр.: Жгарьов Максим Юрійович

Керівник: ст. викладач Дарнапук Євген Сергійович

Ця кваліфікаційна робота присвячена розробці системи розпізнавання мови на базі Raspberry Pi. В роботі проведено огляд існуючих рішень, таких як Amazon Alexa та Google Assistant, аналіз їхніх апаратних вимог та патентів, що стосуються розпізнавання мови. Розглянуто апаратну складову системи, включаючи мікрофон, дисплей та Raspberry Pi. У програмній частині акцент зроблено на використанні Raspbian (Linux), Python та бібліотеки для розпізнавання мови Whisper AI. В ході розробки створено код для обробки та розпізнавання аудіофайлів, а також розроблено інтерфейс користувача. Робота завершується тестуванням та пакуванням готового додатку.

Актуальність роботи зумовлена швидким розвитком технологій розпізнавання мови та їх широким впровадженням у різних сферах діяльності. Використання недорогого обладнання, такого як Raspberry Pi, робить ці технології доступними для широкого кола користувачів.

Метою роботи є створення ефективної та доступної системи розпізнавання мови на базі Raspberry Pi, що може бути використана в домашніх умовах або в невеликих офісах.

Складові роботи включають огляд існуючих рішень, аналіз апаратних та програмних компонентів, розробку програмного забезпечення, тестування та пакування додатку.

Робота містить: 79 сторінок, 19 рисунків, 3 таблиць, 27 джерел посилання та 3 додатки.

Ключові слова: *розпізнавання мови, Raspberry Pi, Whisper AI, Raspbian, Python, інтерфейс користувача.*

ABSTRACT

of the Bachelor’s Thesis

"Speech Recognition System Based on Raspberry Pi and Whisper AI"

Student: Zhgaryov Maksym Yuriyovych

Supervisor: sr. lecturer Yevhen Darnapuk Serhiyovych

This qualification work is dedicated to the development of a speech recognition system based on Raspberry Pi. The work includes an overview of existing solutions such as Amazon Alexa and Google Assistant, an analysis of their hardware requirements, and patents related to speech recognition. The hardware components of the system, including the microphone, display, and Raspberry Pi, are considered. The software part focuses on the use of Raspbian (Linux), Python, and the Whisper AI speech recognition library. During the development process, code for processing and recognizing audio files was created, and a user interface was developed. The work concludes with testing and packaging of the finished application.

The relevance of the work is due to the rapid development of speech recognition technologies and their wide implementation in various fields of activity. The use of inexpensive equipment such as Raspberry Pi makes these technologies accessible to a wide range of users.

The aim of the work is to create an efficient and affordable speech recognition system based on Raspberry Pi that can be used at home or in small offices.

The components of the work include an overview of existing solutions, analysis of hardware and software components, software development, testing, and packaging of the application.

The work contains: 79 pages, 19 figures, 3 tables, 27 references, and 3 appendices.

Keywords*: speech recognition, Raspberry Pi, Whisper AI, Raspbian, Python, user interface.*

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ШІ – штучний інтелект

AI – Artificial Intelligence
API – Application Programming Interface
ARM – Advanced RISC Machines
BLE – Bluetooth Low Energy
GUI – Користувацький інтерфейс
IoT – Інтернет речей
OS – Operating System
RAM – Оперативна пам'ять
USB – Universal Serial Bus
Wi-Fi – Wireless Fidelity

ВСТУП

У цій роботі буде розглянуто основні принципи побудови системи розпізнавання мовлення на базі мікрокомп'ютера Raspberry Pi та нейронної мережі Whisper AI (Artificial Intelligence). Сучасні технології розпізнавання мовлення знайшли своє застосування в багатьох сферах, включаючи побутові пристрої, медичні системи, системи безпеки та автоматизації. Використання Raspberry Pi забезпечує доступність та гнучкість у розробці та налаштуванні подібних систем. [11]

Метою даної кваліфікаційної роботи є розробка та тестування системи розпізнавання мовлення, що працює на базі Raspberry Pi з використанням нейронної мережі Whisper AI. Система має бути здатна точно розпізнавати команди та перетворювати їх у текстовий формат.

Предмет дослідження включає компоненти мікрокомп'ютера Raspberry Pi, програмне забезпечення для розпізнавання мовлення, методи підключення та інтеграції мікрофонів, а також алгоритми та моделі нейронних мереж для обробки звукових сигналів.

Об'єктом дослідження в даній роботі є система розпізнавання мовлення на базі мікрокомп'ютера Raspberry Pi з використанням нейронної мережі Whisper AI. Зокрема, розглядаються:

– мікрокомп'ютер Raspberry Pi;

– нейронна мережа Whisper AI;

– програмне забезпечення для розпізнавання мовлення;

– методи тестування та оптимізації.

Завдання до кваліфікаційної бакалаврської роботи:

– ознайомлення з архітектурою Raspberry Pi та її можливостями для реалізації системи розпізнавання мовлення;

– вивчення основ роботи нейронної мережі Whisper AI та її застосування для розпізнавання мовлення;

– розробка схеми підключення мікрофону до Raspberry Pi;

– написання програмного забезпечення для обробки звукових сигналів та розпізнавання мовлення;

– тестування та оптимізація системи для забезпечення високої точності розпізнавання.

Загалом системи розпізнавання мовлення діляться на кілька основних категорій:

– локальні системи розпізнавання мовлення – ці системи виконують усі обчислення на місцевих пристроях без необхідності з'єднання з інтернетом. Переваги таких систем включають високу швидкість обробки даних та захист конфіденційності користувачів. Недоліки можуть включати обмеженість обчислювальних ресурсів, що впливає на точність і швидкість розпізнавання;

– хмарні системи розпізнавання мовлення – використовують віддалені сервери для обробки звукових даних. Переваги включають високу точність розпізнавання завдяки використанню потужних обчислювальних ресурсів та регулярних оновлень моделей. Недоліки полягають у залежності від стабільного інтернет-з'єднання та можливих проблемах із конфіденційністю даних;

– гібридні системи – поєднують елементи локальних та хмарних систем. Наприклад, початкове оброблення може здійснюватися локально, а складніші завдання – на віддалених серверах. Це дозволяє оптимізувати продуктивність і зменшити затримки, зберігаючи при цьому високий рівень точності;

– системи на основі попередньо натренованих моделей – використовують моделі нейронних мереж, що були натреновані на великих наборах даних. Перевагою таких систем є висока точність розпізнавання і можливість швидкої адаптації до нових задач. Недоліками можуть бути значні обчислювальні ресурси, необхідні для початкового тренування моделей.

Структура роботи складається з декількох розділів, у яких детально розглянуто теоретичні аспекти, апаратні компоненти, програмне забезпечення, методи тестування та результати експериментів.

Система на базі Whisper AI є прикладом локальної системи розпізнавання мовлення, яка використовує потужність нейронних мереж для аналізу та обробки звукових сигналів. Whisper AI дозволяє досягти високої точності розпізнавання завдяки своїм передовим моделям, які були натреновані на великих обсягах даних.

Виконання роботи не потребує тестування з використанням Raspberry Pi, оскільки мікрокомп'ютер за архітектурою подібний до звичайного персонального комп'ютера та здатен виконувати ті ж завдання, що й будь-який інший пристрій на базі Linux. Завдяки цьому, всі етапи розробки та тестування можуть бути виконані на звичайному комп'ютері, а кінцевий результат може бути перенесений на Raspberry Pi без суттєвих змін.

В ході роботи буде розроблено систему, яка складається з кількох основних компонентів: апаратної частини на базі Raspberry Pi, програмного забезпечення для розпізнавання мовлення, побудованого на основі Whisper AI, та інтерфейсу для взаємодії з користувачем. Кожен з цих компонентів буде детально розглянуто.

Загалом, метою даної роботи є створення прототипу системи розпізнавання мовлення, яка буде ефективною, доступною та легкою у використанні. Така система може бути використана в різних прикладних областях, зокрема для автоматизації побутових пристроїв, в освітніх програмах для людей з обмеженими можливостями, або як частина більш складних систем штучного інтелекту.

1 ОГЛЯД ЛІТЕРАТУРИ ТА АНАЛІЗ СУЧАСНОГО СТАНУ ПРОБЛЕМИ

Система розпізнавання мовлення на базі Raspberry Pi та Whisper AI – це інноваційний проєкт, спрямований на створення ефективної системи, яка здатна розпізнавати та інтерпретувати голосові команди, фрази та слова. Вона поєднує в собі потужність мікрокомп'ютера Raspberry Pi та передові алгоритми штучного інтелекту, реалізовані за допомогою Whisper AI або схожих технологій.

Основні компоненти системи включають в себе мікрокомп'ютер Raspberry Pi, який виступає як платформа для виконання програмного забезпечення та обробки аудіосигналів. Whisper AI використовується для аналізу та розпізнавання мовлення з використанням методів машинного навчання та нейронних мереж. Ця комбінація дозволяє системі точно та ефективно розпізнавати мовлення в реальному часі.

Система може бути використана у різних сферах, включаючи домашній розваги (наприклад, голосові помічники), комерційні застосування (автоматизовані системи управління), медичні технології (реабілітація та допомога особам з обмеженими можливостями) та багато інших. Вона може бути використана як самостійний пристрій або вбудована у існуючі системи.

1.1 Існуючі рішення

На сучасному ринку індустрії голосових технологій дійсно існують готові рішення, такі як Amazon Alexa та Google Assistant, які надають широкий спектр функціональності для розпізнавання мовлення та виконання різних завдань. Вони забезпечують зручний та ефективний спосіб взаємодії з електронікою та іншими пристроями за допомогою голосових команд.

Amazon Alexa і Google Assistant мають свої унікальні переваги та функціональність, і вони широко використовуються в різних сферах, включаючи домашній розваги, комерційні застосування та медичні технології.

Вони підтримують інтеграцію з різноманітними пристроями та сервісами, що робить їх дуже популярними серед розробників та користувачів.

1.1.1 Amazon Alexa

Amazon Alexa – це голосовий асистент, розроблений компанією Amazon. Він працює на різних пристроях, включаючи «розумні» динаміки Echo, смартфони, планшети та інші електронні пристрої. Amazon Alexa здатний розпізнавати голосові команди користувача та виконувати різні завдання, такі як відтворення музики, надання інформації про погоду, новини, керування домашніми пристроями та багато іншого.

Рисунок 1.1 – Девайс Amazon Echo Dot, який використовує Amazon Alexa

Основні характеристики Amazon Alexa включають:

– голосове взаємодія з пристроями та послугами;

– розпізнавання голосових команд користувача;

– широкий спектр функцій, від простих завдань, таких як нагадування або нагадування про події, до складніших, таких як керування побутовими пристроями або виконання покупок онлайн.

Amazon Alexa також надає розробникам доступ до своєї платформи, що дозволяє створювати власні навички (Skills) для розширення

функціональності асистента. Це дозволяє розробникам створювати інтерактивні додатки та сервіси, які можна інтегрувати з Amazon Alexa та використовувати для взаємодії з користувачами через голосові команди.

У цілому, Amazon Alexa є потужним та розширюваним голосовим асистентом, який знаходить широке застосування в різних сферах, від домашнього розваг до бізнесу

1.1.2 Google Assistant

Google Assistant – це голосовий асистент, розроблений компанією Google. Він доступний на різних платформах, включаючи смартфони, планшети, розумні динаміки та інші електронні пристрої. Google Assistant працює на основі штучного інтелекту та машинного навчання, що дозволяє йому розпізнавати голосові команди користувача та виконувати різні завдання.

Рисунок 1.2 – Девайс Google Nest Mini, який використовує Google Assistant

Основні характеристики Google Assistant включають:

– голосову взаємодію з пристроями та послугами: користувачі можуть взаємодіяти з Google Assistant за допомогою голосових команд для виконання різних завдань, від пошуку інформації в Інтернеті до керування домашніми пристроями та виконання різних інших операцій;

– персоналізацію інтерфейсу: Google Assistant враховує індивідуальні вподобання користувачів та надає персоналізовані рекомендації та інформацію;

– інтеграцію з Google-екосистемою: Google Assistant інтегрований з іншими продуктами та сервісами Google, такими як Gmail, Google Calendar, Google Maps тощо, що дозволяє виконувати різні завдання за допомогою голосових команд.

Крім того, Google надає платформу Actions on Google для розробників, яка дозволяє створювати власні розширені функції та навички для Google Assistant. Це відкриває можливості для створення інтерактивних додатків та сервісів, які можна інтегрувати з Google Assistant та використовувати для взаємодії зкористувачами через голосові команди.

Загалом, Google Assistant є потужним та розширюваним голосовим асистентом, який має широкий спектр застосувань у різних сферах життя, від особистих задач до бізнесу та розваг.

1.1.3 Порівняння

Отже, проведемо порівняльний аналіз між Amazon Alexa та Google Assistant за кількома ключовими параметрами:

1) характеристики та функціональність:

а) Amazon Alexa: має широкий спектр функцій, включаючи відтворення музики, керування домашніми пристроями, надання інформації про погоду, новини та багато іншого. Підтримує різноманітні навички (Skills), які розширюють її функціонал;

б) Google Assistant: надає подібний широкий спектр функцій, включаючи пошук інформації, керування календарем, виконання завдань та багато іншого. Інтегрований з Google-екосистемою, що дозволяє доступатися до інших сервісів та продуктів Google;

2) персоналізація:

а) Amazon Alexa: підтримує індивідуалізацію, але не на такому рівні, як Google Assistant. Вона працює на основі профілю користувача та історії взаємодії;

б) Google Assistant: має вищий рівень персоналізації, оскільки використовує дані від Google профілю користувача, його інтереси та історію взаємодії;

3) інтеграція та екосистема:

а) Amazon Alexa: має велику кількість сумісних пристроїв та сервісів, але деякі інтеграції можуть бути обмеженими;

б) Google Assistant: інтегрований з широким спектром Google-продуктів та сервісів, що забезпечує більшу сумісність з різноманітними сервісами;

4) розробницька екосистема:

а) Amazon Alexa: має зручний інтерфейс для розробки навичок (Skills) та розширень;

б) Google Assistant: надає доступ до розширеного API (Application Programming Interface) для розробки власних додатків та навичок;

5) Доступність та поширеність:

а) Amazon Alexa: популярний у США та деяких інших країнах, але може мати обмежену підтримку в інших регіонах;

б) Google Assistant: широко поширений та доступний на багатьох мовах та в країнах по всьому світу.

Загальний підсумок полягає в тому, що як Amazon Alexa, так і Google Assistant є потужними голосовими асистентами з великим спектром функцій та можливостей. Обидва сервіси мають свої унікальні переваги та можливості: Amazon Alexa – відома своєю широкою сумісністю з різними пристроями та сервісами, а також має велику кількість навичок (Skills), які можна розширювати; Google Assistant – інтегрований з Google-екосистемою та має вищий рівень персоналізації та ширшу доступність в різних країнах.

1.2 Вимоги до аппаратного забеспечення

Процесор має бути потужним з достатньою обчислювальною потужністю для обробки аудіосигналів у реальному часі та виконання алгоритмів розпізнавання мовлення. Рекомендується використання мікрокомп'ютера Raspberry Pi з процесором класу ARM (Advanced RISC Machines) Cortex-A.

Оперативна пам'ять (RAM) має бути не менше 2 ГБайт для ефективної обробки аудіоданих та виконання алгоритмів машинного навчання.

Пам'ять для збереження моделей машинного навчання та інших необхідних даних має бути не менше 16 ГБайт. Рекомендується використання microSD-карти.

Аудіоінтерфейс має підтримувати високоякісний збір та передачу аудіосигналів з мікрофонів. Рекомендується наявність вбудованого аудіовходу або використання зовнішнього USB (Universal Serial Bus) аудіоінтерфейсу.

Мікрофони мають бути високоякісними з мінімальним рівнем шуму та здатністю до збору якісного аудіосигналу з різних джерел та напрямків.

Наявність необхідних інтерфейсів для підключення додаткових пристроїв, таких як динаміки, кнопки або інші сенсори для керування та взаємодії з системою.

Система повина мати стабільне живлення з достатнім резервом потужності для уникнення перебоїв.

Має бути можливість підключення до Інтернету для отримання оновлень моделей розпізнавання мовлення та передачі даних.

Аналізуючи вимоги до апаратного забезпечення для системи розпізнавання мовлення на базі Raspberry Pi та Whisper AI, можна зробити висновок про необхідність використання потужних та ефективних компонентів для забезпечення оптимальної продуктивності та функціональності системи. Важливо мати процесор з достатньою обчислювальною потужністю, достатню кількість оперативної пам'яті для ефективної роботи алгоритмів розпізнавання мовлення, а також достатній

обсяг зберігального простору для зберігання даних та моделей машинного навчання.

Крім того, важливо мати належний аудіоінтерфейс та мікрофони для збору якісного аудіосигналу, а також належні інтерфейси для взаємодії з іншими пристроями та забезпечення стабільного живлення та підключення до Інтернету. Всі ці компоненти допоможуть забезпечити оптимальну роботу системи розпізнавання мовлення та забезпечити зручну та ефективну взаємодію з користувачем.

1.3 Патенти пов’язані з розпизнанням мови

У базі патентів існують схожі за ідеєю розробки, наприклад носимий пристрій для перетворення мови в текст, який описано у патенті 202211031314 Індії [2]. Цей пристрій спрямований на допомогу людям з вадами слуху, забезпечуючи переклад мови в текст у режимі реального часу.

Ще одним прикладом є портативний пристрій для перекладу мови жестів, описаний у патенті 202211067023 [1] Індії. Цей пристрій також призначений для допомоги людям з вадами слуху, забезпечуючи переклад жестової мови на текст чи мову. Така технологія може бути корисною для розширення можливостей систем розпізнавання мови, зокрема для інтеграції з системами підтримки жестової мови.

Значний інтерес становить патент 201741023010 [3] Індії, який описує енергоефективні методи використання Raspberry Pi. Враховуючи, що системи розпізнавання мови можуть вимагати значних обчислювальних ресурсів, застосування таких методів може значно покращити енергоефективність та тривалість роботи пристроїв, особливо в автономних системах.

Ці патенти демонструють широкий спектр можливостей та інновацій, які можуть бути використані та інтегровані в розробку експериментальної системи розпізнавання мови на базі Raspberry Pi 3, як описано в статті «Experimental Speech Recognition System Based on Raspberry Pi 3».

Використання цих технологій може не лише покращити продуктивність системи, але й забезпечити додаткові функції та підвищити її ефективність.

1.3.1 Носимий пристрій для перетворення мови в текст

Патент 202211031314 [5] описує носимий пристрій для перетворення мови в текст, який призначений для допомоги людям з вадами слуху. Цей пристрій забезпечує переклад мови в текст у режимі реального часу, що значно покращує комунікацію для людей, які не можуть повністю або частково чути.

Основні компоненти та функціональність:

1) мікрофон:

а) основний вхідний компонент пристрою, який захоплює мовні сигнали;

б) мікрофон може бути високочутливим для забезпечення чіткого і точного захоплення звуку в різних умовах оточення;

2) обчислювальний модуль:

а) відповідає за обробку захоплених мовних сигналів;

б) використовує алгоритми розпізнавання мови для перетворення аудіо сигналу в текстові дані;

в) може включати технології машинного навчання для підвищення точності та ефективності розпізнавання;

3) дисплей:

а) виводить перетворений текст у режимі реального часу;

б) дисплей може бути інтегрованим у носимий пристрій, наприклад, у наручний годинник або інший аксесуар, щоб забезпечити зручність використання;

4) живлення:

а) пристрій оснащений джерелом живлення, яке може бути перезаряджаємим;

б) може використовувати енергоефективні технології для забезпечення тривалої роботи без необхідності частих зарядок;

5) зв’язок:

а) може включати можливість бездротового зв’язку для передачі даних на інші пристрої, наприклад, смартфони або комп’ютери;

б) зв’язок може здійснюватися через Bluetooth або Wi-Fi.

Переваги та можливості:

1) покращення комунікації:

а) забезпечує миттєвий переклад мовлення в текст, що значно покращує можливості комунікації для людей з вадами слуху;

б) може використовуватися в різних ситуаціях, від повсякденного спілкування до роботи чи навчання;

2) портативність і зручність:

а) компактний і носимий формат дозволяє користувачам легко переносити пристрій і використовувати його у будь-який час;

б) інтеграція з повсякденними аксесуарами робить пристрій непомітним і зручним для носіння;

3) адаптивність:

а) може адаптуватися до різних мов і акцентів завдяки використанню технологій машинного навчання;

б) забезпечує високу точність розпізнавання в різних умовах оточення, включаючи шумні місця;

4) інтеграція з іншими пристроями:

а) можливість підключення до смартфонів і комп’ютерів дозволяє розширити функціональність пристрою;

б) інтеграція з іншими додатками може забезпечити додаткові можливості, такі як збереження текстових даних або їх подальший аналіз.

Використання та потенційні застосування:

1) щоденне життя:

а) допомагає людям з вадами слуху легко спілкуватися з оточуючими, не відчуваючи дискомфорту чи незручностей;

б) може використовуватися у магазинах, на вулиці, в транспорті та інших громадських місцях;

2) освіта:

а) підтримує навчальний процес, дозволяючи студентам з вадами слуху брати активну участь у заняттях;

б) може використовуватися для перекладу лекцій та інших навчальних матеріалів;

3) робота:

а) допомагає у професійному середовищі, забезпечуючи ефективне спілкування між колегами;

б) може використовуватися на зустрічах, конференціях та інших робочих заходах;

4) медицина:

а) може бути корисним для медичних працівників та пацієнтів з вадами слуху, забезпечуючи точне та швидке спілкування;

б) допомагає в ситуаціях, де важлива швидка передача інформації.

Цей список описує основні компоненти та можливості пристрою для розпізнавання мовлення, включаючи мікрофон, обчислювальний модуль, дисплей, живлення і зв'язок. Кожен з компонентів має свої характеристики та переваги, що забезпечують зручність використання, високу точність розпізнавання мовлення та можливість інтеграції з іншими пристроями, роблячи пристрій корисним у різних сферах життя, таких як щоденне спілкування, освіта, робота та медицина.

1.3.2 Методи підвищення енергоефективності Raspberry Pi

Патент 201741023010 [4] описує методи підвищення енергоефективності при використанні Raspberry Pi. Основна мета полягає в зменшенні енергоспоживання пристрою без зниження його продуктивності.

Основні компоненти та методи:

а) оптимізація програмного забезпечення:

1) використання алгоритмів, які знижують споживання енергії під час виконання завдань.

2) впровадження енергоефективних режимів роботи.

б) апаратні рішення:

1) використання компонентів з низьким енергоспоживанням.

2) оптимізація роботи процесора та інших компонентів.

в) моніторинг і керування енергоспоживанням:

1) постійний моніторинг енергоспоживання пристрою.

2) автоматичне регулювання параметрів роботи для зниження енергоспоживання.

г) застосування:

1) автономні системи: використання у пристроях, що працюють від батарей.

2) інтернет речей (IoT): підвищення ефективності роботи IoT-пристроїв.

3) енергоефективні проєкти: будь-які проєкти, де важливо мінімізувати споживання енергії.

Патент 201741023010 пропонує рішення для ефективного використання енергії в проєктах на базі Raspberry Pi, що робить його цінним для розробників, які прагнуть створювати енергоефективні пристрої.

1.3.3 Висновки

Ці патенти демонструють широкий спектр можливостей та інновацій, які можуть бути використані та інтегровані в розробку експериментальної системи розпізнавання мови на базі Raspberry Pi 3, як описано в статті «Experimental Speech Recognition System Based on Raspberry Pi 3». Використання цих технологій може не лише покращити продуктивність системи, але й забезпечити додаткові функції та підвищити її ефективність.

Таким чином, аналіз існуючих патентів та розробок в цій області дозволяє зробити висновок про значний потенціал для подальших досліджень

і вдосконалень у створенні систем розпізнавання мови, які можуть бути використані в різних сферах життя, від побутових умов до промислових застосувань.

Висновки до розділу

Система розпізнавання мовлення на базі Raspberry Pi та Whisper AI представляє собою інноваційний проєкт, спрямований на створення ефективної системи, яка здатна розпізнавати та інтерпретувати голосові команди, фрази та слова. Вона поєднує в собі потужність мікрокомп'ютера Raspberry Pi та передові алгоритми штучного інтелекту, реалізовані за допомогою Whisper AI або схожих технологій. [6]

Основні компоненти системи включають в себе мікрокомп'ютер Raspberry Pi, який виступає як платформа для виконання програмного забезпечення та обробки аудіосигналів. Whisper AI використовується для аналізу та розпізнавання мовлення з використанням методів машинного навчання та нейронних мереж. Ця комбінація дозволяє системі точно та ефективно розпізнавати мовлення в реальному часі.

У порівнянні з існуючими рішеннями, такими як Amazon Alexa та Google Assistant, система на базі Raspberry Pi та Whisper AI відкриває нові можливості для розробників та користувачів. Вона надає більшу гнучкість та контроль над розпізнаванням мовлення, а також можливість розширення та налаштування за допомогою відкритих інтерфейсів та інструментів.

Досягненням цього проєкту є створення ефективної та доступної системи розпізнавання мовлення, яка може знайти застосування в різних сферах, включаючи домашній розваги, освіту, медицину та бізнес. Завдяки поєднанню передових технологій та відкритих стандартів, система на базі Raspberry Pi та Whisper AI може стати важливим інструментом для покращення якості життя та розвитку нових інноваційних додатків у сфері голосових технологій.

Аналіз існуючих патентів свідчить про наявність численних схожих за ідеєю розробок, що підкреслює важливість і актуальність проєкту. Наприклад, носимий пристрій для перетворення мови в текст, описаний у патенті 202211031314 Індії, забезпечує миттєвий переклад мовлення в текст, що значно покращує комунікацію для людей з вадами слуху.

Портативний пристрій для перекладу мови жестів, описаний у патенті 202211067023 Індії, також є прикладом інноваційного рішення для покращення комунікації. Використання подібних технологій у поєднанні з розпізнаванням мови може розширити можливості нашої системи.

Патент 201741023010 Індії, який описує енергоефективні методи використання Raspberry Pi, є особливо цінним для проєкту. Враховуючи, що системи розпізнавання мови можуть вимагати значних обчислювальних ресурсів, впровадження таких методів дозволить значно покращити енергоефективність і тривалість роботи пристроїв, особливо в автономних системах.

Отже, впровадження інноваційних технологій, описаних у цих патентах, дозволяє створити більш ефективну та потужну систему розпізнавання мови, яка може бути використана у різних сферах життя, забезпечуючи нові можливості для користувачів та розробників.

2 АПАРАТНА ЧАСТИНА СИСТЕМИ

Апаратна частина системи розпізнавання мовлення на базі Raspberry Pi та Whisper AI складається з декількох ключових компонентів, кожен з яких виконує критично важливу функцію для забезпечення загальної ефективності та функціональності системи. Ці компоненти включають мікрокомп'ютер Raspberry Pi, який слугує основною обчислювальною платформою, високочутливий мікрофон для захоплення мовних сигналів, дисплей для виведення перетвореного тексту, а також різноманітні комунікаційні модулі для забезпечення підключення до інших пристроїв та мереж.

Кожен з цих компонентів відіграє свою унікальну роль у системі: мікрокомп'ютер Raspberry Pi забезпечує обчислювальну потужність і керування, мікрофон захоплює аудіосигнали, які потім обробляються Whisper AI, а дисплей надає користувачеві зручний інтерфейс для перегляду результатів розпізнавання мовлення. Комунікаційні модулі дозволяють системі підключатися до інтернету або інших пристроїв, що забезпечує додаткові можливості для інтеграції та розширення функціональності.

У цьому розділі ми детально розглянемо кожен з цих компонентів, описуючи їхні технічні характеристики, функціональні можливості та взаємодію між собою. Це дозволить зрозуміти, як кожен з компонентів сприяє загальній роботі системи та які переваги вони надають у контексті розпізнавання мовлення.

2.1 Мікрофон

В якості мікрофона можна використовувати HOCO L14, який є високочутливим мікрофоном з роз'ємом USB, або будь-яку іншу сумісну гарнітуру. [21]

Рисунок 2.1 – Мікрофон петличка Hoco L14

HOCO L14 забезпечує чітке і точне захоплення звуку, що є критично важливим для ефективної роботи системи розпізнавання мовлення. Висока чутливість мікрофона дозволяє зменшити рівень шуму та покращити якість запису аудіосигналів, що сприяє більш точному розпізнаванню мовлення завдяки Whisper AI.

Таблиця 2.1 – Характеристики мікрофону HOCO L14

Характеристика	**Опис**
Тип	мікрофон петличний
Спосіб кріплення	кліпса
Сумісність	Android, Windows, ПК, Ноутбук, Телефон
Частотний діапазон	20 Гц
Чутливість	-42 дБ
Спрямованість	всеспрямований
Тип перетворювача мікрофона	конденсаторний
Опір	900 Ом
Підключення та роз'єми	
Спосіб підключення	дротовий
Інтерфейс підключення	USB Type-C

Використання саме цього мікрофону не є обов'язковим; можливе застосування будь-якої сумісної гарнітури або мікрофону з роз'ємом USB, який забезпечує високу чутливість і якість захоплення звуку. Це дає гнучкість

у виборі апаратного забезпечення, враховуючи конкретні потреби та доступні ресурси користувача.

Також підійдуть і інші мікрофони конденсаторного або мембраного типу, які забезпечують високу якість запису звуку. Використання студійних моделе забезпечить відмінну чутливість і точність, що робить їх ідеальними для систем розпізнавання мовлення. Завдяки їх високій якості та широкому частотному діапазону, ці мікрофони можуть значно покращити точність розпізнавання мовлення, зменшуючи рівень шуму та забезпечуючи чіткий захоплення аудіосигналів.

2.2 Дисплей

Як і для мікрофона, використання певного типу дисплея не є обов'язковим; можливе застосування будь-якого сумісного дисплея, який забезпечує чітке відображення тексту. Це може бути як невеликий портативний дисплей, інтегрований у пристрій, так і більший монітор, підключений через HDMI. Вибір дисплея залежить від конкретних вимог. Наприклад, для портативних рішень краще підійдуть компактні дисплеї, тоді як для стаціонарних систем можна використовувати більші екрани з високою роздільною здатністю для зручності читання. Використання різних типів дисплеїв дозволяє адаптувати систему до потреб користувачів і забезпечує гнучкість у розробці та експлуатації.

В цій роботі буде використано стаціонарний дисплей HP 24M [22], який забезпечує високу роздільну здатність та чудову якість зображення. Цей дисплей має діагональ 23.8 дюймів і роздільну здатність 1920x1080 пікселів, що дозволяє чітко відображати текст та інші графічні елементи. Завдяки широкому куту огляду 178 градусів, дисплей забезпечує зручність перегляду з різних позицій, що є важливим для комфортного використання.

Рисунок 2.2 – Монітор HP 24M

Характеристика	**Опис**
Діагональ дисплея	23.8"
Частота оновлення	60 Гц
Максимальна роздільна здатність дисплея	1920x1080 (FullHD)
Час реакції матриці	5 мс
Яскравість дисплея	250 кд/м²
Тип матриці	IPS
Інтерфейси	VGA, HDMI
Підсвітка матриці	LED (WLED)
Контрастність дисплея	1000:1
Кут огляду	178°/178°
Відношення сторін	16:9
Колір	Чорний
Покриття	Матове
Максимальна кількість кольорів	16.7 млн
Споживана потужність	22 Вт
Варіанти регулювання положення дисплея	регулювання нахилу

HP 24M також підтримує підключення через HDMI, що робить його сумісним з Raspberry Pi. Це підключення забезпечує стабільну передачу сигналу та високу якість зображення без затримок. Крім того, дисплей має тонкий дизайн і займає мінімум місця на робочому столі, що сприяє створенню ергономічного робочого простору.

Використання такого дисплея у системі розпізнавання мовлення на базі Raspberry Pi дозволяє ефективно відображати результати роботи системи, забезпечуючи зручність та точність взаємодії з користувачем.

2.3 Ruspberry Pi

В якості серця та мозку системи буде використано Raspberry Pi, який забезпечує необхідну обчислювальну потужність для обробки аудіосигналів та виконання алгоритмів розпізнавання мовлення. Raspberry Pi є компактним і економічним мікрокомп'ютером, що дозволяє легко інтегрувати його у різні проєкти. Завдяки своїй універсальності, він підтримує широкий спектр операційних систем і програмного забезпечення, включаючи спеціалізовані бібліотеки для машинного навчання та обробки звуку.

Рисунок 2.3 – Raspberry Pi 4 Model B

Raspberry Pi представлено великою кількістю моделей, кожна з яких має свої особливості та підходить для різних завдань. Найбільш поширеними є Raspberry Pi 3 та Raspberry Pi 4, які відрізняються за своїми характеристиками та можливостями.

Raspberry Pi 3 оснащений ARM Cortex-A53 процесором з частотою 1.2ГГц, 1 ГБайт оперативної пам'яті LPDDR2, а також має інтерфейси HDMI, USB, Ethernet та Wi-Fi. Ця модель є хорошим вибором для більшості проєктів, де важлива компактність і енергоефективність.

Таблиця 2.2 – Характеристики мікрокомп'ютера Raspberry Pi 3 Model B

Характеристика	**Опис**
Процесор	Quad Core 1.2ГГц Broadcom BCM2837 64bit CPU
Оперативна пам'ять	1 Гб
Безпровідні технології	BCM43438 wireless LAN та Bluetooth Low Energy (BLE)
Мережеві порти	100 Base Ethernet
GPIO піни	40-pin
USB порти	4 USB 2.0
Аудіо та відео виходи	4 Pole stereo output and composite video port
Відео порти	1 Full size HDMI®
Порт камери	CSI
Порт дисплея	DSI
Порт накопичувача	Micro SD
Джерело живлення	Micro USB джерело живлення до 2.5A

Raspberry Pi 4 є більш потужною версією, оснащеною ARM Cortex-A72 (Quad Core) процесором з частотою 1.5 ГГц, і може мати від 2 ГБайт до 8 ГБайт оперативної пам'яті LPDDR4. Крім того, Raspberry Pi 4 підтримує дві 4K дисплеї завдяки двом мікро-HDMI портам, має швидший Ethernet та USB 3.0 порти. Ця модель підходить для завдань, що вимагають більшої обчислювальної потужності та швидкості передачі даних. [9, 17]

Таблиця 2.3 – Характеристики мікрокомп'ютера Raspberry 4 Model B

Характеристика	**Опис**
Процесор	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.8ГГц
Оперативна пам'ять	1 ГБайт, 2 ГБайт, 4 ГБайт або 8 ГБайт LPDDR4-3200 SDRAM (залежно від моделі)
Безпровідні технології	2.4 ГГц та 5.0 ГГц IEEE 802.11ac wireless, Bluetooth 5.0, BLE
Ethernet	Gigabit Ethernet
USB порти	2 порти USB 3.0; 2 порти USB 2.0
GPIO	Стандартний 40-контактний GPIO заголовок Raspberry Pi (повністю сумісний з попередніми моделями)
HDMI	2 × micro-HDMI® порти (підтримка до 4kp60)
Порт дисплея	2-лінійний MIPI DSI дисплей порт
Порт камери	2-лінійний MIPI CSI порт камери
Аудіо та відео виходи	4-контактний стерео аудіо та композитний відео порт
Відео декодування/кодування	H.265 (4kp60 декодування), H264 (1080p60 декодування, 1080p30 кодування)
Графіка	OpenGL ES 3.1, Vulkan 1.0
Накопичувачі	Слот для карт Micro-SD для завантаження операційної системи та зберігання даних
Джерело живлення	5V DC через USB-C роз'єм (мінімум 3A*); 5V DC через GPIO заголовок (мінімум 3A*)
Живлення через Ethernet	Підтримка PoE (потребує окремого PoE HAT)
Робоча температура	0 – 50 градусів C

Також нещодавно на ринку з'явилась модель Raspberry Pi 5, яка пропонує ще вищу продуктивність і розширені можливості порівняно з попередніми моделями. Raspberry Pi 5 оснащений новітнім процесором ARM Cortex-A76 (Quad Core) з частотою 2.0 ГГц, що забезпечує значне підвищення обчислювальної потужності.

Ця модель підтримує до 16 ГБайт оперативної пам'яті LPDDR4X, що робить її ідеальною для найвимогливіших завдань, включаючи обробку великих обсягів даних, машинне навчання та мультимедійні додатки. Крім того, Raspberry Pi 5 має вдосконалену систему охолодження, що дозволяє працювати при високих навантаженнях без ризику перегріву.

Таблиця 2.5 – Характеристики мікрокомп'ютера Raspberry Pi 5 Model B

Характеристика	**Опис**
Процесор	Quad Core 2.0ГГц ARM Cortex-A76 (64-bit) CPU
Оперативна пам'ять	2 ГБайт, 4 ГБайт, 8 ГБайт або 16 ГБайт LPDDR4X-4266 SDRAM (залежно від моделі)
Безпровідні технології	2.4 ГГц та 5.0 ГГц IEEE 802.11ax (Wi-Fi 6), Bluetooth 5.2, BLE
Ethernet	Dual Gigabit Ethernet
USB порти	2 порти USB 3.1; 2 порти USB 2.0
GPIO	Стандартний 40-контактний GPIO заголовок Raspberry Pi (повністю сумісний з попередніми моделями)
HDMI	2 × micro-HDMI® порти (підтримка до 4kp120)
Порт дисплея	2-лінійний MIPI DSI дисплей порт
Порт камери	2-лінійний MIPI CSI порт камери
Аудіо та відео виходи	4-контактний стерео аудіо та композитний відео порт
Відео декодування/кодування	H.265 (4kp120 декодування), H264 (1080p120 декодування, 1080p60 кодування)
Графіка	OpenGL ES 3.2, Vulkan 1.1
Накопичувачі	Слот для карт Micro-SD для завантаження операційної системи та зберігання даних
Джерело живлення	5V DC через USB-C роз'єм (мінімум 3.5A*); 5V DC через GPIO заголовок (мінімум 3.5A*)
Живлення через Ethernet	Підтримка PoE+ (потребує окремого PoE+ HAT)
Робоча температура	0 – 50 градусів C

Raspberry Pi 5 також підтримує нові стандарти бездротового зв'язку, такі як Wi-Fi 6, що забезпечує вищу швидкість передачі даних та кращу стабільність з'єднання. Dual Gigabit Ethernet дозволяє одночасно використовувати дві мережеві підключення, що може бути корисно для проєктів, які потребують високу пропускну здатність мережі.

Завдяки підтримці двох micro-HDMI портів з роздільною здатністю до 4K при 120 Гц, Raspberry Pi 5 може бути використаний у високоякісних мультимедійних системах. Підтримка нових графічних стандартів OpenGL ES 3.2 та Vulkan 1.1 відкриває додаткові можливості для розробників графічних додатків [23].

Таким чином, Raspberry Pi 5 представляє собою найпотужнішу та найсучаснішу модель у лінійці Raspberry Pi, що забезпечує високу продуктивність та широкий спектр можливостей для різноманітних проєктів, від простих автоматизаційних завдань до складних мультимедійних та обчислювальних додатків.

Ще одним критерієм вибору є ціна, яка зазвичай залежить від кількості пам'яті та додаткових можливостей, що надаються різними моделями Raspberry Pi. Raspberry Pi 3, завдяки своїй простоті та базовим характеристикам, є більш доступним варіантом для проєктів з обмеженим бюджетом. Це робить його привабливим вибором для освітніх проєктів, прототипування та домашнього використання, де висока обчислювальна потужність не є критичною вимогою.

З іншого боку, Raspberry Pi 4, особливо моделі з більшим обсягом оперативної пам'яті, мають вищу ціну, але також пропонують значно більшу продуктивність і функціональність. Це дозволяє використовувати їх у більш вимогливих додатках, таких як обробка великих обсягів даних, машинне навчання та мультимедійні системи. Вищий обсяг пам'яті дозволяє ефективніше виконувати складні завдання, що потребують багато ресурсів.

Таблиця 2.5 – Порівняння цінових категорій Raspberry Pi 3 та 4

Модель	**Ціна (USD)**	**Ціна (UAH)**
Raspberry Pi 3 Model B	$35 – $40	1,364.30 – 1,559.20 грн.
Raspberry Pi 4 Model B (2 ГБайт RAM)	$45 – $50	1,754.10 – 1,949.00 грн.
Raspberry Pi 4 Model B (4 ГБайт RAM)	$55 – $60	2,143.90 – 2,338.80 грн.
Raspberry Pi 4 Model B (8 ГБайт RAM)	$75 – $80	2,923.50 – 3,118.40 грн.
Raspberry Pi 5 Model B (4 ГБайт RAM)	$95 – $100	3,703.10 – 3,898.00 грн
Raspberry Pi 5 Model B (8 ГБайт RAM)	$110 – $115	4,287.80 – 4,482.70 грн

Ціна також може варіюватися залежно від регіону та постачальника, але загалом Raspberry Pi 3 залишається більш доступним варіантом. Для проєктів, де важлива висока продуктивність, доцільніше розглянути Raspberry Pi 4 з більшим обсягом оперативної пам'яті, незважаючи на її вищу вартість.

У висновку можна зазначити, що вибір моделі Raspberry Pi для системи розпізнавання мовлення на базі Whisper AI залежить від конкретних вимог проєкту та бюджету. Raspberry Pi 3 є чудовим варіантом для проєктів з обмеженим бюджетом, забезпечуючи базову функціональність та енергоефективність. Він підходить для освітніх проєктів, прототипування та домашнього використання.

Raspberry Pi 4, з його вищою продуктивністю та можливістю вибору оперативної пам'яті до 8 ГБайт, підходить для більш вимогливих завдань, таких як обробка великих обсягів даних та мультимедійні додатки. Модель Raspberry Pi 5, яка нещодавно з'явилася на ринку, пропонує ще більшу продуктивність і функціональність, роблячи її ідеальним вибором для найвимогливіших проєктів.

Ціна на моделі Raspberry Pi варіюється залежно від обсягу оперативної пам'яті та додаткових можливостей, що робить Raspberry Pi 3 більш доступним варіантом, а Raspberry Pi 4 та 5 – оптимальним вибором для проєктів, що потребують високої продуктивності. Таким чином, вибір конкретної моделі

має базуватися на балансі між вимогами до продуктивності та бюджетом проєкту.

Висновки до розділу

ле, як було зазначено раніше, проєкт не потребує наявності самого мікрокомп'ютера, адже основні обчислювальні завдання можуть виконуватися на будь-якій сумісній платформі, здатній забезпечити необхідну обчислювальну потужність. Це відкриває можливість використовувати альтернативні рішення, такі як інші одноплатні комп'ютери або навіть стандартні настільні ПК, які можуть мати вищу продуктивність і кращу підтримку периферійних пристроїв.

Це забезпечує гнучкість у виборі апаратного забезпечення залежно від доступних ресурсів та конкретних вимог проєкту. Наприклад, використання потужнішого настільного ПК може бути доцільним для розробки та тестування системи, тоді як у фінальному продукті можна використовувати більш компактні та енергоефективні одноплатні комп'ютери.

Таким чином, вибір платформи для системи розпізнавання мовлення на базі Whisper AI залишається відкритим, дозволяючи використовувати різноманітні апаратні рішення для досягнення оптимального балансу між продуктивністю, вартістю та зручністю інтеграції в кінцевий продукт. Це також сприяє розширенню можливостей системи та адаптації її до різних умов експлуатації, що є важливим для створення універсального і ефективного рішення для розпізнавання мовлення.

Для тестування і налаштувань буде використовуватись система Windows з використанням Windows Subsystem for Linux (WSL). Це дозволяє запускати Linux-дистрибутиви безпосередньо на Windows, забезпечуючи середовище, схоже на те, що використовується на Raspberry Pi. Завдяки WSL можна легко тестувати та налаштовувати програмне забезпечення, включаючи бібліотеки для машинного навчання та обробки звуку, які необхідні для роботи Whisper AI.

Використання Windows з WSL має кілька переваг:

– зручність: Windows є широко розповсюдженою операційною системою, що забезпечує зручний інтерфейс та підтримку різноманітного програмного забезпечення, включаючи інструменти для розробки та налагодження;

– сумісність: WSL дозволяє використовувати більшість інструментів та бібліотек, які доступні в Linux, що робить процес розробки та тестування більш гнучким і менш залежним від конкретної апаратної платформи;

– ефективність: Використання WSL забезпечує ефективне виконання Linux-додатків на Windows без необхідності подвійного завантаження або використання віртуальних машин, що може знижувати продуктивність.

Таким чином, для початкового етапу розробки, тестування та налаштування програмного забезпечення для системи розпізнавання мовлення на базі Whisper AI буде використовуватися Windows з WSL. Це забезпечить зручне та ефективне середовище для розробки, яке можна легко перенести на остаточну апаратну платформу, таку як Raspberry Pi, коли всі налаштування будуть завершені.

3 ПРОГРАМНА ЧАСТИНА СИСТЕМИ

В якості операційної системи буде використовуватися Raspbian на Raspberry Pi, яка спеціально оптимізована для цього апаратного забезпечення. На етапі розробки та тестування буде використовуватися Windows з WSL для забезпечення сумісності з Linux-середовищем.

Проєкт буде виконано на мові програмування Python, що забезпечує простоту і зручність написання коду, а також підтримує широкий спектр бібліотек для обробки аудіо та машинного навчання.

Для цього буде використано бібліотеки:

– wave для роботи з WAV файлами, що забезпечує можливість читання і запису аудіофайлів у форматі WAV;

– tkinter для створення графічного інтерфейсу користувача (GUI), який дозволяє розробити зручний і зрозумілий інтерфейс для взаємодії з системою розпізнавання мовлення;

– pyaudio для роботи з аудіоінтерфейсами в реальному часі, забезпечуючи високу якість звуку та низьку затримку при захопленні і відтворенні аудіосигналів.

Основною частиною, навколо якої будується весь проєкт, є Whisper AI. Це потужний компонент для розпізнавання мовлення, що використовує алгоритми машинного навчання та нейронні мережі. Whisper AI Python API дозволяє інтегрувати ці моделі у Python-додатки, забезпечуючи високу точність та ефективність розпізнавання мовлення.

3.1 Raspbian (Linux)

Для розробки і тестування буде використовуватись звичайний Linux гілки Debian-подібних дистрибутивів, таких як Ubuntu або Debian. Це забезпечить сумісність з Raspbian, що дозволить проводити розробку та тестування в середовищі, максимально наближеному до цільового. Використання таких дистрибутивів надає доступ до великої кількості

інструментів та бібліотек, необхідних для розробки програмного забезпечення, що полегшує процес налаштування та тестування системи.

Налаштування таких систем не є складним завданням завдяки великій кількості доступних інструментів та документації. Використання Debian-подібних дистрибутивів спрощує процес налаштування середовища розробки, оскільки більшість необхідних бібліотек і пакетів можна легко встановити за допомогою менеджера пакетів, такого як apt.

У список таких пакетів входить:

– python3: інтерпретатор мови Python 3;

– python3-pip: менеджер пакетів для встановлення додаткових бібліотек Python;

– python3-tk: бібліотека для роботи з графічним інтерфейсом tkinter;

– pyinstaller: набір інструментів для компіляції та побудови програмного забезпечення написаного на мові Python 3;

– git: система контролю версій для керування вихідним кодом проєкту.

У результаті система буде ідентичною до тієї, що працюватиме на Raspberry Pi з Raspbian, що дозволить проводити розробку та тестування в умовах, максимально наближених до реальних. Це забезпечить безперебійне перенесення проєкту на остаточну апаратну платформу з мінімальними змінами в конфігурації та налаштуваннях.

В свою чергу, компіляція Python-додатку у артефакт дозволить забезпечити зручне розгортання та запуск програми на різних платформах без необхідності встановлення додаткових залежностей. Це значно спростить процес інсталяції та конфігурації системи, зменшить ймовірність виникнення помилок, пов'язаних з несумісністю бібліотек або середовища виконання.

3.2 Python та бібліотеки

Розробку програмного забезпечення було виконано на мові Python версії 3.12. Він є популярною мовою серед розробників завдяки своїй простоті,

зрозумілості та великій кількості доступних бібліотек. Він забезпечує потужні інструменти для роботи з даними, обробки звуку та машинного навчання, що робить його ідеальним вибором для створення системи розпізнавання мовлення.

Пакування додатку у артефакт буде відбуватись за допомогою PyInstaller. PyInstaller дозволяє упакувати Python-додаток разом з усіма необхідними бібліотеками та залежностями в один виконуваний файл, що значно спрощує процес розгортання та запуску програми на різних платформах. Використання PyInstaller забезпечує зручне розгортання, мінімізуючи ризики виникнення помилок, пов'язаних з несумісністю бібліотек або середовища виконання.

Використання бібліотеки wave у контексті проєкту забезпечить можливість ефективного читання та запису аудіофайлів у форматі WAV. Ця бібліотека є стандартною для Python і дозволяє легко працювати з аудіоданими, що є важливим для зберігання та подальшої обробки звукових сигналів. Вона забезпечує зручний доступ до аудіоданих, дозволяючи читати, записувати та маніпулювати WAV файлами без додаткових зусиль.

Tkinter є доволі зручною та потужною бібліотекою для створення графічного інтерфейсу користувача (GUI) у Python. Вона дозволяє розробити зручний і зрозумілий інтерфейс для взаємодії з системою розпізнавання мовлення, що полегшує користувачеві роботу з системою. Tkinter надає широкий набір інструментів для створення вікон, кнопок, текстових полів та інших елементів інтерфейсу, що дозволяє швидко та ефективно створювати GUI для різних додатків.

У контексті проєкту, tkinter є важливим інструментом для створення графічного інтерфейсу користувача (GUI). Його використання дозволяє розробити зручний та інтуїтивно зрозумілий інтерфейс для взаємодії з системою розпізнавання мовлення, що значно полегшує користувачеві роботу з системою. Tkinter надає можливість створення різноманітних елементів інтерфейсу, таких як вікна, кнопки, текстові поля, етикетки та інші

компоненти, що дозволяє швидко та ефективно розробляти GUI для Python-додатків.

PyAudio в свою чергу дозволить працювати з аудіо інтерфейсами в реальному часі. Використання PyAudio забезпечить захоплення та відтворення аудіо сигналів з високою якістю звуку та низькою затримкою, що є критично важливим для системи розпізнавання мовлення. PyAudio надає інструменти для роботи з мікрофонами та іншими аудіо пристроями, що дозволяє легко інтегрувати захоплення звуку у ваш додаток на Python.

3.2.1 Висновок

Розробка програмного забезпечення була виконана на мові Python версії 3.12, яка є популярною серед розробників завдяки своїй простоті, зрозумілості та великій кількості доступних бібліотек. Python забезпечує потужні інструменти для роботи з даними, обробки звуку та машинного навчання, що робить його ідеальним вибором для створення системи розпізнавання мовлення. Використання бібліотеки wave у контексті проєкту забезпечує ефективне читання та запис аудіофайлів у форматі WAV, що є важливим для зберігання та обробки звукових сигналів.

Пакування додатку у артефакт здійснюється за допомогою PyInstaller, який дозволяє упакувати Python-додаток разом з усіма необхідними бібліотеками та залежностями в один виконуваний файл. Це значно спрощує процес розгортання та запуску програми на різних платформах, мінімізуючи ризики, пов'язані з несумісністю бібліотек або середовища виконання. Tkinter є важливим інструментом для створення графічного інтерфейсу користувача, дозволяючи розробити інтуїтивно зрозумілий інтерфейс для взаємодії з системою розпізнавання мовлення. Використання PyAudio забезпечує захоплення та відтворення аудіо сигналів у реальному часі, що критично важливо для високоякісного розпізнавання мовлення.

Розробка програмного забезпечення для системи розпізнавання мовлення була виконана за допомогою Whisper AI, який є передовою

технологією в області обробки природної мови. Whisper AI забезпечує високу точність розпізнавання мовлення завдяки використанню сучасних алгоритмів машинного навчання та глибоких нейронних мереж. Ця технологія дозволяє ефективно обробляти аудіо сигнали різної якості, забезпечуючи точність та швидкість розпізнавання навіть у складних акустичних умовах.

3.2.2 Загальний опис систем розпізнання мови на базі ШІ

Системи розпізнавання мови на базі штучного інтелекту (ШІ) функціонують завдяки складним алгоритмам та моделям, що базуються на машинному навчанні, зокрема, на нейронних мережах. Основним завданням таких систем є перетворення аудіосигналу мови на текст, який може бути оброблений та зрозумілий іншими комп'ютерними системами. Робота таких систем включає кілька ключових етапів: попередня обробка аудіосигналу, розпізнавання фонем, побудова акустичних моделей, мовних моделей, і, зрештою, створення кінцевого тексту. [12]

Для більш глибокого розуміння роботи таких систем можна ознайомитися з науковими публікаціями та статтями, які описують методи та алгоритми, що використовуються. Рекомендовані джерела включають:

– «Автоматичне розпізнаннння мовлення: метод машиного навчання» (англ. Automatic Speech Recognition: A Deep Learning Approach), Донг Ю та Лі Денг;

– «Обробка мови та мовлення» (англ. Speech and Language Processing), Даніель Журавський та Джеймс Мартін;

– статті на ресурсах таких як arXiv, IEEE Xplore та Google Scholar.

Рисунок 3.1 – Модель фонемно-орієнтованої СРМ

Перший етап – попередня обробка аудіосигналу, вона полягає в очищенні та нормалізації вхідного звукового сигналу. Це включає видалення шуму, нормалізацію рівня гучності та розбиття сигналу на менші сегменти. Потім ці сегменти перетворюються на числові представлення, що використовуються для подальшого аналізу. Використання методів обробки сигналів, таких як перетворення Фур'є або фільтри Мель, дозволяє виділити ключові характеристики звукових хвиль. [13]

Наступний крок – розпізнавання фонем. Це найменші одиниці звуку, які можуть бути розпізнані системою. Нейронні мережі, такі як рекурентні нейронні мережі (RNN) або довготривалі короткочасні пам'яті (LSTM), використовуються для побудови акустичних моделей, що здатні ідентифікувати фонеми в реальному часі. Ці моделі тренуються на великих обсягах даних для покращення точності розпізнавання.

Далі відбувається побудова мовних моделей. Мовні моделі використовують статистичні методи та методи машинного навчання для передбачення ймовірності послідовностей слів. Нейронні мережі, зокрема трансформери, значно покращили якість мовних моделей, дозволяючи системам розпізнавати більш складні та контекстно залежні фрази. Такі моделі можуть враховувати граматичні структури, семантику та контекст попередніх слів для підвищення точності розпізнавання. [18]

Завершальний етап включає перетворення послідовності розпізнаних фонем у текст. Цей процес може включати корекцію помилок та використання додаткових мовних правил для покращення якості кінцевого тексту. Після

цього текст може бути переданий іншим системам для подальшої обробки, наприклад, для машинного перекладу, текстового аналізу або керування голосом.

3.2.3 Особливості Whisper AI

Технічно, Whisper AI являє собою потужну платформу для розпізнавання мовлення, побудовану на основі сучасних алгоритмів глибокого навчання. Використовуючи складні архітектури нейронних мереж, Whisper AI здатен обробляти аудіо сигнали з високою точністю, навіть у складних акустичних умовах. [10] Модель тренувалася на величезних обсягах даних, що включають різноманітні мовні акценти, інтонації та фоновий шум, що забезпечує її здатність надійно розпізнавати мовлення в реальному світі.

Whisper AI використовує методи обробки сигналів для попередньої обробки аудіо даних, такі як нормалізація гучності, фільтрація шуму та виявлення ключових сегментів мовлення. Нейронна мережа платформи обробляє ці дані, генеруючи текстовий вихід з високою точністю та швидкістю. Інтеграція Whisper AI з іншими інструментами, такими як PyInstaller для пакування додатків, бібліотека wave для роботи з аудіофайлами, Tkinter для створення графічних інтерфейсів користувача, та PyAudio для реального часу обробки аудіо сигналів, дозволяє створювати комплексні та надійні системи розпізнавання мовлення, що відповідають потребам сучасних користувачів.

3.2.4 Архітектура системи

Whisper AI використовує просту архітектуру «end-to-end,» реалізовану у вигляді трансформера з компонентами encoder-decoder. [7] Вхідний аудіосигнал ділиться на 30-секундні фрагменти, перетворюється у лог-Мел спектрограму, і потім передається до енкодера. Декодер навчений передбачати відповідний текстовий опис, включаючи спеціальні токени, які направляють модель виконувати завдання, такі як ідентифікація мови, визначення часових

міток на рівні фраз, багатомовна транскрипція мовлення та переклад мовлення на англійську мову.

Ця архітектура забезпечує високу точність та універсальність розпізнавання мовлення завдяки сучасним методам обробки аудіосигналів та глибокого навчання. Спеціальні токени дозволяють моделі одночасно виконувати кілька завдань, роблячи її ефективною та гнучкою для широкого спектра застосувань у сфері обробки природної мови.

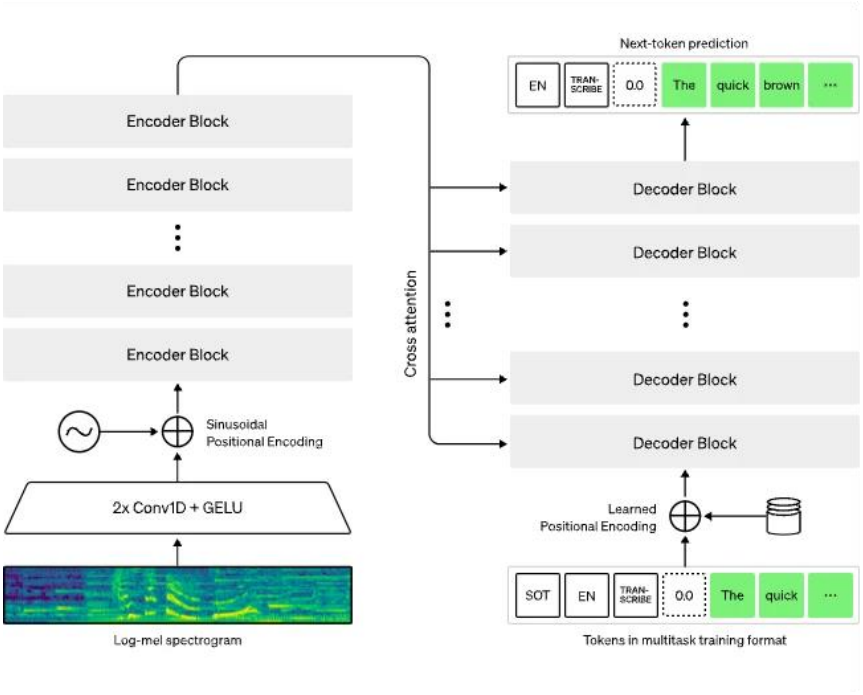


Рисунок 3.2 – Схема архітектури Whisper AI

Схема демонструє архітектуру Whisper AI, яка використовує підхід «encoder-decoder» для розпізнавання мовлення:

– вхідний сигнал: вхідний аудіосигнал перетворюється в лог-Мел спектрограму;

– енкодер: складається з кількох блоків, кожен з яких обробляє спектрограму та додає синусоїдальне позиційне кодування для збереження інформації про послідовність вхідних даних;

– перехресна увага: вихідні дані з енкодера передаються до декодера через механізм перехресної уваги (англ. cross attention);

– декодер: на вхід декодера подаються спеціальні токени та позиційні кодування, що направляють модель на виконання різних завдань;

– вихідні токени: декодер генерує вихідні текстові токени, що відповідають розпізнаному мовленню;

– різні завдання: в архітектурі передбачено використання токенів для багатозадачного навчання.

Інші існуючі підходи часто використовують менші, тісніше пов'язані аудіо-текстові набори даних для навчання або застосовують широке, але не контрольоване попереднє навчання на аудіо. Whisper був навчений на великому та різноманітному наборі даних і не був точно налаштований на жоден зокрема, тому він не перевершує моделі, спеціалізовані на виконанні тесту LibriSpeech, відомого конкурентного еталону у розпізнаванні мовлення. Однак, при вимірюванні нульового виконання Whisper на різноманітних наборах даних ми виявили, що він значно більш стійкий і робить на 50% менше помилок, ніж ці моделі. [15]

Приблизно третина аудіо-даних Whisper складається з неанглійських записів, і він поперемінно виконує завдання транскрипції на оригінальну мову або переклад на англійську. Було виявлено, що цей підхід особливо ефективний для навчання перекладу мовлення в текст і перевершує інші ШІ при перекладі на англійську в режимі нульового виконання.

3.2.5 Можливості у контексті проєкту

Whisper AI представляє собою потужний інструмент для розпізнавання мовлення, що відкриває широкі можливості для інтеграції у різні проєкти. Його здатність обробляти великі обсяги даних і працювати з різними мовами робить його незамінним для розробки багатомовних систем розпізнавання мовлення. Whisper AI використовує передові алгоритми глибокого навчання, що забезпечують високу точність навіть у складних акустичних умовах. Це робить його ідеальним для створення систем, які можуть ефективно

розпізнавати мовлення у різних середовищах, від офісних приміщень до галасливих вулиць.

Однією з ключових переваг Whisper AI є його здатність працювати з широким спектром мов і діалектів. Приблизно третина навчальних даних моделі складається з неанглійських аудіо, що дозволяє моделі ефективно транскрибувати і перекладати мовлення з різних мов. Це особливо важливо для проєктів, спрямованих на глобальну аудиторію, де необхідно забезпечити підтримку багатомовної комунікації. Whisper AI може автоматично визначати мову вхідного аудіо і виконувати транскрипцію або переклад залежно від завдання, що значно спрощує розробку багатомовних додатків.

Інтеграція Whisper AI з іншими інструментами та бібліотеками, такими як PyInstaller, wave, Tkinter та PyAudio, дозволяє створювати комплексні рішення для розпізнавання мовлення. PyInstaller забезпечує зручне пакування додатків, що полегшує їх розгортання на різних платформах. Бібліотека wave дозволяє ефективно працювати з аудіофайлами, а Tkinter надає інструменти для створення інтуїтивно зрозумілого графічного інтерфейсу користувача. PyAudio, у свою чергу, забезпечує реального часу захоплення та відтворення аудіо сигналів, що є критично важливим для точного та швидкого розпізнавання мовлення.

Whisper AI також відзначається своєю стійкістю та надійністю при роботі з різними наборами даних. Завдяки широкому та різноманітному набору даних, на якому він був навчений, Whisper AI здатен з високою точністю розпізнавати мовлення навіть у незнайомих умовах. [16] Це робить його ідеальним для застосування у проєктах, де необхідно забезпечити високу точність розпізнавання мовлення без попередньої адаптації моделі до конкретних даних. Наприклад, при створенні систем голосових помічників або автоматичних перекладачів, де важливо забезпечити стабільну роботу незалежно від умов використання.

Таким чином, можливості Whisper AI у контексті проєкту відкривають широкі перспективи для розробки інноваційних рішень у сфері розпізнавання

мовлення. Завдяки його потужним алгоритмам, багатомовній підтримці та інтеграції з іншими інструментами, Whisper AI може стати основою для створення ефективних та надійних систем розпізнавання мовлення, здатних задовольнити потреби найвимогливіших користувачів.

3.2.6 Висновки

Whisper AI демонструє значний потенціал у контексті проєктів, що потребують високоточного розпізнавання мовлення. Використання сучасних алгоритмів глибокого навчання та нейронних мереж забезпечує ефективну обробку аудіо сигналів різної якості, що дозволяє досягати високої точності навіть у складних акустичних умовах. Завдяки великому та різноманітному набору даних, на якому була навчена модель, Whisper AI виявляє високу стійкість та надійність при роботі з різними мовами та діалектами, що робить його ідеальним для багатомовних додатків.

Загалом, Whisper AI відкриває широкі перспективи для розвитку інноваційних рішень у сфері розпізнавання мовлення. Завдяки його потужним можливостям, багатомовній підтримці та інтеграції з іншими інструментами, включаючи можливості використання з Raspberry Pi, Whisper AI може стати основою для створення надійних систем, здатних задовольнити потреби найвимогливіших користувачів, забезпечуючи високу точність та надійність розпізнавання мовлення.

Висновки до розділу

Інтеграція Whisper AI з іншими інструментами та бібліотеками, такими як PyInstaller, wave, Tkinter та PyAudio, дозволяє створювати комплексні рішення для розпізнавання мовлення, що відповідають сучасним вимогам користувачів. PyInstaller спрощує процес розгортання додатків на різних платформах, а бібліотека wave забезпечує ефективну роботу з аудіофайлами. Tkinter надає інструменти для створення інтуїтивно зрозумілого графічного

інтерфейсу, тоді як PyAudio забезпечує реального часу захоплення та відтворення аудіо сигналів.

Висока точність розпізнавання мовлення та універсальність Whisper AI робить його ідеальним для застосування у різних середовищах, від офісних приміщень до галасливих вулиць. Модель здатна виконувати різноманітні завдання, такі як ідентифікація мови, транскрипція, визначення часових міток та переклад, що робить її надзвичайно гнучкою та ефективною. Whisper AI також може бути ефективно інтегрований у проєкти на базі Raspberry Pi, забезпечуючи компактні та потужні рішення для розпізнавання мовлення у реальному часі.

Розробка програмного забезпечення для цієї системи була виконана на мові Python версії 3.12, яка відзначається своєю простотою, зрозумілістю та великою кількістю доступних бібліотек. Використання бібліотек, таких як wave для роботи з WAV файлами, Tkinter для створення графічного інтерфейсу користувача та PyAudio для роботи з аудіо інтерфейсами в реальному часі, забезпечує високий рівень функціональності та зручності у розробці системи. Пакування додатку за допомогою PyInstaller значно спрощує процес розгортання та запуску програми на різних платформах.

Загалом, система розпізнавання мовлення на базі Raspberry Pi та Whisper AI демонструє значний потенціал для створення інноваційних рішень у сфері розпізнавання мовлення. Завдяки інтеграції з потужними інструментами та бібліотеками, багатомовній підтримці та високій точності розпізнавання, ця система здатна задовольнити потреби найвимогливіших користувачів. Вона відкриває широкі перспективи для розвитку ефективних та надійних систем, що можуть бути використані у різних умовах та для різних задач, забезпечуючи високу якість та зручність у експлуатації.

4 РОЗРОБКА СИСТЕМИ РОЗПІЗНАВАННЯ МОВЛЕННЯ

Розробка системи розпізнавання мовлення за допомогою Python та бібліотеки Whisper AI включає декілька важливих етапів. Whisper AI є потужною відкритою бібліотекою, яка забезпечує високу точність розпізнавання мовлення. Процес включає встановлення бібліотеки, підготовку аудіофайлів, завантаження моделі, розпізнавання мовлення та обробку результатів.

Першим етапом є встановлення необхідних бібліотек та інструментів, таких як Python і pip. Для встановлення інтерпретатора на систему Raspbian необхідно виконати кілька простих кроків. Спочатку оновлюються існуючі пакети за допомогою команд «sudo apt update» та sudo «apt upgrade».

Далі безпосередньо встановлюється сам Python, для цього використовується команда «sudo apt install python3». Також необхідно встановити пакетний менеджер pip, що досягається виконанням «sudo apt install python3-pip» у терміналі системи. Це дозволить встановлювати необхідні пакети та бібліотеки Python, так для використання whisper додамо необхідний модуль, використовуючи команду «pip3 install whisper.»

4.1 Обробка та розпізнання аудіо файлів

Розпізнавання аудіо за допомогою бібліотеки Whisper працює завдяки використанню сучасних технологій глибокого навчання та нейронних мереж. Процес включає кілька етапів: попередня обробка аудіосигналу, розпізнавання фонем, побудова акустичних та мовних моделей, і, зрештою, генерування тексту з аудіо. Whisper використовує попередньо навчені моделі, що здатні аналізувати аудіофайли та перетворювати їх у текстові дані з високою точністю. Завдяки цьому, система може ефективно розпізнавати мовлення у реальному часі та обробляти його для подальшого використання.

4.1.1 Написання коду для розпізнання мови

Виклик «import whisper» у коді призначений для імпортування бібліотеки Whisper AI у поточне середовище розробки. Ця операція дозволяє використовувати всі функції та методи, які надає бібліотека. Після імпорту стають доступними такі можливості, як завантаження моделей розпізнавання мовлення, обробка аудіофайлів, і розпізнавання тексту з аудіо. Бібліотека Whisper включає в себе попередньо навчені моделі та інструменти для ефективного розпізнавання мовлення, що забезпечує високу точність та продуктивність при роботі з аудіоданими.

```
"""Entry point of program"""

import warnings
import whisper

warnings.simplefilter("ignore")

MODEL = whisper.load_model("base")

result = MODEL.transcribe(audio="audio1.mp3")

print(result)
```

Рисунок 4.1 – Код простого сценарію для розпізнання мови з аудіо файлу

Оскільки ми використовуємо лише певні функції імпорт можна переписати наступним чином:

```
"""Entry point of program"""

from warnings import simplefilter
from whisper import load_model

simplefilter("ignore")

MODEL = load_model("base")

result = MODEL.transcribe(audio="audio1.mp3")

print(result)
```

Рисунок 4.2 – Організація імортів

Виклик «warnings.simplefilter("ignore")» у Python призначений для зміни поведінки системи щодо попереджень. Ця команда налаштовує фільтр попереджень таким чином, щоб ігнорувати всі попередження, які можуть виникати під час виконання програми.

Зазвичай, Python генерує попередження для інформування розробника про потенційні проблеми або небажані поведінки, які можуть не бути критичними помилками, але заслуговують на увагу. Використання «warnings.simplefilter("ignore")» означає, що такі повідомлення не будуть відображатися, що може бути корисним у певних сценаріях, коли розробник знає про потенційні проблеми і свідомо вирішує їх ігнорувати для забезпечення чистоти виводу або для зменшення шуму в повідомленнях під час виконання програми.

Виклик «whisper.load_model("base")» у Python призначений для завантаження певної моделі розпізнавання мовлення з бібліотеки Whisper AI. Під час цього процесу функція виконує кілька важливих дій.

По-перше, вона визначає та завантажує ваги моделі для конкретної конфігурації («base» в даному випадку) з локального сховища або з інтернету, якщо модель ще не завантажена. Ці ваги є параметрами нейронної мережі, що використовуються для перетворення аудіо сигналів у текст.

По-друге, функція ініціалізує модель з відповідною архітектурою та параметрами, що дозволяє її використовувати для подальшого розпізнавання мовлення. Модель «base» є збалансованою по точності та продуктивності, що робить її підходящою для багатьох завдань.

У підсумку, після виклику whisper.load_model(«base»), модель готова до використання для розпізнавання мовлення з аудіофайлів за допомогою методів бібліотеки Whisper AI.

Виклик «result = model.transcribe("audio.wav")» в Python ініціює процес розпізнавання мовлення з аудіофайлу «audio.wav» за допомогою завантаженої моделі Whisper. Під час цього виклику аудіофайл спочатку обробляється для виділення звукових характеристик. Потім модель застосовує свої нейронні

мережі для аналізу цих характеристик, розпізнає окремі фонеми та слова, і генерує текстову версію вхідного аудіо. Результат, який містить розпізнаний текст, зберігається в змінній result і може бути виведений або використаний далі в програмі.

Виклик «print(result)» у Python відображає на екрані значення змінної result. Після виконання функції «model.transcribe("audio.wav")», змінна result містить результат розпізнавання мовлення, включаючи текстову інтерпретацію аудіофайлу. Команда «print(result)» виводить цей результат у консоль або термінал, що дозволяє розробнику або користувачу побачити розпізнаний текст або інші дані, що містяться у змінній result. Це корисно для перевірки та налагодження роботи моделі розпізнавання мовлення.

4.1.2 Перевірки роботи сценарію

В ході виконання програми виникла помилка, яка може говорити про те, шо бібліотека нейроної мережі PyTorch була встановлена неправильно, для вирішення цієї проблеми можна перейти на офіційну сторінку бібліотеку та сгенерувати посилання для корректного встановлення модулю.

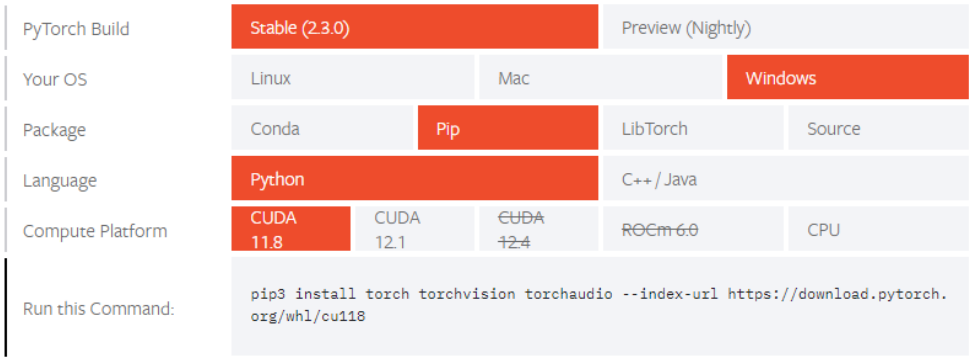


Рисунок 4.3 – Генерація посилання для встановлення бібліотеки PyTorch

Також для коректної роботи Whisper AI потрібно встановити бібліотеку ffmpeg, яка відповідає за обробку мультимедійних даних. FFmpeg дозволяє декодувати, кодувати, перетворювати і стримити аудіо та відео. Це необхідно для забезпечення сумісності з різними форматами аудіофайлів, які використовуються у процесі розпізнавання мовлення.

Для перевірки роботи сценарію буде використано дві пісні: «They Don't Care About Us» Майкла Джексона для тестування англійською мовою та «Човен» гурту Один в каное для тестування українською мовою.

```
PS E:\code\python\whiperai> python main.py
 All I wanna say is that they don't really care about us. Don't worry when people
us. Enough is enough for this garbage. All I wanna say is that they don't really c
hat they don't really care about us. All I wanna say is that they don't really car
I wanna say is that they don't really care about us. All I wanna say is that they
e about us. All I wanna say is that they don't really care about us. All I wanna s
don't really care about us. Skinhead dead dead dead, everybody gone. I'm back, shi
at. All I wanna do is everybody gone. I'm back, back, shut dead, everybody gone. I
anna say is that they don't really care about us. Beat me, hate me, you can never
rybody do me. Kick me, kill me. Don't you black away. All I wanna say is that they
re about us. I got me, but I've become my man. I have a man that you killed and yo
in the big, the mother, big. You bring me up, I'm proud of the god's sake. I'm bac
d dead, hit everybody gone. I'm back, chop it, just speculation. Everybody allocat
man. Don't flop out the angels. All I wanna say is that they don't really care abo
 but I've become my man. I'm back, I'm proud of the god's sake. I'm back, I'm prou
 proud of the god's sake. I'm back, I'm proud of the god's sake. I'm back, I'm pro
m proud of the god's sake. I'm back, I'm proud of the god's sake. He won the last
```

Рисунок 4.4 – Результати обробки пісні «They Don't Care About Us» Майкла Джексона

У результаті обробки пісні «They Don't Care About Us» Майкла Джексона маємо наступний розпізнаний текст. Це демонструє можливості моделі Whisper AI у розпізнаванні англійського мовлення з аудіофайлів. Видно, що розпізнаний текст містить багато повторюваних фраз і певні неточності, що можуть бути пов'язані з якістю аудіо або характеристиками моделі. Для порівнянн повний текст пісні наведено у ДОДАТОК Б.

Для точнішого розпізнавання можна використовувати більш високоякісні аудіофайли або більш потужні моделі Whisper AI, а також додаткові методи очищення та нормалізації аудіо перед обробкою.

Наявність помилок у розпізнанні зумовлена кількома факторами. По-перше, якість аудіофайлу грає важливу роль: шуми, ехо та інші перешкоди можуть вплинути на точність розпізнавання. По-друге, обрана модель може мати свої обмеження; менші моделі (наприклад, «tiny» або «base») можуть не мати достатньої потужності для точного розпізнавання складних аудіозаписів. Також впливає мовна специфіка пісень та акцент виконавця, що може додатково ускладнювати процес розпізнавання.

Також, оскільки для тесту було використано пісню, аудіофайл наповнений музикою та різними шумами, що ускладнює процес розпізнавання мовлення. Музичні інструменти, фонові звуки та ефекти можуть створювати додаткові перешкоди для алгоритму, знижуючи точність розпізнавання. Для досягнення кращих результатів можна використовувати чисті аудіозаписи мовлення без музичних супроводів або застосовувати методи фільтрації шуму та попередньої обробки звуку.

Ще модель надає інформацію про саме аудіо. Це включає параметри, які відображають різні аспекти розпізнавання, напираклад мову тексту, яку вона визначила як англійську ('language': 'en')

Перевіримо можливості розпізнання для українскької пісні:

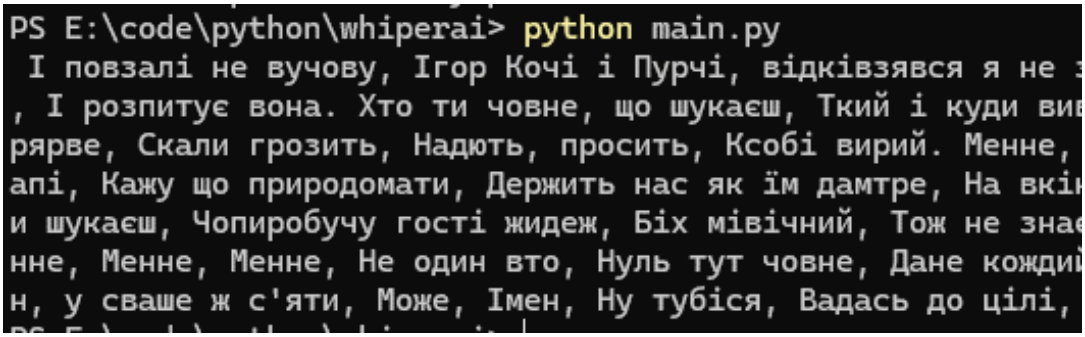


Рисунок 4.5 – Результати обробки пісні «Човен» гурту Один в каное

У результаті обробки пісні «Човен» гурту Один в каное маємо такий результат. Мова аудіофайлу була розпізнана як білоруська, що може свідчити про недостатню точність моделі для української мови або про схожість мов, яка іноді може викликати помилки в автоматичному розпізнаванні. Це також може бути пов'язано з якістю аудіозапису, присутністю музичного супроводу та інших шумів, які впливають на процес розпізнавання. Для більш точної роботи моделі з українською мовою можна спробувати використати більш потужну модель Whisper або додатково налаштувати обробку аудіо.

У інтерфейсі бібліотеки Whisper AI є можливість вказати мову аудіофайлу, що може підвищити точність розпізнавання мовлення. Це робиться шляхом встановлення параметра language у методі transcribe. Це

дозволяє моделі зосередитися на певній мові, покращуючи результати розпізнавання, особливо для мов, що можуть бути схожими.

```
def transcribe_audio(self, filename):
    language = self.language_combobox.get()
    if language:
        result = MODEL.transcribe(audio=filename, language=language)
    else:
        result = MODEL.transcribe(audio=filename)
    self.transcription_text.insert(tk.END, result['text'] + "\n")
```

Рисунок 4.6 – Код з вказанням мови аудіо файлу

Вказавши мову українською («uk»), можна очікувати кращих результатів розпізнавання для українських аудіофайлів. Після чого маємо наступний результат:

```
PS E:\code\python\whiperai> python main.py
 І повзалі не вучову, Ігор Кочі і Пурчі, відківзявся я не
, І розпитує вона. Хто ти човне, що шукаєш, Ткий і куди в
рярве, Скали грозить, Надють, просить, Ксобі вирий. Менне
апі, Кажу що природомати, Держить нас як їм дамтре, На вк
и шукаєш, Чопиробучу гості жидеж, Біх мівічний, Тож не зн
нне, Менне, Менне, Не один вто, Нуль тут човне, Дане кожд
н, у сваше ж с'яти, Може, Імен, Ну тубіся, Вадась до цілі
```

Рисунок 4.7 – Розпізнаний текст після вказування мови

Він свідчить про певні помилки та неточності у тексті. Це може бути викликано кількома факторами, такими як фонові шуми, музичний супровід та можливі обмеження самої моделі розпізнавання. Використання параметра «language="uk"» при розпізнаванні українських аудіофайлів могло покращити результати, але очевидно, що моделі все ще необхідна додаткова оптимізація для більш точного розпізнавання української мови. Для порівняння повний текст пісні у ДОДАТОК В.

Спробуємо записати і розпізнати фразу яка містить дві мови: українську і англійську.

```
PS E:\code\python\whiperai> python main.py
 Приклад тексту української мовию. 3 asample of attacks в інші лінішній лініші.
PS E:\code\python\whiperai>
```

Рисунок 4.8 – Розпізнаний текст з використанням двох мов

У результаті маємо наступне: «Приклад тексту українською мовою. 3 asample of attacks в інші лінійний лінії.» Це свідчить про те, що модель здатна розпізнавати багатомовні фрази, але можуть виникати певні неточності та змішування мов. Модель розпізнала частину англійського тексту, але з помилками, і зберегла український текст майже без змін.

Таке змішування може ускладнювати точність розпізнавання та потребувати додаткового налаштування та обробки для досягнення кращих результатів.

4.1.3 Висновки

Після проведення трьох тестів розпізнавання мовлення за допомогою моделі Whisper AI можна зробити кілька висновків. По-перше, при обробці англомовної пісні Майкла Джексона модель змогла розпізнати текст, але з певними неточностями, зумовленими музичним супроводом та шумами. Це свідчить про потребу в більш чистих аудіозаписах для досягнення кращих результатів.

По-друге, під час розпізнавання української пісні «Човен» гурту Один в каное, модель зіткнулася з труднощами, визначивши мову як білоруську. Вказівка конкретної мови допомогла покращити точність, але все ще були присутні певні помилки. Нарешті, тест з багатомовною фразою продемонстрував здатність моделі розпізнавати обидві мови, хоча й з деякими змішаннями і неточностями. Це вказує на можливість використання моделі для багатомовних задач, але з необхідністю додаткового налаштування для підвищення точності.

Якщо порівнювати точність розпізнавання української та англійської мов за допомогою моделі Whisper AI, можна зробити кілька спостережень. Модель продемонструвала вищу точність для англійської мови, що було

помітно при обробці пісні Майкла Джексона, де розпізнаний текст був більш читабельним і зрозумілим, хоча й містив деякі помилки через музичний супровід. Для української мови, навіть з вказанням параметра «language="uk"», точність була нижчою, з більшою кількістю помилок і неточностей. Це може свідчити про кращу оптимізацію моделі для англійської мови та потребу в додатковому налаштуванні або тренуванні для української мови.

4.2 Написання інтерфейсу для взаємодії

Для написання інтерфейсу буде використано бібліотеку Tkinter, яка є стандартним засобом для створення графічних інтерфейсів користувача (GUI) в Python. Tkinter дозволяє легко створювати вікна, кнопки, текстові поля та інші елементи інтерфейсу.

Для запису та збереження аудіо буде використовуватись бібліотека pyaudio, яка є популярною для роботи з аудіо в Python. Ця бібліотека дозволяє здійснювати запис звуку з мікрофону та зберігати його у файл.

4.2.1 Написання додатку

Для зручності користувача напишемо додаток з графічним інтерфейсом, який дозволятиме записувати аудіо, зберігати його та розпізнавати текст. Використовуватимемо бібліотеки Tkinter для інтерфейсу, PyAudio для запису звуку та Whisper для розпізнавання мовлення.

Розіб'ємо код на два класи: «AudioRecorder» для запису аудіо та «Application» для графічного інтерфейсу користувача. Це допоможе краще організувати код і зробити його більш читабельним та підтримуваним.

```
class Application(tk.Tk):
    def __init__(self):
        super().__init__()
        self.title("Аудіозаписувач та Транскрибатор")
        self.geometry("500x400")

        self.recorder = AudioRecorder()

        self.start_button = tk.Button(
            self, text="Почати запис", command=self.start_recording)
        self.start_button.pack(pady=10)

        self.stop_button = tk.Button(
            self, text="Зупинити запис", command=self.stop_recording)
        self.stop_button.pack(pady=10)

        self.language_label = tk.Label(
            self, text="Оберіть мову (необов'язково):")
        self.language_label.pack(pady=5)

        self.language_combobox = ttk.Combobox(
            self, values=["", "en", "uk", "es", "de", "fr"], state="readonly")
        self.language_combobox.pack(pady=5)
        self.language_combobox.current(0)  # За замовчуванням не обрана мова

        self.transcription_text = scrolledtext.ScrolledText(self, wrap=tk.WORD)
        self.transcription_text.pack(
            expand=True, fill=tk.BOTH, padx=10, pady=10)
```

Рисунок 4.9 – Код для інтерфейсу на Tkinter

Спочатку ініціалізується основне вікно програми та встановлюються його властивості, такі як заголовок і розміри. Далі додаються необхідні елементи інтерфейсу: кнопки для запуску та зупинки запису аудіо, випадаючий список для вибору мови транскрибування, а також текстове поле зі скролінгом для відображення результатів розпізнавання. Кожному елементу призначаються відповідні функції обробки подій, щоб взаємодіяти з користувачем та виконувати необхідні дії (запис аудіо, збереження, транскрибування).

```
def start_recording(self):
    self.recorder.start_recording()
    self.transcription_text.insert(tk.END, "Запис розпочато...\n")

def stop_recording(self):
    audio_filename = "recorded_audio.wav"
    self.recorder.stop_recording(audio_filename)
    self.transcription_text.insert(
        tk.END, "Запис зупинено. Транскрибування...\n")
    self.transcribe_audio(audio_filename)
    self.recorder.terminate_audio()

def transcribe_audio(self, filename):
        language = self.language_combobox.get()
        if language:
            result = MODEL.transcribe(audio=filename, language=language)
        else:
            result = MODEL.transcribe(audio=filename)
        self.transcription_text.insert(tk.END, result['text'] + "\n")
```

Рисунок 4.10 – Код кнопок інтерфесу

Функція start_recording запускається при натисканні кнопки «Почати запис». Вона ініціює метод start_recording об'єкта AudioRecorder, який починає процес запису аудіо з мікрофону. Після цього в текстове поле додається повідомлення про початок запису, інформуючи користувача про поточний стан процесу.

Функція stop_recording активується кнопкою «Зупинити запис». Вона зупиняє запис аудіо, зберігаючи записаний звук у файл під назвою recorded_audio.wav. Після зупинки запису текстове поле оновлюється повідомленням про зупинку та початок транскрибування. Потім викликається метод transcribe_audio для обробки аудіофайлу, а terminate_audio завершує роботу з аудіо інтерфейсом.

Функція transcribe_audio відповідає за розпізнавання мовлення з записаного аудіо. Вона отримує вибрану мову з випадаючого списку і виконує транскрибування за допомогою моделі Whisper. Якщо мова вказана, вона передається як параметр для покращення точності. Результат розпізнавання тексту додається до текстового поля, дозволяючи користувачу переглянути розпізнаний текст.

```
class AudioRecorder:

    def __init__(self):
        self.chunk = 1024  # Запис у блоках по 1024 зразки
        self.sample_format = pyaudio.paInt16  # 16 біт на зразок
        self.channels = 1  # Кількість каналів змінено на 1
        self.fs = 44100  # Запис з частотою 44100 зразків на секунду
        self.p = pyaudio.PyAudio()  # Створення інтерфейсу для PortAudio
        self.frames = []  # Ініціалізація масиву для зберігання кадрів
        self.stream = None
        self.is_recording = False

    def start_recording(self): ···

    def record(self): ···

    def stop_recording(self, filename): ···

    def terminate_audio(self): ···
```

Рисунок 4.11 – Код логіки рекордеру

Функція «__init__» ініціалізує об'єкт AudioRecorder, налаштовуючи параметри для запису аудіо, такі як формат зразків, кількість каналів, частоту

дискретизації, і створює інтерфейс для PortAudio. Вона також ініціалізує масив для зберігання кадрів аудіо та змінну для зберігання стану запису.

Функція start_recording починає процес запису аудіо. Вона ініціалізує масив кадрів, створює новий аудіопотік з параметрами, визначеними у функції «__init__», і запускає потік для запису аудіо, встановлюючи змінну стану запису у значення True.

Функція record визначає внутрішню функцію callback, яка здійснює запис аудіоданих у циклі, поки змінна стану запису є True. Вона зчитує аудіодані з потоку і додає їх до масиву кадрів, обробляючи можливі помилки введення/виведення.

Функція stop_recording зупиняє запис аудіо, встановлюючи змінну стану запису у значення False. Вона зупиняє і закриває аудіопотік, зберігаючи записані кадри у файл у форматі WAV. Після цього масив кадрів очищується.

Функція terminate_audio завершує роботу з інтерфейсом PortAudio, звільняючи всі ресурси, пов'язані з об'єктом PyAudio. Якщо об'єкт PyAudio існує, він закривається і змінна стану змінюється на None.

Загалом діаграма роботи коду буде виглядати наступним чином:

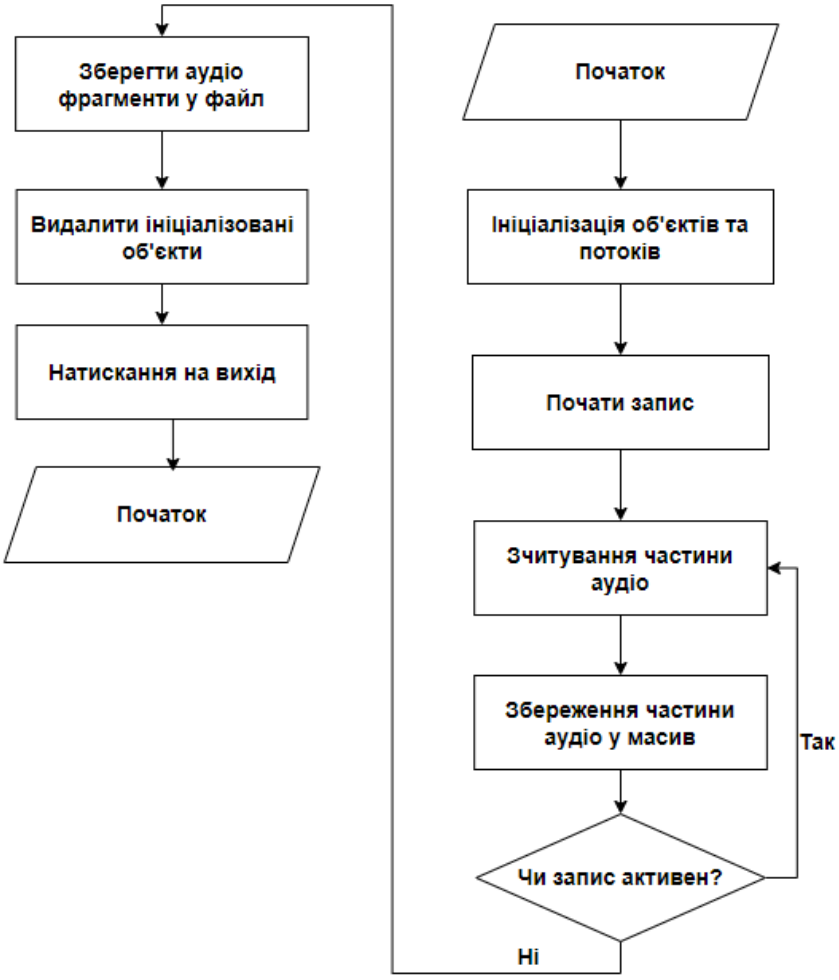


Рисунок 4.12 – Діаграма логіки додатку

На діаграмі проілюстровано процес запису аудіо та його збереження в файл, а також ініціалізацію та завершення аудіооб'єктів. На лівій частині діаграми показано етапи завершення запису: збереження аудіо фрагментів у файл, видалення ініціалізованих об'єктів та натискання на вихід. Кожен етап підписаний і з'єднаний стрілками, які вказують на послідовність дій.

Права частина діаграми демонструє процес запису аудіо: початок процесу, ініціалізація об'єктів та потоків, початок запису, зчитування частини аудіо і збереження її у масив. Процес повторюється в циклі, доки запис активний. Якщо запис неактивний, аудіофрагменти зберігаються у файл і процес завершується. Діаграма використовує овали для початку і кінця,

прямокутники для дій, та ромби для умовних операторів, що забезпечує чітке розуміння процесу.

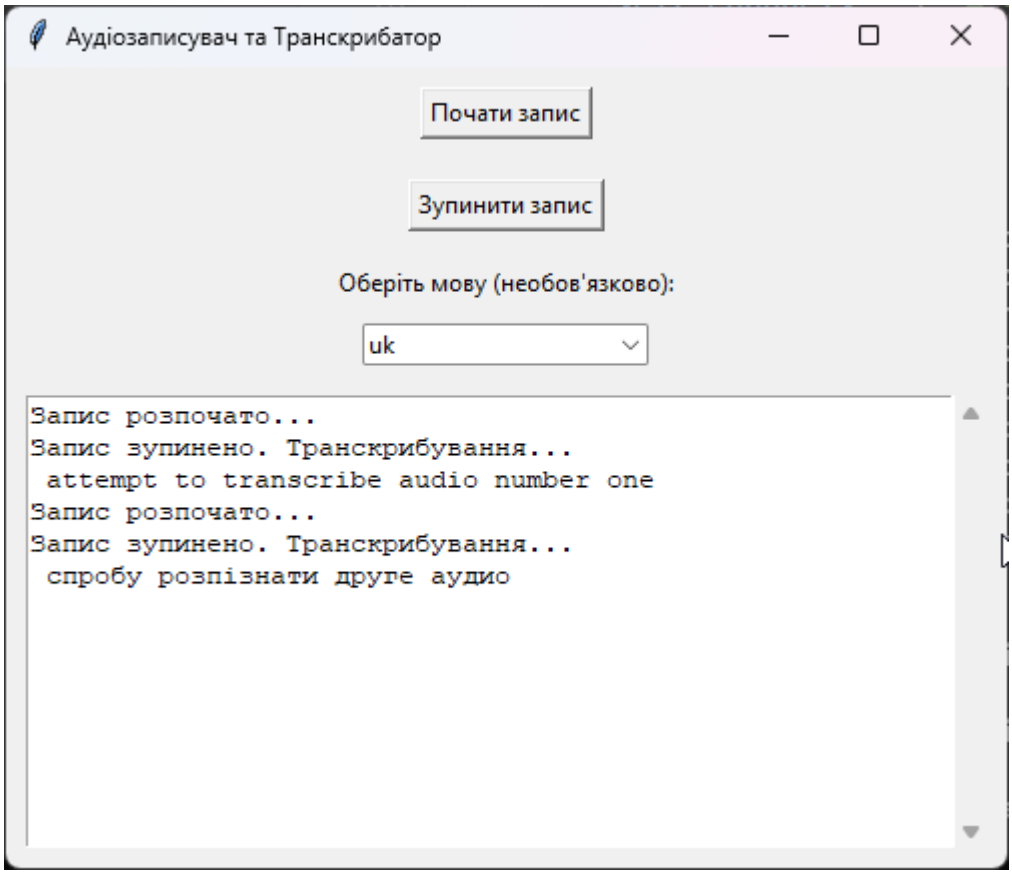


Рисунок 4.13 – Інтерфейс застосунку для розпізнання мовлення

На зображенні показано графічний інтерфейс програми «Аудіозаписувач та Транскрибатор», створеної за допомогою бібліотеки Tkinter. Вікно програми містить кнопки «Почати запис» та «Зупинити запис», які керують процесом запису аудіо. Нижче розташований випадаючий список для вибору мови транскрибування, що дозволяє покращити точність розпізнавання мовлення.

У нижній частині вікна розташоване текстове поле зі скролінгом, яке відображає повідомлення про стан запису та результати транскрибування. Наприклад, показано повідомлення про початок і зупинку запису, а також повідомлення про транскрибування аудіофайлів. Це дозволяє користувачу бачити процес у режимі реального часу та отримувати результати розпізнавання тексту.

4.2.2 Висновок

У результаті маємо функціональний додаток для запису та транскрибування аудіо, створений за допомогою бібліотек Tkinter, PyAudio та Whisper. Інтерфейс користувача забезпечує зручність у використанні, дозволяючи легко взаємодіяти з програмою. Користувач може почати і зупинити запис аудіо, вибрати мову транскрибування та переглянути розпізнаний текст. Кнопка «Почати запис» запускає функцію, яка починає запис аудіо, а кнопка «Зупинити запис» зупиняє його, зберігаючи файл та ініціюючи процес транскрибування.

Клас AudioRecorder відповідає за технічні аспекти запису аудіо, використовуючи PyAudio для створення аудіопотоків та збереження аудіофайлів. Він також забезпечує обробку аудіоданих у режимі реального часу, зчитуючи та зберігаючи фрагменти звуку у файл формату WAV. Завдяки реалізації потокової обробки, запис аудіо виконується ефективно і без перерв.

Функціональність транскрибування здійснюється за допомогою бібліотеки Whisper, яка розпізнає мовлення з аудіофайлів. Модель Whisper використовує передові технології глибокого навчання для точного розпізнавання тексту з аудіо. Інтерфейс дозволяє користувачу вибрати мову транскрибування, що покращує точність розпізнавання.

Додаток має інтуїтивно зрозумілий графічний інтерфейс, який дозволяє користувачам легко починати та зупиняти запис, обирати мову для транскрибування та переглядати результати у реальному часі. Це робить програму зручною та ефективною для широкого кола користувачів, надаючи потужні можливості для роботи з аудіоданими.

4.3 Пакування додатку

Для пакування додатку буде використано інструмент PyInstaller, який дозволяє створювати самостійні виконувані файли з Python-скриптів. Це зручно для розповсюдження програми, оскільки користувачам не потрібно встановлювати Python та необхідні бібліотеки окремо. PyInstaller збирає всі

залежності та ресурси у один виконуваний файл, що значно спрощує процес інсталяції та використання додатку на різних системах.

Команда «pyinstaller --onefile --windowed main.py» створить виконуваний файл з вашого скрипта main.py, який включатиме всі необхідні залежності та ресурси. Прапорець «--onefile» вказує на те, що всі файли повинні бути зібрані в один виконуваний файл, а прапорець «--windowed» вимикає консольне вікно (зручно для GUI-додатків).

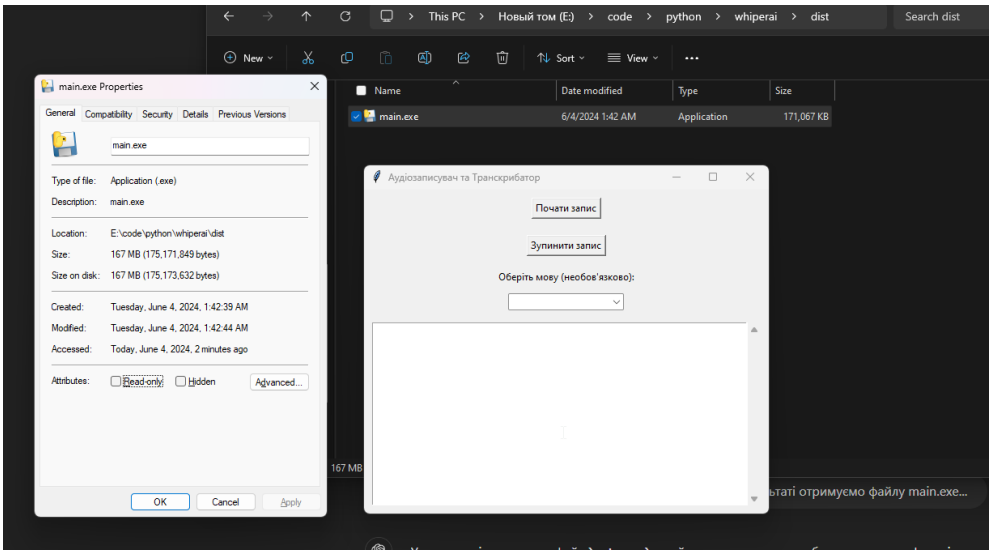


Рисунок 4.14 – Результати пакування додатку

У результаті отримуємо файл main.exe, який можна запускати на будь-якому комп'ютері з операційною системою Windows без необхідності встановлення Python та інших бібліотек. Використання PyInstaller забезпечує зручність у розповсюдженні програми, дозволяючи користувачам легко використовувати ваш додаток. Файл main.exe містить усі необхідні компоненти для коректної роботи програми, включаючи інтерфейс, функції запису та транскрибування аудіо. Це значно спрощує процес інсталяції та забезпечує стабільність роботи додатку.

Перевіримо роботу створеного файлу main.exe за допомогою Windows Sandbox. Windows Sandbox є ізольованим середовищем, яке дозволяє безпечно запускати та тестувати програми, не впливаючи на основну систему. Для цього потрібно скопіювати файл main.exe у середовище Windows Sandbox та

запустити його. Це дозволить перевірити, чи правильно працює додаток, чи всі функції виконуються коректно, і чи немає проблем з залежностями або середовищем виконання. Такий підхід забезпечує безпеку та ефективність тестування.

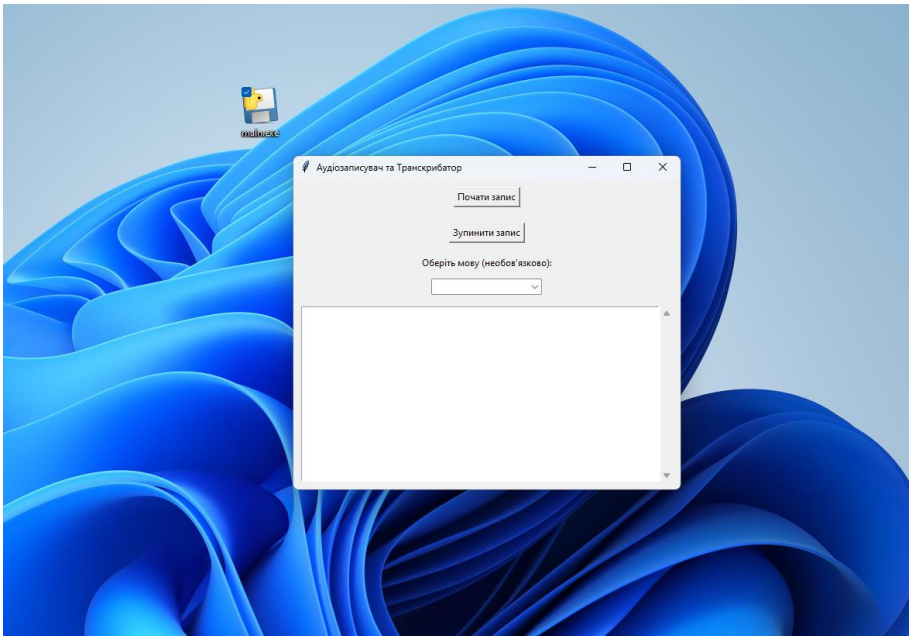
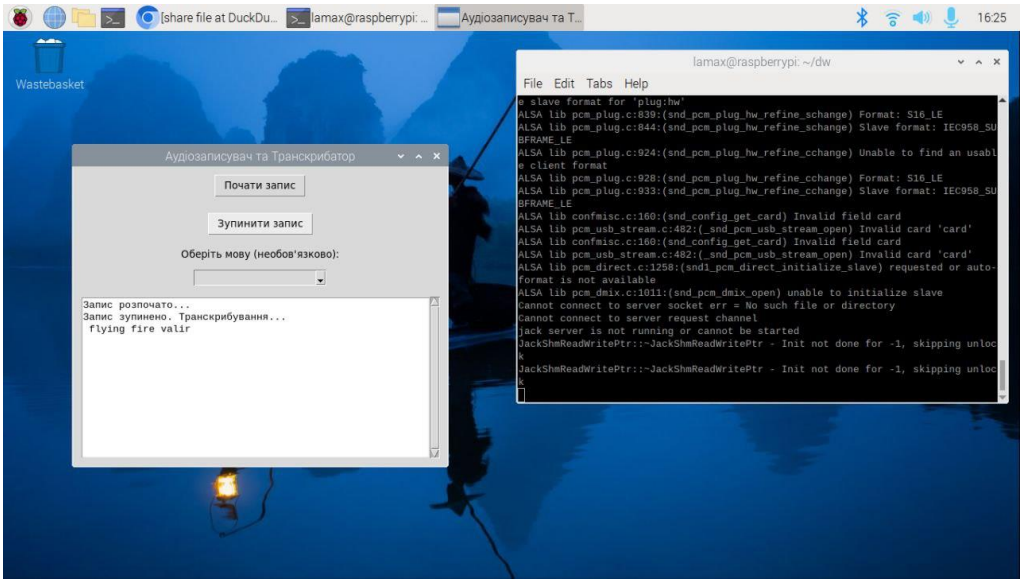


Рисунок 4.15 – Перевірка спакованого додатку у середовищі

Додаток запустився у середовищі Windows Sandbox, що видно на зображенні. Інтерфейс програми відображає кнопки «Почати запис» і «Зупинити запис», а також випадаючий список для вибору мови та текстове поле для відображення результатів транскрибування. Успішний запуск у Windows Sandbox підтверджує, що додаток функціонує коректно і готовий до використання, забезпечуючи зручність у записі та розпізнаванні аудіо.

Щоб зібрати додаток для виконання на системах ARM з ОС на базі Linux, потрібно використовувати інструменти, які підтримують компіляцію для архітектури ARM. Один із способів це зробити – скористатися PyInstaller на ARM-платформі, наприклад, на Raspberry Pi.



Р

И

С

У

Н

О

К

S

F

Q

Після зборки отримаємо виконуваний файл, який можна запускати на системах ARM з ОС на базі Linux. Цей файл міститиме всі необхідні залежності та ресурси, зібрані PyInstaller, що дозволяє запускати додаток без необхідності додаткового встановлення Python чи бібліотек. Виконуваний файл можна переносити на інші ARM-пристрої, де він буде функціонувати автономно, забезпечуючи той самий інтерфейс та функціональність для запису та транскрибування аудіо, як і на інших платформах.

Для спрощення процесу встановлення залежностей на системах Linux можна написати скрипт, який автоматично встановить Python, всі необхідні бібліотеки, PyTorch та Whisper. [14]

```
#!/bin/bash

# Оновлення системи та встановлення Python
sudo apt update -y
sudo apt upgrade -y
sudo apt install -y python3 python3-pip

# Встановлення PyAudio залежностей
sudo apt install -y portaudio19-dev python3-pyaudio

# Встановлення інших необхідних бібліотек
pip3 install tkinter scrolledtext

# Встановлення PyTorch
pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cpu

# Встановлення Whisper
pip3 install whisper

# Перевірка встановлення
echo "Перевірка встановлення:"
python3 -c "import tkinter; import pyaudio; import torch; import whisper; print('Всі залежності
встановлені успішно!')"
```

Рисунок 4.17 – Скрипт для втсановлення всіх залежностей

З

Для використання файл треба зробити виконуваним за допомогою команди chmod +x install.sh і запустити його командою «./install.sh» або «sudo sh ./install.sh», якщо виникають проблеми з правами.

Цей скрипт оновлює систему, встановлює Python та необхідні пакети, включаючи PyTorch та Whisper, [19] а також перевіряє успішність встановлення залежностей. Це дозволяє автоматизувати процес підготовки середовища для роботи з вашим додатком. Повний листінг коду скрипту знаходиться у ДОДАТОК Д, код для розпізнання мовленя знаходитться у ДОДАТОК Г.

Висновки до розділу

Алгоритм розпізнавання мовлення з використанням бібліотеки Whisper AI включає кілька етапів. Спочатку аудіосигнал обробляється для виділення його ключових характеристик. Ці характеристики передаються в нейронну мережу, яка розпізнає фонеми та слова, створюючи текстову інтерпретацію аудіофайлу. Whisper AI використовує передові методи глибокого навчання для забезпечення високої точності розпізнавання.

Розробка додатку для запису та транскрибування аудіо на Python включає створення інтуїтивно зрозумілого інтерфейсу користувача за допомогою бібліотеки Tkinter. Інтерфейс містить кнопки для початку і зупинки запису, випадаючий список для вибору мови транскрибування та текстове поле для відображення результатів. Це дозволяє користувачам легко взаємодіяти з додатком, забезпечуючи зручність у використанні.

Технічна частина запису аудіо реалізована класом AudioRecorder, який використовує PyAudio для створення аудіопотоків та збереження аудіо у файл формату WAV. Процес запису включає ініціалізацію аудіопотоку, зчитування даних у циклі та збереження їх у масив. Після зупинки запису дані зберігаються у файл.

Транскрибування аудіо здійснюється за допомогою бібліотеки Whisper. Модель Whisper використовує нейронні мережі для розпізнавання тексту з

аудіофайлів. Інтерфейс програми дозволяє вибрати мову транскрибування для підвищення точності розпізнавання. Результати розпізнавання відображаються у текстовому полі, що забезпечує зворотний зв'язок у реальному часі.

Для створення виконуваного файлу, що містить усі необхідні залежності, використовується інструмент PyInstaller. Команда «pyinstaller --onefile --windowed main.py» створює файл main.exe, який можна запускати на будь-якому комп'ютері з Windows без необхідності встановлення Python чи бібліотек. Це значно спрощує процес інсталяції та використання програми.

Після створення виконуваного файлу його можна перевірити у середовищі Windows Sandbox, яке забезпечує безпечне тестування без впливу на основну систему. Це дозволяє переконатися у коректній роботі програми та перевірити всі її основні функції.

Для пакування додатку для систем ARM з ОС на базі Linux використовується PyInstaller. Після встановлення необхідних бібліотек та залежностей команда створює виконуваний файл, який можна запускати на ARM-пристроях з Linux. Це забезпечує підтримку різних платформ.

У результаті розроблено ефективний додаток для запису та транскрибування аудіо з інтуїтивно зрозумілим інтерфейсом. Використання сучасних технологій глибокого навчання забезпечує високу точність розпізнавання мовлення. Додаток легко розповсюджувати та використовувати на різних платформах, що робить його зручним для широкого кола користувачів.

ВИСНОВКИ

Проєкт розпізнавання мовлення за допомогою Whisper AI включає кілька важливих етапів, починаючи з теоретичного огляду проблеми. Існуючі технології розпізнавання мовлення, такі як Google Speech-to-Text та Amazon Alexa, мають свої переваги та недоліки, проте Whisper AI виділяється завдяки високій точності та здатності працювати з різними мовами. Аналіз існуючих рішень дозволив визначити найефективніші підходи та методи, що стали основою для розробки нашого додатку.

На етапі розробки вимог до програмного та апаратного забезпечення було визначено, що додаток повинен бути сумісним з різними операційними системами, підтримувати багато мов та бути легким у використанні. Для цього були обрані відповідні технології, такі як Tkinter для створення інтерфейсу, PyAudio для запису аудіо та PyInstaller для пакування додатку. Вибір апаратної частини включав підтримку як x86, так і ARM архітектур, що дозволяє використовувати додаток на широкому спектрі пристроїв, включаючи Raspberry Pi.

Огляд програмної частини включав аналіз і вибір відповідних бібліотек та інструментів. Tkinter був обраний для створення інтуїтивно зрозумілого графічного інтерфейсу користувача. PyAudio забезпечив можливість запису аудіо у режимі реального часу, а Whisper AI, використовуючи технології глибокого навчання, дозволив досягти високої точності розпізнавання мовлення. Розробка програмної частини включала створення функціоналу для запису, збереження та транскрибування аудіо, а також інтеграцію цих функцій у єдиний інтерфейс.

Пакування додатку за допомогою PyInstaller дозволило створити виконувані файли, які можна легко розповсюджувати та запускати на різних платформах. Це значно спростило процес інсталяції для кінцевих користувачів, забезпечуючи їм можливість використовувати додаток без необхідності встановлення додаткових залежностей. Крім того, був створений скрипт для

автоматичної установки всіх необхідних бібліотек та залежностей, що забезпечує легкість і зручність у підготовці середовища для роботи з додатком.

Результатом проєкту став функціональний додаток для запису та транскрибування аудіо, який відзначається високою точністю розпізнавання мовлення завдяки використанню сучасних технологій глибокого навчання. Інтерфейс додатку забезпечує зручність у використанні, дозволяючи користувачам легко взаємодіяти з програмою. Підтримка різних операційних систем і архітектур робить додаток доступним для широкого кола користувачів.

Тестування додатку у середовищі Windows Sandbox підтвердило його стабільність та коректну роботу, забезпечуючи користувачам безпеку під час використання. Додаток дозволяє легко починати і зупиняти запис аудіо, обирати мову для транскрибування та отримувати результати у режимі реального часу. Це робить його корисним інструментом для різноманітних завдань, пов'язаних з розпізнаванням мовлення.

Завдяки розробленому скрипту для установки, процес підготовки середовища для роботи з додатком став максимально простим і автоматизованим. Це дозволяє швидко налаштувати систему для роботи з додатком без необхідності ручного встановлення всіх залежностей, що значно економить час і зусилля користувачів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Portable Sign Language Interpreter Device: пат. 202211067023 Індія. № 202211067023; заявл. 22.11.2022; опубл. 02.12.2022,

2. A Wearable Speech to Text Device for Hearing Impaired: пат. 202211031314 Індія. № 202211031314; заявл. 01.06.2022; опубл. 10.06.2022,

3. Energy Efficacy Using Raspberry Pi: пат. 201741023010 Індія. № 201741023010; заявл. 30.06.2017; опубл. 04.01.2019,

4. Industrial Automation Using IoT with Raspberry Pi: пат. 201721025628 Індія. № 201721025628; заявл. 19.07.2017; опубл. 13.10.2017,

5. Low Latency Audio Enhancement: пат. 3075738 Канада. № 3075738; заявл. 12.09.2018; опубл. 21.03.2019,

6. Experimental Speech Recognition System Based on Raspberry Pi 3. URL: https://www.researchgate.net/publication/317154253_Experimental_speech_recognition_system_based_on_Raspberry_Pi_3 (дата звернення: 10.05.2024).

7. Development of a Portable Speech Recognition System for Visually Impaired People using Raspberry Pi. URL: https://www.jardcs.org/abstract.php?id=365 (дата звернення: 10.05.2024).

8. A Review on Speech Recognition System Based on Raspberry Pi. URL: https://turcomat.org/index.php/turkbilmat/article/view/9919 (дата звернення: 10.05.2024).

9. Raspberry Pi Documentation. URL: https://www.raspberrypi.com/documentation/ (дата звернення: 10.05.2024).

10. What is Generative AI? URL: https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-generative-ai (дата звернення: 10.05.2024).

11. Definition of Generative AI. URL: https://www.techtarget.com/searchenterpriseai/definition/generative-AI (дата звернення: 10.05.2024).

12. IBM – Speech Recognition. URL: https://www.ibm.com/topics/speech-recognition (дата звернення: 10.05.2024).

13. Automatic Speech Recognition (ASR) System Development. URL: https://mobidev.biz/blog/automatic-speech-recognition-asr-system-development (дата звернення: 10.05.2024).

14. Raspberry Pi Guide. URL: https://raspberrypi-guide.github.io/ (дата звернення: 10.05.2024).

15. How to Turn Audio to Text Using OpenAI Whisper. URL: https://www.freecodecamp.org/news/how-to-turn-audio-to-text-using-openai-whisper/ (дата звернення: 10.05.2024).

16. Whisper: Robust and Efficient Speech Recognition at the Edge. URL: https://cdn.openai.com/papers/whisper.pdf (дата звернення: 10.05.2024).

17. Raspberry Pi Models Comparison. URL: https://socialcompare.com/en/comparison/raspberrypi-models-comparison (дата звернення: 10.05.2024).

18. How Does Speech Recognition Work? URL: https://www.lexacom.co.uk/how-does-speech-recognition-work/ (дата звернення: 10.05.2024).

19. BuiltIn – Artificial Intelligence. URL: https://builtin.com/artificial-intelligence (дата звернення: 10.05.2024).

20. Open Source Raspberry Pi Resources. URL: https://opensource.com/resources/raspberry-pi (дата звернення: 10.05.2024).

21. Мікрофон петличний Hoco L14 Type-C Black. URL: https://rozetka.com.ua/ua/412759164/p412759164/ (дата звернення: 26.05.2024).

22. Монітор HP 24M (3WL46AA). URL: https://hard.rozetka.com.ua/ua/311097403/p311097403/ (дата звернення: 26.05.2024).

23. Мікрокомп'ютер Raspberry Pi 5 Board 8 ГБайт. URL: https://evo.net.ua/mikrokompiuter-raspberry-pi-5-board-8ГБайт/ (дата звернення: 26.05.2024).

24. YouTube. Understanding Transformers: An Animation by 3Blue1Brown. URL: https://youtu.be/u_xNvC9PpHA (дата звернення: 28.05.2024)

25. OpenAI Whisper. URL: https://openai.com/index/whisper/ (дата звернення: 28.05.2024)

26. Statistical Learning from a Regression Perspective by Richard Berk. URL: https://link.springer.com/book/10.1007/978-1-4471-5779-3 (дата звернення: 28.05.2024)

27. Speech and Language Processing (3rd edition draft) by Dan Jurafsky and James H. Martin. URL: https://stanford.edu/~jurafsky/slp3/ (дата звернення: 28.05.2024)

ДОДАТОК А

Довідка

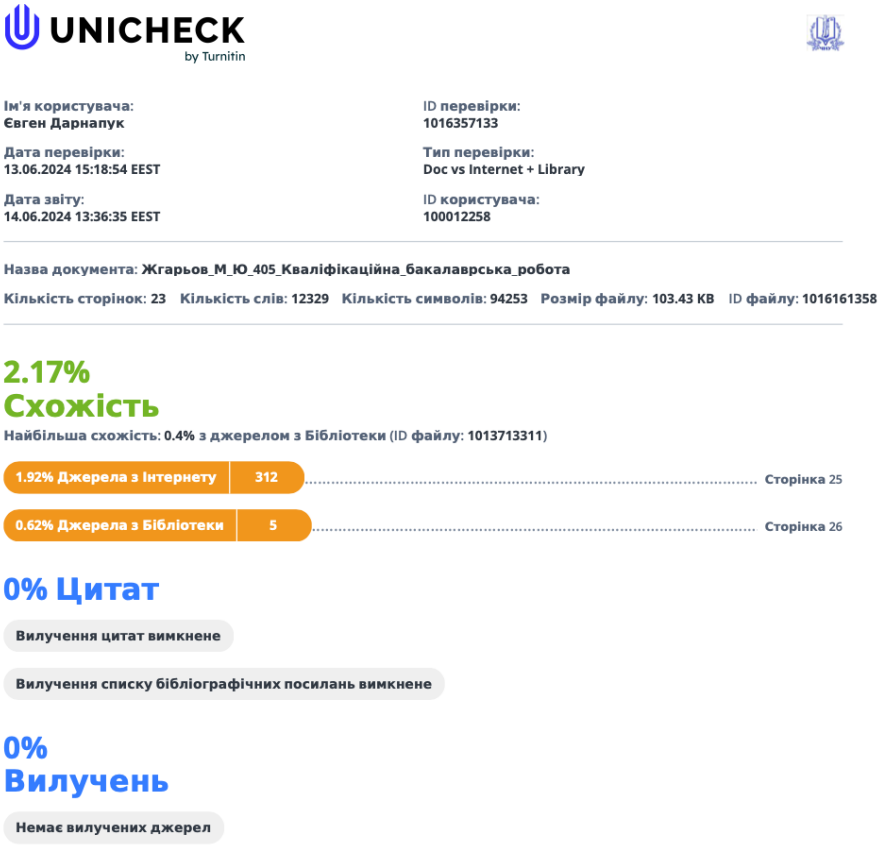
про перевірку на унікальність пояснювальної записки

бакалаврської кваліфікаційної роботи на тему:
«Система розпізнавання мовлення на базі Raspberry Pi та Whisper AI»

студента спеціальності 123 «Комп'ютерна інженерія», 405групи
Жгарьов Максим Юрійович
прізвище, ім'я, по-батькові

Перевірку тексту здійснено сервісом: онлайн-сервіс Unicheck

Результат перевірки тексту бакалаврської кваліфікаційної роботи: схожість складає 2.17%.


UNICHECK
by Turnitin

Ім'я користувача:
Євген Дарнапук

ID перевірки:
1016357133

Дата перевірки:
13.06.2024 15:18:54 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
14.06.2024 13:36:35 EEST

ID користувача:
100012258

Назва документа: Жгарьов_М_Ю_405_Кваліфікаційна_бакалаврська_робота

Кількість сторінок: 23 Кількість слів: 12329 Кількість символів: 94253 Розмір файлу: 103.43 KB ID файлу: 1016161358

2.17%
Схожість

Найбільша схожість: 0.4% з джерелом з Бібліотеки (ID файлу: 1013713311)

1.92% Джерела з Інтернету 312 Сторінка 25

0.62% Джерела з Бібліотеки 5 Сторінка 26

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Здобувач:

_____ М. Ю. Жгарьов
підпис ініціали, прізвище

Керівник:
ст. викладач
_____ Є. С. Дарнапук
підпис ініціали, прізвище

Дата: «___» _____ 2024 р.

ДОДАТОК Б

Текст для тестування розпізнання англійської мови

All I want to say is that they don't really care about us
Don't worry what people say, we know the truth
All I want to say is that they don't really care about us
Enough is enough of this garbage
All I want to say is that they don't really care about us
Skin head, dead head
Everybody gone bad
Situation aggravation
Everybody, allegation
In the suite on the news
Everybody, dog food
Bang-bang, shock dead
Everybody's gone mad
All I wanna say is that they don't really care about us
All I wanna say is that they don't really care about us
Beat me, hate me
You can never break me
Will me, thrill me
You can never kill me
Jew me, sue me
Everybody, do me
Kick me, kike me
Don't you black or white me
All I wanna say is that they don't really care about us
All I wanna say is that they don't really care about us
Tell me what has become of my life
I have a wife and two children who love me
I'm a victim of police brutality, now (Mhhm)

I'm tired of bein' the victim of hate

Your rapin' me of my pride

Oh, for God's sake

I look to heaven to fulfill its prophecy...

Set me free

Skin head, dead head

Everybody, gone bad

Trepidation speculation

Everybody, allegation

In the suite on the news

Everybody, dog food

Black man, black mail

Throw the brother in jail

All I wanna say is that they don't really care about us

All I wanna say is that they don't really care about us

Tell me what has become of my rights

Am I invisible 'cause you ignore me?

Your proclamation promised me free liberty, now

I'm tired of bein' the victim of shame

They're throwin' me in a class with a bad name

I can't believe this is the land from which I came

You know I really do hate to say it

The government don't wanna see

But it Roosevelt was livin', he wouldn't let this be, no, no

Skinhead, deadhead

Everybody, gone bad

Situation, speculation

Everybody, litigation

Beat me, bash me

You can never trash me

Hit me, kick me

You can never get me

All I wanna say is that they don't really care about us

All I wanna say is that they don't really care about us

Some things in life they just don't wanna see (Ah)

But if Martin Luther was livin'

He wouldn't let this be, no, no

Skinhead, deadhead (Yeah, yeah)

Everybody's gone bad

Situation, segregation (Woo-hoo)

Everybody, allegation

In the suite on the news

Everybody dog food (Woo-ho)

Kick me, kike me

Don't you wrong or right me

All I wanna say is that they don't really care about us

All I wanna say is that they don't really care about us

All I wanna say is that they don't really care about us

All I wanna say is that they don't really care about us

All I wanna say is that they don't really care about us

All I wanna say is that they don't really care about us

ДОДАТОК В

Текст для тестування розпізнання української мови

І повзе ліниво човен, і воркоче, і бурчить:
Відки взявся я – не знаю; чим прийдеться закінчить
Хвиля радісно плюскоче та леститься до човна,
Мов диття, цікава, шепче і розпитує вона:
Хто ти, човне? Що шукаєш? Відки і куди пливеш?
І за чим туди шукаєш? Що пробув? Чого ще ждеш?
Біг мій вічний – тож не знаю. Хвиля носить, буря рве,
Скали грозять, надять-просять к собі береги мене.
Що ж тут думать, що тужити, що питатися про ціль!
Нині жити, завтра гнити, нині страх, а завтра біль.
Кажуть, що природа – мати держить нас, як їм там тре,
А в кінці мене цілого знов до себе відбере.
Хто ти, човне? Що шукаєш? Відки і куди пливеш?
І за чим туди шукаєш? Що пробув? Чого ще ждеш?
Біг мій вічний – тож не знаю. Хвиля носить, буря рве,
Скали грозять, надять-просять к собі береги мене.
Не один втонув тут човен, та не кождий же втонув;
Хоч би й дев'ять не вернуло, то десятий повернув
А хто знає, може, в бурю іменно спасешся ти!
Може, іменно тобі ся вдасть до цілі доплисти!
Хто ти, човне? Що шукаєш? Відки і куди пливеш?
І за чим туди шукаєш? Що пробув? Чого ще ждеш?
Біг мій вічний – тож не знаю. Хвиля носить, буря рве,
Скали грозять, надять-просять к собі береги мене.

ДОДАТОК Г

Листінг коду програми

```
import tkinter as tk
from tkinter import scrolledtext, ttk
import pyaudio
import wave
import threading
from whisper import load_model
from warnings import simplefilter

# Завантаження моделі Whisper
simplefilter("ignore")
MODEL = load_model("base")



class AudioRecorder:
    def __init__(self):
        self.chunk = 1024  # Запис у блоках по 1024 зразки
        self.sample_format = pyaudio.paInt16  # 16 біт на зразок
        self.channels = 1  # Кількість каналів змінено на 1
        self.fs = 44100  # Запис з частотою 44100 зразків на секунду
        self.p = pyaudio.PyAudio()  # Створення інтерфейсу для PortAudio
        self.frames = []  # Ініціалізація масиву для зберігання кадрів
        self.stream = None
        self.is_recording = False

    def start_recording(self):
        self.frames = []
        if self.p is None:
            self.p = pyaudio.PyAudio()
```

```
    self.stream = self.p.open(format=self.sample_format,
                              channels=self.channels,
                              rate=self.fs,
                              frames_per_buffer=self.chunk,
                              input=True)
    self.is_recording = True
    self.record()

def record(self):
    def callback():
        while self.is_recording:
            try:
                data = self.stream.read(self.chunk)
                self.frames.append(data)
            except IOError as e:
                print(f"Recording error: {e}")
                self.is_recording = False

    self.thread = threading.Thread(target=callback)
    self.thread.start()

def stop_recording(self, filename):
    if self.is_recording:
        self.is_recording = False
        self.thread.join()
        if self.stream is not None:
            self.stream.stop_stream()
            self.stream.close()
            self.stream = None
```

```
            with wave.open(filename, 'wb') as wf:
                wf.setnchannels(self.channels)
                wf.setsampwidth(self.p.get_sample_size(self.sample_format))
                wf.setframerate(self.fs)
                wf.writeframes(b".join(self.frames))
            self.frames = []

    def terminate_audio(self):
        if self.p is not None:
            self.p.terminate()
            self.p = None


class Application(tk.Tk):
    def __init__(self):
        super().__init__()
        self.title("Аудіозаписувач та Транскрибатор")
        self.geometry("500x400")

        self.recorder = AudioRecorder()

        self.start_button = tk.Button(
            self, text="Почати запис", command=self.start_recording)
        self.start_button.pack(pady=10)

        self.stop_button = tk.Button(
            self, text="Зупинити запис", command=self.stop_recording)
        self.stop_button.pack(pady=10)

        self.language_label = tk.Label(
```

```
            self, text="Оберіть мову (необов'язково):")
        self.language_label.pack(pady=5)

        self.language_combobox = ttk.Combobox(
            self, values=["", "en", "uk", "es", "de", "fr"], state="readonly")
        self.language_combobox.pack(pady=5)
        self.language_combobox.current(0)  # За замовчуванням не обрана мова

        self.transcription_text = scrolledtext.ScrolledText(self, wrap=tk.WORD)
        self.transcription_text.pack(
            expand=True, fill=tk.BOTH, padx=10, pady=10)

    def start_recording(self):
        self.recorder.start_recording()
        self.transcription_text.insert(tk.END, "Запис розпочато...\n")

    def stop_recording(self):
        audio_filename = "recorded_audio.wav"
        self.recorder.stop_recording(audio_filename)
        self.transcription_text.insert(
            tk.END, "Запис зупинено. Транскрибування...\n")
        self.transcribe_audio(audio_filename)
        self.recorder.terminate_audio()

    def transcribe_audio(self, filename):
        language = self.language_combobox.get()
        if language:
            result = MODEL.transcribe(audio=filename, language=language)
```

```
        else:
            result = MODEL.transcribe(audio=filename)
        self.transcription_text.insert(tk.END, result['text'] + "\n")


if __name__ == "__main__":
    app = Application()
    app.mainloop()
```

ДОДАТОК Д

Листінг коду сценарію для встановлення залежностей

```
#!/bin/bash

# Оновлення системи та встановлення Python
sudo apt update -y
sudo apt upgrade -y
sudo apt install -y python3 python3-pip

# Встановлення PyAudio залежностей
sudo apt install -y portaudio19-dev python3-pyaudio

# Встановлення інших необхідних бібліотек
pip3 install tkinter scrolledtext

# Встановлення PyTorch
pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cpu

# Встановлення Whisper
pip3 install whisper

# Перевірка встановлення
echo "Перевірка встановлення:"
python3 -c "import tkinter; import pyaudio; import torch; import whisper; print('Всі залежності встановлені успішно!')"
```