КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

ФАКУЛЬТЕТ РАДІОФІЗИКИ, ЕЛЕКТРОНІКИ ТА КОМП'ЮТЕРНИХ СИСТЕМ

Кафедра радіотехніки та радіоелектронних систем

«На правах рукопису»

Робота допущена до захисту в ЕК
рішенням кафедри радіотехніки та радіоелектронних систем
від 20 травня 2024 року, протокол № 111.
Завідувач кафедри доктор фіз.-мат. наук, професор
_____ Ігор АНІСІМОВ

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему:

«Розробка персонального голосового асистента»

Виконав:
студент 4-го курсу
денної форми навчання
спеціальності 172 - Телекомунікації та радіотехніка
ОПП «Інформаційна безпека телекомунікаційних систем та мереж»
Фалюш Володар Сергійович _____

Науковий керівник:
Асистент кафедри радіотехніки та
радіоелектронних систем, кандидат
фізико-математичних наук
Котов Михайло Миколайович_____

Рецензент:
Доцент кафедри комп'ютерної інженерії
кандидат фізико-математичних наук
Загороднюк Сергій Петрович_____

Засвідчую, що у цій бакалаврській роботі
немає запозичень з праць інших авторів без
відповідних посилань
Студент ___ Фалюш Володар

Київ – 2024

РЕФЕРАТ

Дипломна робота: 30 сторінок, 8 рисунків, 8 джерел.

Ключові слова: ГОЛОСОВИЙ АСИСТЕНТ, NLP, РОЗПІЗНАВАННЯ МОВИ, СИНТЕЗ МОВИ, ІНТЕГРАЦІЯ З ХМАРНИМИ СЕРВІСАМИ, БЕЗПЕКА ДАНИХ.

Об’єкт розроблення – персональний голосовий асистент, здатний розуміти та виконувати запити користувачів, забезпечуючи високий рівень точності та персоналізації.

Мета роботи – розробка власного голосового помічника, здатного інтегруватися з розумними пристроями та сервісами, підтримувати багатомовність та забезпечувати високу конфіденційність даних.

Методи дослідження – системний аналіз, моделювання, методи розпізнавання мовлення, методи синтезу мовлення, аналіз даних, програмування, тестування та оптимізація систем.

Результати роботи – створений голосовий асистент, який використовує методи обробки природної мови (NLP) для розпізнавання та синтезу мовлення, інтегрується з різноманітними хмарними сервісами для виконання завдань, таких як управління розумними пристроями, надання інформаційних послуг, організація розкладу та багато іншого. Асистент забезпечує високу точність розпізнавання мовлення, швидкий час відповіді та високий рівень безпеки даних користувача.

Практичне значення отриманих результатів – розроблений голосовий асистент може бути використаний в різних сферах, таких як розумні будинки, автомобілі, мобільні пристрої, офіси та інші, де необхідна інтерактивна взаємодія з користувачем. Високий рівень персоналізації дозволяє налаштовувати асистента під конкретні потреби користувача, що робить його незамінним інструментом у повсякденному житті.

ЗМІСТ

РЕФЕРАТ **2**

ЗМІСТ **3**

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ **4**

ВСТУП **5**

1. ТЕХНОЛОГІЧНІ ОСНОВИ **7**

1.2. Принципи розпізнавання мови **11**

1.2.1. Акустичні моделі **11**

1.2.2. Лінгвістичні моделі **12**

1.3. Синтез мови **12**

1.4. Обробка природної мови (NLP) **14**

2. АРХІТЕКТУРА СИСТЕМИ ГОЛОСОВОГО ПОМІЧНИКА **16**

2.1. Модульна структура **16**

2.1.1. Вхідний модуль (розпізнавання мови) **16**

2.1.2. Модуль обробки даних (NLP) **17**

2.1.3. Вихідний модуль (синтез мови) **17**

2.2. Інтеграція з хмарними сервісами **17**

2.3. Безпека і конфіденційність даних **18**

3. ТЕХНІЧНІ ЗАСОБИ І ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ **19**

3.1. Програмне забезпечення **19**

3.1.1. Бібліотеки для розпізнавання мови (CMU Sphinx, Google Speech API) **19**

3.1.2. Інструменти для обробки мови (NLTK, spaCy) **20**

3.1.3. Платформи для синтезу мови (Google TTS, Amazon Polly) **20**

4. РОЗРОБКА ТА ТЕСТУВАННЯ **21**

4.1. Перша версія помічника **21**

4.2. Друга версія **24**

4.3. Нова третя версія **26**

4.4. Наступні поліпшення **27**

ВИСНОВОК **29**

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ **30**

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

ASR - Automatic Speech Recognition (Автоматичне розпізнавання мови)

NLP - Natural Language Processing (Обробка природної мови)

TTS - Text-to-Speech (Перетворення тексту в мовлення)

API - Application Programming Interface (Інтерфейс програмування додатків)

GPT - Generative Pre-trained Transformer (Генеративний попередньо навчений трансформер)

AWS - Amazon Web Services (Веб-сервіси Амазон)

HMM - Hidden Markov Model (Прихована модель Маркова)

RNN - Recurrent Neural Network (Рекурентна нейронна мережа)

CMU Sphinx - система для автоматичного розпізнавання мови, розроблена в університеті Карнегі-Меллон.

spaCy - бібліотека для обробки природної мови.

NLTK - Natural Language Toolkit, набір бібліотек для роботи з природною мовою.

Вступ

Голосові помічники стали значною частиною сучасних технологій, вони змінили спосіб взаємодії людей з комп'ютерами та інформаційними системами. Це програмне забезпечення, яке використовує обробку природної мови для виконання завдань і надання інформації за допомогою голосових команд. Голосові помічники вбудовані в смартфони, колонки, автомобілі та інші пристрої та мають простий, інтуїтивно зрозумілий інтерфейс для користувачів.

Ранні версії голосових помічників мали обмежену кількість функціональних можливостей, головним чином виконуючи прості завдання, як-от відтворення музики, налаштування нагадувань і надання інформації про погоду. Однак швидкий розвиток розпізнавання мовлення та штучного інтелекту призвів до того, що сучасні голосові помічники мають більшу потужність і різноманітність. Вони мають можливість керувати розумним будинком, купувати речі, призначати зустрічі, надавати персоналізовані рекомендації тощо.

Такі популярні голосові помічники, як Alexa, Siri та Google, тепер є невід'ємною частиною повсякденного життя. Вони часто використовуються в реальному світі, це допомагає скоротити час і зусилля. Крім того, голосові помічники є каталізатором розвиток технології розумного будинку. Ця технологія дозволяє користувачам керувати освітленням, температурою, безпекою та іншими аспектами свого будинку за допомогою голосових команд.

Однією з головних переваг голосових помічників є їхня здатність розуміти природну мову та реагувати на неї, що робить їх використання інтуїтивно зрозумілим навіть для людей, які не мають спеціального технічного досвіду. Це відкриває нові можливості для охоплення технологій, які можуть використовуватися більшою кількістю користувачів, щоб користуватися перевагами цифрової ери.

Голосові помічники мають конкретні цілі для розвитку, зокрема покращення точності розпізнавання мовлення, збереження конфіденційності даних і покращення функціональності. Подолання цих проблем має вирішальне значення

для створення ефективніших і ефективніших систем, які стануть важливими для нашого повсякденного життя.

Отже вони представляють собою захоплюючу область досліджень та розробок, яка має великий потенціал для подальшого вдосконалення та інтеграції в різні аспекти нашого життя. Розуміння їхніх можливостей та обмежень є ключовим для успішного впровадження та використання цих технологій у майбутньому.

Метою даного дипломного проекту є розробка власного голосового помічника, здатного розуміти та виконувати запити користувачів, забезпечуючи високий рівень точності та персоналізації. Проект передбачає створення системи, яка в подальшому зможе інтегруватися з різноманітними розумними пристроями та сервісами, підтримувати багатомовність та забезпечувати високу конфіденційність даних. Успішна реалізація цього проекту дозволить значно спростити взаємодію користувачів з технологіями, зробивши їх більш доступними та зручними у повсякденному житті.

1. ТЕХНОЛОГІЧНІ ОСНОВИ

1.1. Сучасні голосові помічники

Голосові помічники, такі як Алекса, Сірі та Google Assistant, стали невід'ємною частиною нашого повсякденного життя. Вони допомагають виконувати різні завдання за допомогою голосових команд, значно спрощуючи взаємодію з технологіями. Огляд їхнього розвитку та сучасних можливостей допоможе краще зрозуміти їхні принципи роботи та особливості.

Amazon Alexa була запущена у 2014 році, спочатку інтегрована в пристрій Amazon Echo. Перші версії Алекси могли виконувати базові команди, такі як відтворення музики, встановлення будильників, надання інформації про погоду та новини. Сучасні можливості Алекси включають управління розумним домом, виконання складніших завдань, таких як здійснення покупок та бронювання квитків, а також інтеграцію з тисячами сторонніх навичок, що дозволяє розширювати функціонал. Особливості Алекси включають відкриту платформу для розробників (Alexa Skills Kit), інтеграцію з великою кількістю розумних пристроїв та високу точність розпізнавання голосу завдяки постійному вдосконаленню технологій.

Рисунок 1.1. Станції Alexa

Apple Siri була запущена у 2011 році як частина iOS. Початково вона могла виконувати базові завдання, такі як надсилання повідомлень, встановлення нагадувань та пошук інформації в інтернеті. Сучасні можливості Сірі включають глибоку інтеграцію з екосистемою Apple (iPhone, iPad, Mac, Apple Watch), управління розумним домом через Apple HomeKit та використання машинного навчання для розуміння контексту і надання більш персоналізованих відповідей. Особливості Сірі включають високу ступінь приватності даних користувачів, підтримку багатьох мов і діалектів, а також інтеграцію з усіма сервісами Apple, що робить її незамінним помічником для користувачів продукції Apple.

Рисунок. 1.2 Інтерфейс Siri

Google Assistant був запущений у 2016 році як частина програми Google Now. Перші версії цього помічника виконували базові функції, такі як пошук інформації, встановлення нагадувань та відтворення музики. Сучасні можливості Google Assistant включають глибоку інтеграцію з сервісами Google (Gmail, Google Calendar, Google Maps), виконання складніших завдань, таких як бронювання ресторанів через Google Duplex, та підтримку багатьох мов з можливістю вести діалоги природною мовою. Особливості Google Assistant включають потужні алгоритми машинного навчання та штучного інтелекту, високу точність розпізнавання голосу та розуміння контексту, а також широку інтеграцію з пристроями на базі Android та розумними пристроями.

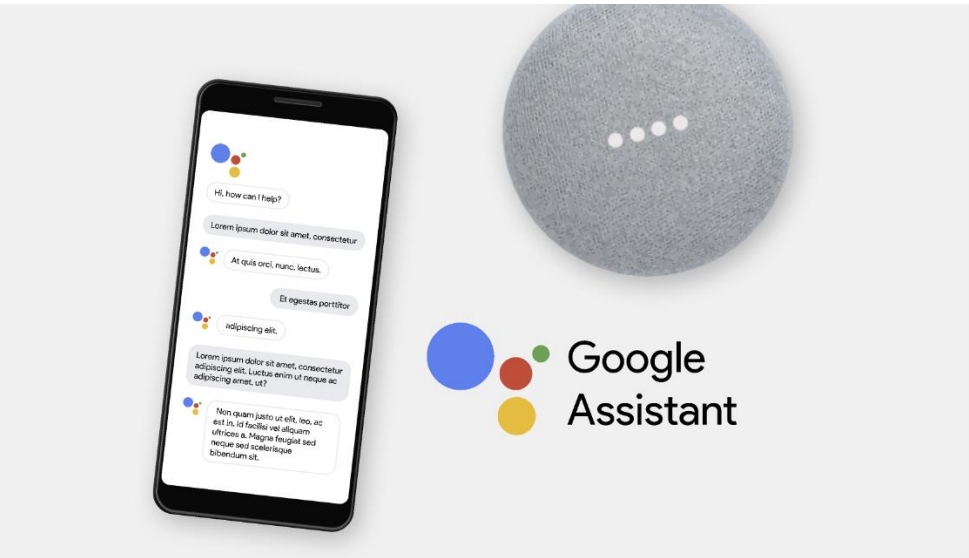

Рисунок. 1.3 Інтерфейс Google Assistant

Загальні переваги голосових помічників включають зручність, швидкість, мобільність та персоналізацію. Вони дозволяють виконувати завдання без необхідності ручного введення даних, миттєво реагують на команди, доступні на різних пристроях та можуть запам'ятовувати уподобання користувача для надання персоналізованих рекомендацій.

Принцип роботи голосових помічників базується на розпізнаванні мови, обробці природної мови (NLP) та синтезі мови. Розпізнавання мови здійснюється за допомогою акустичних моделей, що перетворюють голос у текст. Обробка природної мови аналізує текст для визначення інтенцій користувача, а синтез мови перетворює текст у природну мову для надання відповідей. Навчання моделі здійснюється на великих обсягах даних, що постійно вдосконалює точність розпізнавання та розуміння.

Попри значні досягнення у сфері розпізнавання та обробки мови, сучасні голосові помічники мають певні недоліки. Однією з головних проблем є помилки розпізнавання мови. Голосові помічники можуть мати труднощі з розпізнаванням акцентів або діалектів, що призводить до помилок у виконанні команд. Робота у шумному середовищі також знижує точність розпізнавання мови, оскільки мікрофони можуть вловлювати сторонні звуки.

Ще однією суттєвою проблемою є обмежене розуміння контексту. Голосові помічники часто не можуть адекватно обробляти складні або багатозначні запити, що вимагають глибокого розуміння контексту. Тривалі діалоги або розмови з кількома учасниками можуть створювати проблеми для помічників, оскільки вони не завжди можуть визначити, хто саме говорить або на що слід реагувати.

Питання приватності і безпеки також викликають занепокоєння у користувачів. Деякі користувачі стурбовані тим, що голосові помічники можуть записувати розмови без їхнього відома. Голосові помічники зберігають дані про користувача, що може становити ризик витоку конфіденційної інформації.

Обмежена функціональність є ще одним недоліком голосових помічників. Не всі голосові помічники можуть інтегруватися з усіма сторонніми сервісами або пристроями, а підтримка мов і локальних сервісів може бути обмеженою в певних регіонах. Більшість голосових помічників вимагають постійного підключення до інтернету для виконання своїх функцій, що значно обмежує їх можливості в умовах відсутності інтернету.

Ще однією важливою проблемою є відсутність емоційного інтелекту у голосових помічників. Вони не здатні точно розпізнавати емоційний стан користувача, що може впливати на якість взаємодії. Крім того, для забезпечення високої точності розпізнавання та обробки мови потрібні потужні апаратні засоби, що може бути недоступним для деяких користувачів.

Окремо варто згадати етичні питання, пов'язані з використанням голосових помічників. Існує ризик використання голосових помічників для збору даних з метою таргетованої реклами, а також можливість скорочення робочих місць у певних секторах через автоматизацію процесів.

Попри ці вади, голосові помічники продовжують розвиватися та вдосконалюватися. Вирішення цих проблем лежить у сфері покращення технологій розпізнавання мови, забезпечення більшої конфіденційності даних, а також розвитку етичних стандартів використання. Це допоможе зробити голосових помічників більш надійними та корисними у повсякденному житті.

Принцип роботи голосових помічників базується на розпізнаванні мови, обробці природної мови (NLP) та синтезі мови. Розпізнавання мови здійснюється за допомогою акустичних моделей, що перетворюють голос у текст. Обробка природної мови аналізує текст для визначення запиту користувача, а синтез мови перетворює текст у природну мову для надання відповідей. Навчання моделі здійснюється на великих обсягах даних, що постійно вдосконалює точність розпізнавання та розуміння.

1.2. Принципи розпізнавання мови

Акустичні та лінгвістичні моделі є фундаментальними компонентами систем розпізнавання мови, забезпечуючи перетворення звукових сигналів у зрозумілі слова та фрази. Вони використовуються у багатьох сучасних системах розпізнавання мови, включаючи голосові помічники (Amazon Alexa, Google Assistant, Apple Siri), та інших пристроях. Забезпечуючи високу точність і ефективність розпізнавання мови, дозволяючи користувачам взаємодіяти з технологіями за допомогою природної мови.

Синтаксичний аналіз визначає граматичну структуру речення, що дозволяє системі зрозуміти, як слова взаємодіють між собою в реченні. Семантичний аналіз аналізує зміст слів і фраз для визначення їхнього значення в контексті, допомагаючи системі розпізнавання мови зрозуміти сенс сказаного і відповісти відповідно. Контекстний аналіз використовує контекст для підвищення точності розпізнавання і розуміння мови, враховуючи попередні запити користувача, його звички і уподобання.

1.2.1. Акустичні моделі

Акустичні моделі починають свій робочий процес із захоплення звуку, де голосові команди записуються за допомогою мікрофонів, що перетворюють звукові хвилі у електричні сигнали. Потім ці сигнали оцифровуються з використанням частоти дискретизації, зазвичай 16 кГц, що забезпечує достатню

якість для розпізнавання мови. На етапі попередньої обробки звукові дані обробляються для поліпшення якості сигналу через нормалізацію рівня, шумозаглушення та виділення ключових ознак звуку, таких як мел-кепстральні коефіцієнти (MFCC). Завершується цей процес акустичним моделюванням, де приховані Марковські моделі (HMM) створюють статистичні моделі для кожної фонеми, описуючи ймовірність того, що певна послідовність звукових ознак відповідає певній фонемі. Це дозволяє обробляти варіативність у вимові та шуми, що присутні в оточенні.

1.2.2. Лінгвістичні моделі

Лінгвістичні моделі, в свою чергу, займаються перетворенням послідовностей фонем у зрозумілі слова та фрази. Фонемне розпізнавання передбачає порівняння ймовірнісних послідовностей фонем зі словником, який містить усі можливі комбінації фонем для побудови слів. Таким чином, система визначає, які слова найімовірніше відповідають розпізнаним фонемам. Словниковий модуль є важливим компонентом лінгвістичної моделі, оскільки він містить перелік слів та їхніх фонемних представлень. Моделювання мови включає використання моделей мови для визначення ймовірності того, що певна послідовність слів відповідає граматичним і семантичним правилам мови. Наприклад, n-грамні моделі враховують ймовірність появи слова в залежності від попередніх слів, тоді як більш складні методи, такі як нейронні мережі, використовуються для обробки природної мови (NLP).

1.3. Синтез мови

Синтез мови, також відомий як технологія перетворення тексту в мову (Text-to-Speech, TTS), цей процес дозволяє перетворювати текстову інформацію у природню мову яку ми сприймає на слух, і це дає змогу системам відповідати користувачам голосом.

Принципи синтезу мови

Синтез мови складається з декількох етапів, кожен з яких виконує певну роль у створенні природного звучання:

- Текстовий аналіз: Перший крок включає обробку тексту для визначення його структури і значення. Це включає розбиття тексту на речення і слова, визначення пунктуації, абревіатур, чисел, символів та інших елементів. Також враховується контекст, щоб правильно інтонувати фрази.

- Лінгвістичний аналіз: На цьому етапі визначаються частини мови, граматичні зв'язки, акценти, інтонація і ритм. Це дозволяє створити більш природне звучання шляхом імітації природних особливостей мови.

- Акустичний аналіз: Здійснюється перетворення лінгвістичної інформації в акустичні параметри, такі як висота звуку, інтенсивність, тривалість фонем та пауз. Ці параметри необхідні для створення природного звучання.

Існує кілька основних методів синтезу мови які використовуються для голосових асистентів, а саме:

Формантний синтез: Цей метод базується на використанні моделей формантів, які є резонансними частотами, що визначають звучання голосних звуків. Формантний синтез дозволяє генерувати мову з нуля без використання записаних голосових зразків, що робить його дуже гнучким. Проте цей метод може створювати менш природне звучання.

Конкатенативний синтез: Цей метод використовує попередньо записані зразки мови, які з'єднуються для створення нових фраз. Зразки можуть бути цілими словами, складами або навіть окремими фонемами. Конкатенативний синтез забезпечує високоякісне і природне звучання, проте вимагає великого обсягу даних і складної обробки.

Статистичне параметричне моделювання: Цей метод використовує статистичні моделі, такі як приховані Марковські моделі (HMM) та глибокі

нейронні мережі, для генерування акустичних параметрів мови. Статистичне параметричне моделювання дозволяє генерувати мову з високою гнучкістю і відтворювати різноманітні голоси та інтонації. Проте якість звучання може бути нижчою порівняно з конкатенативним синтезом.

Нейронний синтез: Цей метод використовує глибокі нейронні мережі, такі як WaveNet та Tacotron, для синтезу мови. Нейронні мережі можуть вивчати складні патерни мови і генерувати дуже природне звучання з високою якістю. Цей метод є дуже перспективним, проте вимагає великих обчислювальних ресурсів і обсягів даних для навчання.

1.4. Обробка природної мови (NLP)

Обробка природної мови (Natural Language Processing, NLP) займається взаємодією між комп'ютерами і людьми за допомогою природної мови. Метою NLP є розуміння, інтерпретація та генерація людської мови таким чином, щоб комп'ютери могли розуміти та реагувати на неї. NLP охоплює широкий спектр технологій і методів, спрямованих на аналіз і генерацію природної мови. Основні принципи включають морфологічний аналіз, який вивчає структуру слів, включаючи розбиття на морфеми (це найменші значущі одиниці мови) та визначення їх граматичних властивостей. Синтаксичний аналіз визначає граматичну структуру речень, включаючи ідентифікацію частин мови та їх зв'язків, що дозволяє зрозуміти, як слова взаємодіють у реченні. Семантичний аналіз аналізує значення слів і фраз у контексті, включаючи визначення значення слів, враховуючи контекст їх використання, і розуміння сенсу висловлювання. Прагматичний аналіз вивчає, як мова використовується в реальних комунікативних ситуаціях, включаючи аналіз намірів мовця та контексту спілкування

Існує кілька основних методів NLP, які використовуються для обробки природної мови. Правила і словники є традиційними методами, які використовують набори правил і словники для аналізу і генерації мови. Ці методи можуть бути ефективними для обробки структурованих текстів, але обмежені в обробці складних і варіативних мовних конструкцій. Статистичні методи

використовують статистичні моделі для аналізу великих обсягів тексту і виявлення закономірностей, включаючи моделі n-грам, які оцінюють ймовірність появи слова в залежності від попередніх слів, та інші статистичні підходи. Машинне навчання використовує алгоритми для навчання моделей на великих наборах даних, включаючи класифікацію текстів, розпізнавання іменованих сутностей, аналіз настроїв та інші завдання. Алгоритми машинного навчання можуть виявляти складні патерни і забезпечувати високу точність обробки мови. Глибоке навчання використовує глибокі нейронні мережі, такі як рекурентні нейронні мережі (RNN) та трансформери (наприклад, BERT і GPT), для обробки природної мови, вивчаючи складні взаємозв'язки в тексті і забезпечуючи високу точність і гнучкість у розумінні та генерації мови.

2. АРХІТЕКТУРА СИСТЕМИ ГОЛОСОВОГО ПОМІЧНИКА

Архітектура системи голосового помічника складається з кількох ключових компонентів, які працюють разом для забезпечення ефективної взаємодії з користувачем. Ці компоненти забезпечують розпізнавання голосу, розуміння природної мови, генерацію відповідей та синтез мови. Розглянемо основні елементи архітектури системи голосового помічника.

2.1. Модульна структура

Акустичний фронт-енд здійснює попередню обробку аудіосигналу, модуль розпізнавання мови (ASR) перетворює звукові сигнали в текст, модуль обробки природної мови (NLP) розуміє та інтерпретує текстові запити, а модуль синтезу мови (TTS) перетворює текстові відповіді у природно звучащу мову. Інтеграційний шар координує взаємодію між компонентами та користувачем, забезпечуючи зручний та ефективний інтерфейс.

2.1.1. Вхідний модуль (розпізнавання мови)

До нього можна включити 2 пункти:

Акустичний фронт-енд виконує попередню обробку вхідного звукового сигналу. Цей етап включає збір аудіосигналу через мікрофон, фільтрацію шумів та нормалізацію гучності. Потім здійснюється функціональний аналіз, який перетворює аудіосигнал у спектральні ознаки, наприклад, мел-частотні кепстральні коефіцієнти (MFCC).

Та модуль ASR, який перетворює звукові сигнали у текст. Він використовує акустичне моделювання для визначення послідовностей фонем, лексичне моделювання для перетворення фонем у слова та мовне моделювання для визначення ймовірності послідовностей слів на основі контексту.

2.1.2. Модуль обробки даних (NLP)

Модуль NLP відповідає за розуміння текстових запитів користувача та генерацію відповідей. Він виконує семантичний аналіз для визначення значення слів і фраз у контексті, синтаксичний аналіз для визначення граматичної структури речення, розуміння намірів користувача, розпізнавання іменованих сутностей та генерацію відповідей на основі запиту.

2.1.3. Вихідний модуль (синтез мови)

Модуль TTS перетворює текстові відповіді у природно звучащу мову. Він включає текстовий аналіз для визначення граматичних акцентів, лінгвістичний аналіз для визначення фонемної структури тексту та синтез мови, який генерує звуковий сигнал на основі акустичних параметрів.

2.2. Інтеграція з хмарними сервісами

Інтеграція голосових помічників з хмарними сервісами є важливою для забезпечення масштабованості, надійності та продуктивності. Хмарні платформи, такі як AWS, GCP та Azure, надають можливості для зберігання та обробки великих обсягів даних, що забезпечує надійний доступ до інформації та аналіз даних у реальному часі.

Хмарні API, як Google Cloud Speech-to-Text, Amazon Transcribe та Microsoft Azure Speech API, дозволяють інтегрувати високоякісні сервіси для розпізнавання мови без створення власних алгоритмів. Вони також пропонують інструменти для розробки, тренування та деплою моделей машинного навчання та штучного інтелекту. Використовуючи сервіси, як Google Cloud AI Platform, AWS SageMaker та Azure Machine Learning, розробники можуть створювати і вдосконалювати моделі для розпізнавання мови та NLP.

Хмарні сервіси забезпечують безпеку даних, включаючи шифрування, контроль доступу та моніторинг безпеки, що захищає дані користувачів і забезпечує відповідність нормативним вимогам. Масштабованість хмарних

платформ дозволяє адаптуватися до змінних навантажень, обробляючи велику кількість запитів без втрати продуктивності.

2.3. Безпека і конфіденційність даних

Безпека даних користувачів забезпечується через шифрування даних у стані спокою і під час передачі, що запобігає несанкціонованому доступу. Контроль доступу включає автентифікацію користувачів, управління ролями та багатофакторну автентифікацію. Моніторинг та аудит дозволяють відслідковувати доступ до даних і виявляти підозрілі активності, що допомагає швидко реагувати на можливі загрози. Хмарні провайдери дотримуються міжнародних стандартів безпеки і конфіденційності, таких як GDPR, HIPAA та ISO/IEC 27001, забезпечуючи відповідність нормативним вимогам. Фізична безпека інфраструктури гарантує захист від фізичних загроз, таких як пожежі і несанкціонований доступ до обладнання.

3. ТЕХНІЧНІ ЗАСОБИ І ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ

Використання мікрофонів забезпечує чітке захоплення аудіо сигналу, необхідне для точного розпізнавання мови, а аудіосистеми відповідають за відтворення звуку. Потужні процесори та спеціалізовані чіпи обробляють аудіо і виконують алгоритми машинного навчання, оптимізовані для обробки і синтезу мови. Надійне мережеве обладнання забезпечує стабільне і швидке підключення до інтернету для інтеграції з хмарними сервісами.

3.1. Програмне забезпечення

Для голосового помічника Amazon Alexa використовується спеціальна платформа Alexa Voice Service (AVS), яка містить у собі алгоритми розпізнавання мови і обробки природної мови, а також доступ до хмарних сервісів Amazon для виконання запитів користувачів.

Google Assistant використовує аналогічні технології, включаючи власну платформу для обробки голосових команд та доступ до різних сервісів Google.

Для Apple Siri використовується інша платформа, що базується на технологіях компанії Apple для обробки голосових команд і надання користувачеві відповідей.

3.1.1. **Бібліотеки для розпізнавання мови (CMU Sphinx, Google Speech API)**

Найбільш відомими є CMU Sphinx і Google Speech API. CMU Sphinx, розроблена в університеті Карнегі-Меллон, є однією з найстаріших та найбільш широко використовуваних бібліотек для автоматичного розпізнавання мови (ASR). Вона надає можливості для створення офлайн-систем розпізнавання мови, що особливо корисно для застосунків, які потребують роботи без інтернет-з’єднання. CMU Sphinx підтримує різні мови і має відкритий вихідний код, що дозволяє розробникам адаптувати її для своїх потреб.

Google Speech API, з іншого боку, є хмарним сервісом, який надає Google. Він використовує потужні алгоритми машинного навчання і великий обсяг даних для забезпечення високої точності розпізнавання мови. Google Speech API підтримує велику кількість мов і діалектів, що робить його зручним для глобального використання. Цей сервіс дозволяє розробникам інтегрувати розпізнавання мови у свої додатки за допомогою простих RESTful API запитів, забезпечуючи доступ до передових технологій обробки мови.

3.1.2. Інструменти для обробки мови (NLTK, spaCy)

Такі інструменти, як NLTK і spaCy, широко використовуються для обробки природної мови (NLP) Інструментарій природної мови (NLTK) - одна з найпопулярніших бібліотек для обробки тексту в лінгвістичних дослідженнях і програмуванні. Вона надає широкий спектр інструментів для роботи з текстом, включаючи токенізацію, частковий аналіз мови, розпізнавання власних виразів та синтаксичний аналіз. NLTK також містить велику кількість освітніх ресурсів і корисних даних для дослідників і розробників, що робить його ідеальним для академічних і дослідницьких проектів.

SpaCy, з іншого боку, є сучасною, швидкою бібліотекою обробки природної мови, призначеною для промислового використання. Вона оптимізована для продуктивності та масштабованості і надає високоякісні моделі для різних мов. spaCy підтримує такі завдання, як токенізація, частковий аналіз мови, розпізнавання сутностей і лематизація, а також має вбудовані функції для обробки великих обсягів тексту. Крім того, spaCy можна легко інтегрувати з іншими бібліотеками машинного навчання, такими як TensorFlow і PyTorch, для створення потужних і гнучких систем обробки природної мови.

3.1.3. Платформи для синтезу мови (Google TTS, Amazon Polly)

Платформи перетворення тексту в мовлення, такі як Google Text-to-Speech (TTS) і Amazon Polly, є потужними інструментами для перетворення тексту в природне мовлення. Google Text-to-Speech (TTS) використовує передові алгоритми машинного навчання для забезпечення якісного синтезу мовлення для створення високоякісного тексту в мовлення. Платформа підтримує велику кількість мов і голосів, надаючи розробникам широкий спектр можливостей. Google TTS інтегрується з іншими сервісами Google, такими як Google Assistant і Android, що робить її корисною для створення багатомовних і багатофункціональних додатків. Він також надає API для легкого доступу до функцій перетворення тексту в мовлення через хмару.

Amazon Polly - це хмарний сервіс перетворення тексту в мовлення, що надається Amazon Web Services (AWS). Amazon Polly використовує передові нейронні мережі для генерування природної мови з різними параметрами, такими як швидкість мовлення та висота тону. Polly підтримує кілька мов і акцентів, що дозволяє розробникам створювати багатомовні додатки. Крім того, Amazon Polly пропонує такі функції, як синтез мови розмітки Speech Synthesis Markup Language (SSML) для більш точного контролю над вимовою та інтонацією. Як частина екосистеми AWS, Polly доступний з Amazon S3 і Lambda і легко інтегрується з іншими сервісами Amazon, такими як Amazon S3 і Lambda, що забезпечує масштабованість і надійність.

4. РОЗРОБКА ТА ТЕСТУВАННЯ

4.1 Перша версія помічника

Мовою програмування я обрав Python, через наступні переваги:

1. Легкість у використанні

Python відомий своєю простотою та легкістю у використанні. Його синтаксис дозволяє швидко писати та читати код, що особливо важливо при розробці складних додатків, таких як голосовий помічник.

2. Велика кількість бібліотек

Python має велику кількість бібліотек і модулів, що спрощують розробку голосових помічників:

- **SpeechRecognition**: Бібліотека для розпізнавання мови.
- **pyttsx3**: Бібліотека для синтезу мовлення.
- **OpenAI API**: Легко інтегрується з API OpenAI для використання GPT-3.

3. Підтримка спільноти

Python має велику та активну спільноту розробників, що забезпечує доступ до великої кількості ресурсів, включаючи документацію, форуми та навчальні матеріали. Це робить процес розробки більш ефективним та менш стресовим.

Створення прототипу

Перша версія голосового асистента постійно прослуховуватиме мікрофон в фоновому режимі, і розумітиме що звертаються до нього. З тестового функціоналу будуть можливості: відповісти яка зараз година, запуск додатку та гри, запуск музики і відповідь декількома примітивними жартами. Модель розуміє і спілкується українською, у ній також присутня проста, проте ефективна система нечіткого розпізнавання команд.

Для спілкування в проекті застосовується модуль мови Python pyttsx3, в додаток до якого потрібно встановити pypiwine32 та pywin32.

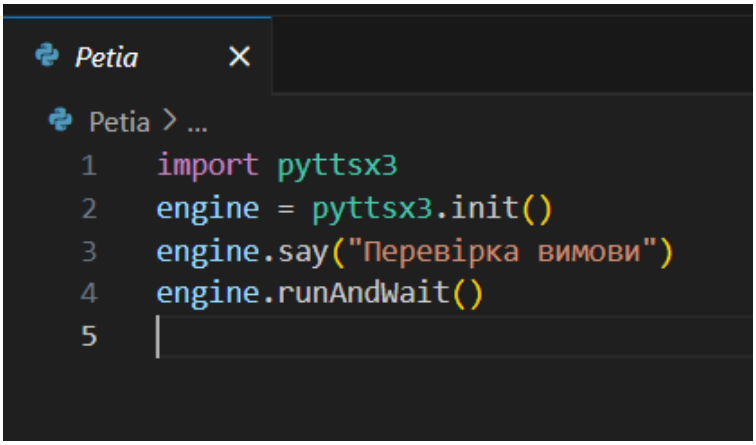

Рисунок. 4.1 Код модуля спілкування

Код для перевірки коректності роботи голосового виводу. Після його запуску написана нами фраза озвучиться комп'ютером.

Наступне завдання це розпізнавання нашої мови. Спершу розглядалась бібліотека pocketsphinx, вона може працювати оффлайн, проте вона не достатньо добре розпізнає українську і завантажує близько 500 Мб власної бази в оперативну пам'ять під час запуску, що сповільнює розпізнавання сказаних слів. Тому для розпізнавання було обрано Google API, Google Speech Recognition. Додатково було встановлено бібліотеку PyAudio, для живого запису звуків з мікрофону.

Ще один невеликий код для перевірки коректності роботи модуля розпізнавання, і вже можемо перетворювати мову в текст, і текст у мову.

```
r = sr.Recognizer()
m = sr.Microphone(device_index = 1)

with m as source:
    r.adjust_for_ambient_noise(source)

speak_engine = pyttsx3.init()

#
voices = speak_engine.getProperty('voices')
speak_engine.setProperty('voice', voices[4].id)
```

Рисунок. 4.2 Код для перетворення і озвучення мови

Цей код запускає голосового помічника, і слухає фон в межах однієї секунди, щоб відрізняти шум і голос людини.

```
import os
import time
import speech_recognition as sr
from fuzzywuzzy import fuzz
import pyttsx3
import datetime

opts = {
    "alias": (),
    "tbr": (),
    "cmds": {}
}
```

Рисунок. 4.3 Початкова конфігурація голосового асистента

Alias – відповідає за всі можливі варіанти звернень до асистента.

Tbr – (to be removed) відповідатиме за всі слова які можна видалити з голосової команди для скорішої обробки

Cmds – зберігатиме всі можливі назви виконуваних команд нашим асистента.

Дописавши код ми отримуємо першу робочу версію помічника який може робити лише те, що чітко прописано в його коді. З недоліків можна виділити необхідність постійного підключення до інтернету. Малу кількість виконуваних команд, стандартний надто роботизований голос, довге очікування відповіді.

4.2 Друга версія

Для подальших покращень, встановимо додатково бібліотеку silero, яка замінить pyttsx3 для синтезу мови. Переваги silero у тому, що вона більш новіша й навчена з допомогою нейронних мереж. Для кращого відтворення мови також додамо такі бібліотеки як sounddevice та numpy. А для розпізнавання скористаємось VOSK. Ця бібліотека дає нам рушій для розпізнавання багатьох мов, зокрема української. Відредагувавши код, з новими можливостями отримуємо значно краще розпізнавання та озвучення.

Наступний етап, це зменшення затримки до відповіді:

Проблема у тому що наш помічник відкликається лише на декілька звернень. А в українській мові складається з близька 256 тисяч слів. Найшвидша нейромережа для української мови, розпізнає сказане слово за ± 500 мс. Враховуючи час на вимову, час на оброблення та простій, від початку вимови до активації та відповіді голосового помічника проходить 4-6 секунд.

Щоб зменшити цей час скористаємось алгоритмом WakeWorkDetection. Він дає нам змогу прослуховувати середовище навколо, в пошуках активаційної фрази для проміжку 30 мс. Активаційні фрази використовуються у всіх помічниках. Для продуктів Apple це “Привіт Siri”, у Google асистента це “Ok Google”. Задавши свою активаційну фразу і додавши її до коду ми отримуємо майже миттєву відповідь після звернення.

В доповнення можна підключити API з GPT 3.5. Для цього необхідно встановити бібліотеку “openai” та отримати ініціалізований ключ API з офіційного сайту, і додавши наступний код, ми отримуємо дію яка буде зверненням до GPT 3.5, а отримана відповідь відразу озвучуватиметься.

```
def get_gpt_response(prompt):
response = openai.Completion.create(
engine="davinci-codex",
prompt=prompt, max_tokens=150
 )
return response.choices[0].text.strip()
```

Після цих покращень наш асистент в межах секунди відгукується на звернення, має приємний голос та має інтеграцію з GPT 3.5 для відповіді на будь які питання.

4.3 Нова третя версія

Для наступної версії помічника було вирішено обрати мову програмування Rust.
З його переваг можна виділити

1. Висока продуктивність

Rust є мовою системного програмування, яка забезпечує продуктивність на рівні з C і C++. Це дозволяє створювати високопродуктивні додатки, що особливо важливо для обробки голосу в реальному часі.

2. Безпека пам'яті

Він пропонує безпеку пам'яті без використання збірника сміття (garbage collector). Це знижує ризик помилок, пов'язаних з використанням пам'яті, таких як null вказівники та переповнення буферів, що робить ваш код більш надійним.

3. Паралелізм

Також він має вбудовану підтримку паралельного програмування, що дозволяє ефективно використовувати багатоядерні процесори. Це важливо для обробки голосу та інших завдань, що потребують високої продуктивності.

4. Крос-платформність

Rust дозволяє легко створювати крос-платформні додатки, які можна запускати на різних операційних системах, таких як Windows, macOS та Linux.

Для розробки помічника було встановлено безліч необхідних бібліотек. У нового помічника було розроблено парсер команд, та покращено алгоритм нечіткого розпізнавання.
Найпростіший код для робочого помічника з GPT на мові програмування Rust.

```rust
use rust_google_speech::recognize;
use rust_openai::GPT;
use std::io::{self, Write};

fn main() {
    // Ініціалізація GPT API
    let gpt = GPT::new("ваш_API_ключ");

    // Ініціалізація голосового синтезатора (приклад використання системного виклику)
    let synth = |text: &str| {
        let _ = std::process::Command::new("say")
            .arg(text)
            .status();
    };

    loop {
        println!("Слухаю...");

        // Виклик функції розпізнавання мови
        let speech_result = recognize().expect("Не вдалося розпізнати мову");

        let query = speech_result.alternatives[0].transcript.clone();
        println!("Ви сказали: {}", query);

        // Отримання відповіді від GPT
        let response = gpt.complete(&query).expect("Не вдалося отримати відповідь від GPT");

        println!("Відповідь: {}", response);
        synth(&response);

        if query.to_lowercase() == "вийти" || query.to_lowercase() == "зупинити" {
            println!("Завершення роботи голосового помічника.");
            break;
        }
    }
```

Рисунок. 4.4 Програма голосового асистента на мові Rust

4.4 Наступні поліпшення

Інтеграція голосового помічника з розумними пристроями дозволить користувачам керувати своїм будинком за допомогою голосових команд. Це можуть бути пристрої такі як освітлення, термостати, системи безпеки, розетки та інше. Для цього можна використати різні протоколи та сервіси, такі як:

- **MQTT**: Легкий протокол для обміну повідомленнями між пристроями.
- **Home Assistant**: Популярна платформа для управління розумним будинком.
- **IFTTT**: Сервіс, що дозволяє автоматизувати взаємодію між різними пристроями та сервісами.

Приклад коду для цієї інтеграції:

```rust
use paho_mqtt as mqtt;

fn control_device(command: &str) {
    let client = mqtt::Client::new("tcp://broker.hivemq.com:1883").unwrap();
    client.connect(None).unwrap();

    let message = mqtt::Message::new("home/device/command", command, mqtt::QOS_1);
    client.publish(message).unwrap();

    client.disconnect(None).unwrap();
}
```

Рисунок. 4.5 Код інтеграції для відправки команд

Також для економії ресурсів ваш голосовий помічник може перебувати у стані сну допоки не почує активаційну фразу, наприклад "Привіт, помічнику". Це можна реалізувати за допомогою бібліотеки для постійного прослуховування аудіопотоку та розпізнавання ключової фрази.

ВИСНОВОК

Розробка персонального голосового асистента, який здатний розуміти та виконувати запити користувачів, є важливим кроком у розвитку сучасних технологій. В ході виконання дипломної роботи була створена система, яка використовує методи обробки природної мови (NLP) для розпізнавання та синтезу мовлення. Асистент інтегрується з різноманітними хмарними сервісами, що дозволяє йому виконувати завдання з управління розумними пристроями, надання інформаційних послуг, організації розкладу та багато іншого.

Основною метою роботи було створення голосового помічника, який забезпечує високу точність розпізнавання мовлення та персоналізацію під потреби користувача. Для досягнення цієї мети було використано системний аналіз, моделювання, методи розпізнавання мовлення, методи синтезу мовлення, аналіз даних, програмування, тестування та оптимізацію систем. Створений голосовий асистент показав високу ефективність у виконанні поставлених завдань. Він використовує передові алгоритми для розпізнавання мовлення, що забезпечує високу точність та швидкість відповіді. Інтеграція з хмарними сервісами забезпечує надійний доступ до інформації та можливість обробки великих обсягів даних у реальному часі.

Практичне значення отриманих результатів полягає у можливості використання розробленого голосового асистента в повсякденному житті, що спрощує взаємодію з технологіями та робить їх більш доступними і зручними. Високий рівень персоналізації дозволяє налаштовувати асистента під конкретні потреби користувача, що робить його незамінним інструментом. Успішна реалізація проекту свідчить про великий потенціал подібних систем та їх важливість для сучасного суспільства. Розуміння можливостей та обмежень голосових асистентів є ключовим для їх успішного впровадження та використання у майбутньому.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Що таке Automatic Speech Recognition (ASR)
   URL: https://tseivo.com/b/memecode/t/0nek6ary6r (дата звернення 15.05.2024)

2. The much-needed reinvention of the voice assistant is almost here
   URL: https://www.theverge.com/2024/6/14/24177991/apple-intelligence-siri-voice-assistant-amazon-alexa-generative-ai (дата звернення 11.05.2024)

3. Alexa, Google Assistant и Apple HomeKit
   URL: https://superhome.pro/alexa-google-assistant-i-apple-homekit-vyiberete-svoyu-ekosistemu-umnogo-doma/ (дата звернення 20.05.2024)

4. What is NLP (natural language processing)?
   URL: https://www.ibm.com/topics/natural-language-processing (дата звернення 12.05.2024)

5. API тегування частин мови (POS) та розбору залежностей на основі spaCy
   URL: https://nlpcloud.com/uk/nlp-part-of-speech-pos-tagging-api.html (дата звернення 14.05.2024)

6. Обробка природної мови
   URL: https://nlpcloud.com/uk/introduction-what-is-nlp-natural-language-processing.html (дата звернення 14.05.2024)

7. Speech into text using Google AI
   URL: https://cloud.google.com/speech-to-text (дата звернення 18.05.2024)

8. Системи розпізнавання речей з відкритим вихідним кодом
   URL: https://lvee.org/uk/abstracts/273 (дата звернення 14.05.2024)