Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Інформаційна система первинної підготовки запитів до chatGPT

(тема)

Виконав:
студент 2 курсу, групи СШІм-21-2
Масловський М.Ю.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

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

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Масловському Микиті Юрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Інформаційна система первинної підготовки запитів до chatGPT

затверджена наказом університету від 31 березня 20 23 р. № 306Ст

2. Термін подання студентом роботи до екзаменаційної комісії 19 травня 20 23 р.

3. Вихідні дані до роботи Науково-технічні публікації, дані інтернет-джерел та відомих проектів щодо аналізу GPT, Flutter документація, Android Studio документація.

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз визначеної предметної області
2) Постановка задачі
3) Вибір моделі для використання
4) Метод побудови додатку
5) Розробка додатку

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Ознайомлення із завданням. Уточнення ТЗ	01.04.2023	Виконано
2	Підбір матеріалів за темою роботи	08.04.2023	Виконано
3	Виконання розділу 1	15.04.2023	Виконано
4	Виконання розділу 2	22.04.2023	Виконано
5	Виконання розділу 3	24.04.2023	Виконано
6	Виконання розділу 4	28.04.2023	Виконано
7	Виконання розділу 5	30.04.2023	Виконано
8	Оформлення пояснювальної записки	07.05.2023	Виконано
9	Оформлення презентації	12.05.2023	Виконано
10	Захист перед ЕК	19.05.2023	

Дата видачі завдання 3 квітня 2023 р.

Студент _____
(підпис)

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

РЕФЕРАТ

Пояснювальна записка: 59 с., 23 рис., 1 дод., 20 джерел.

АНАЛІЗ, ІНТЕЛЕКТУАЛЬНИЙ ПОМІЧНИК, ШТУЧНИЙ ІНТЕЛЕКТ, GPT, FLUTTER.

Об’єкт дослідження – інформаційна система первинної підготовки запитів до GPT.

Предмет дослідження – процес первинної підготовки запитів до GPT, який включає в себе збір та обробку вхідної інформації, вибір алгоритмів та налаштування параметрів системи.

Мета роботи – розробка та впровадження ефективної інформаційної системи первинної підготовки запитів до GPT для забезпечення якісної та швидкої обробки запитів користувачів.

Методи дослідження – аналіз наукової літератури з питань інформаційної системи первинної підготовки запитів до GPT, експериментальне визначення оптимальних параметрів системи на основі тестування та оцінки її продуктивності, розробка програмного забезпечення для реалізації системи та тестування її функцій.

ABSTRACT

Explanatory note: 59 p., 23 fig., 1 ann., 20 sources.

ANALYSIS, ARTIFICIAL INTELLIGENCE, INTELLECTUAL ASSISTANT, GPT, FLUTTER

Research object – information system for primary query preparation for GPT.

Research subject – the process of primary query preparation for GPT, which includes collecting and processing input information, selecting algorithms, and configuring system parameters.

Research aim – to develop and implement an effective information system for primary query preparation for GPT to ensure high-quality and fast query processing for users.

Research methods – analysis of scientific literature on the issues of the information system for primary query preparation for GPT, experimental determination of optimal system parameters based on testing and evaluation of its performance, development of software to implement the system, and testing its functions.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AI – Artificial Intelligence – штучний інтелект;

API – Application Programming Interface – інтерфейс прикладного програмування;

GPT – Generative Pre-trained Transformer – генеративний попередньо навчений трансформатор;

ML – Machine Learning – машинне навчання;

SDK – Software Development Kit – набір для розробки програмного забезпечення.

ВСТУП

Зростаюча популярність та розвиток обробки природної мови призвели до створення великих мовних моделей, таких як GPT. Однак, ефективність та точність таких моделей в значній мірі залежать від якості та відповідності вхідних запитів. Щоб забезпечити надійні та релевантні відповіді від GPT та інших подібних моделей, необхідна ефективна інформаційна система первинної підготовки запитів. Розробка такої системи потребує глибокого розуміння процесу первинної підготовки запитів та використання відповідних методів інформаційної технології, які забезпечують вибір оптимальних алгоритмів та параметрів системи. Мета дослідження полягає в розробці ефективної інформаційної системи первинної підготовки запитів до GPT, що дозволить забезпечити найвищу якість та швидкість обробки запитів користувачів. Для досягнення цієї мети необхідно провести дослідження методів та алгоритмів первинної підготовки запитів, розробити та випробувати програмне забезпечення системи, провести експерименти з оцінки продуктивності та ефективності системи та визначити її оптимальні параметри. Результати цього дослідження будуть корисними для розробки інформаційних систем на основі інших мовних моделей та забезпечують розвиток природної обробки мови в цілому. Крім того, розробка ефективної інформаційної системи первинної підготовки запитів до GPT є важливою для вирішення різних завдань у багатьох галузях, включаючи медицину, науку, бізнес, освіту та інші. Застосування інформаційної системи первинної підготовки запитів до GPT допоможе зменшити навантаження на користувачів, які можуть не мати достатнього досвіду або знань для правильної підготовки запитів, та забезпечує більш точні та нерелевантні відповіді. Тому, розробка ефективної інформаційної системи первинної підготовки запитів до GPT є актуальною і важливою проблемою, яку необхідно вирішити для

подальшого розвитку природної обробки мови та її застосування в різних галузях.

Штучний інтелект (Artificial Intelligence, AI) – це галузь комп'ютерної науки, яка займається розробкою програм та систем, здатних виконувати завдання, які зазвичай пов'язані з інтелектом людини, наприклад, розпізнавання мови, розуміння текстів, прийняття рішень, навчання тощо. Системи штучного інтелекту базуються на математичних моделях та алгоритмах, які навчаються на великих об'ємах даних, а потім здатні застосовувати здобуті знання для вирішення завдань та задач.

Одним з найбільш відомих і поширених застосувань штучного інтелекту є системи глибокого навчання, такі як GPT, які здатні вирішувати завдання, пов'язані з природною мовою, включаючи генерацію тексту, розпізнавання мовлення та аналіз тексту. Штучний інтелект також знаходить застосування в інших галузях, таких як робототехніка, автоматизація виробництва, медицина, фінанси, наука та інші.

Штучний інтелект є однією з найбільш швидко розвиваючихся галузей комп'ютерної науки, що вже зараз має значний вплив на наші життя та розвиток суспільства в цілому. Розробка систем штучного інтелекту потребує високої кваліфікації та розуміння різних аспектів комп'ютерної науки та математики, але при цьому може мати значний вплив на розвиток бізнесу та науки, технічний прогрес та поліпшення якості життя людей.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність GPT для вирішення проблем

Останнім часом велика кількість людей використовує ШІ для знаходження відповіді на своє питання. Хоч більшість і використовує його ради забави, але деяка частина хоче більш ефективно використовувати цю систему для отримання найбільш точної відповіді. Для отримання найбільш точної відповіді потрібно правильно поставити питання і для цього і був розроблений цей додаток на основі GPT цей додаток є однією з найбільш відомих та використовуваних моделей глибокого навчання, що використовуються для розв'язання проблем, пов'язаних з обробкою природної мови. Основна ідея застосування GPT полягає в тому, щоб попередньо підготувати модель на великій кількості даних, після чого вона здатна зрозуміти залежності в тексті та згенерувати новий текст з використанням цих знань.

Актуальність GPT для вирішення проблем полягає у тому, що ця модель може знайти застосування в різних галузях, де важливо аналізувати та генерувати текстові дані. Наприклад, в медичній галузі можна використовувати GPT для аналізу медичних записів, визначення діагнозів та розробки планів лікування. Також GPT може знайти застосування в сфері фінансів, де важливо аналізувати тексти звітів та прогнозувати тренди на ринку.

Окрім того, GPT може бути використаний для покращення ефективності бізнес-процесів. Наприклад, він може допомогти в автоматичному створенні контенту для маркетингу та реклами, що зменшить трудомісткість та збільшить продуктивність працівників. Також GPT може бути використаний для автоматичної генерації відповідей на запитання користувачів у сфері клієнтського сервісу.

У цілому, можна стверджувати, що GPT є актуальним та перспективним напрямком розвитку штучного інтелекту та машинного навчання, що здатний знайти застосування в багатьох галузях. Однак, варто пам'ятати про можливі ризики та етичні питання, пов'язані з його використанням, та розробляти відповідні стандарти для забезпечення безпеки та захисту прав людей.

1.2 Аналіз наявних додатків GPT генерування тексту

Додатки для генерування тексту є досить популярними і корисними інструментами для тих, хто працює з великим обсягом текстового контенту або хоче економити час на написання власних текстів. Такі додатки використовують різні алгоритми та штучні інтелектуальні системи, включаючи GPT, щоб автоматично генерувати текст відповідно до введеного користувачем контексту.

Один з прикладів додатків для генерування тексту на Android - GPT Playground. Цей додаток використовує мережу GPT від OpenAI, щоб генерувати текст з різних тем та стилів на основі короткого введення користувача. Він дозволяє вибирати різні параметри генерації, такі як кількість речень, тематику та стиль мови, приклад додатку (рисунок 1.1).

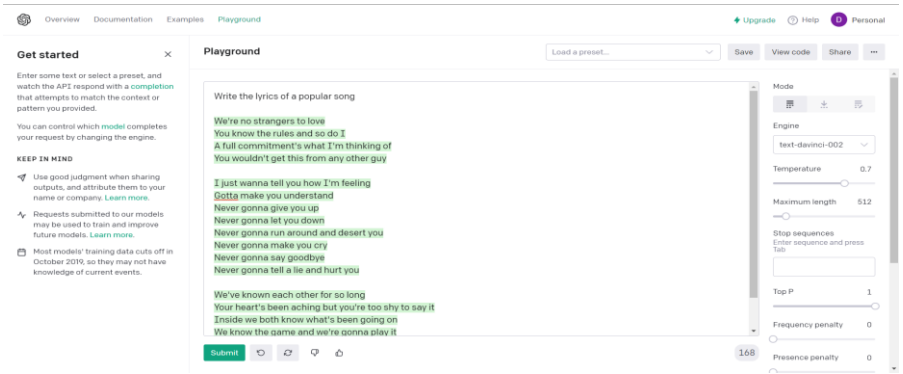

Рисунок 1.1 – Додаток для генерування тексту GPT-Playground

Основні функції GPT Playground включають:

– введення тексту: користувач може ввести короткий текст або ключові слова, на основі яких буде генеруватись далі текст;

– вибір параметрів генерації: користувач може налаштувати різні параметри генерації, такі як кількість речень, довжину тексту, тематику та стиль мови;

– попередній перегляд згенерованого тексту: після введення вхідного тексту та налаштування параметрів генерації, користувач може переглянути попередній варіант згенерованого тексту;

– редагування згенерованого тексту: після того, як користувач отримав згенерований текст, він може редагувати його, виправляти помилки або додавати нові ідеї;

– збереження та експорт згенерованого тексту: після створення тексту користувач може зберегти його у додатку або експортувати у форматі текстового файлу.

GPT Playground є корисним інструментом для тих, хто працює з великим обсягом текстового контенту або хоче економити час на написання власних текстів. Однак, варто зазначити, що застосування штучного інтелекту та автоматичного генерування тексту може викликати етичні та правові питання, зокрема, пов'язані з авторством та правами на інтелектуальну власність.

Інший приклад – Text Blaze. Цей додаток дозволяє створювати шаблони тексту та зберігати їх для майбутнього використання. Крім того, він має функцію автоматичного доповнення тексту на основі введеного користувачем контексту, що може значно заощадити час та зусилля при написанні повторюваних текстів (рисунок 1.2).

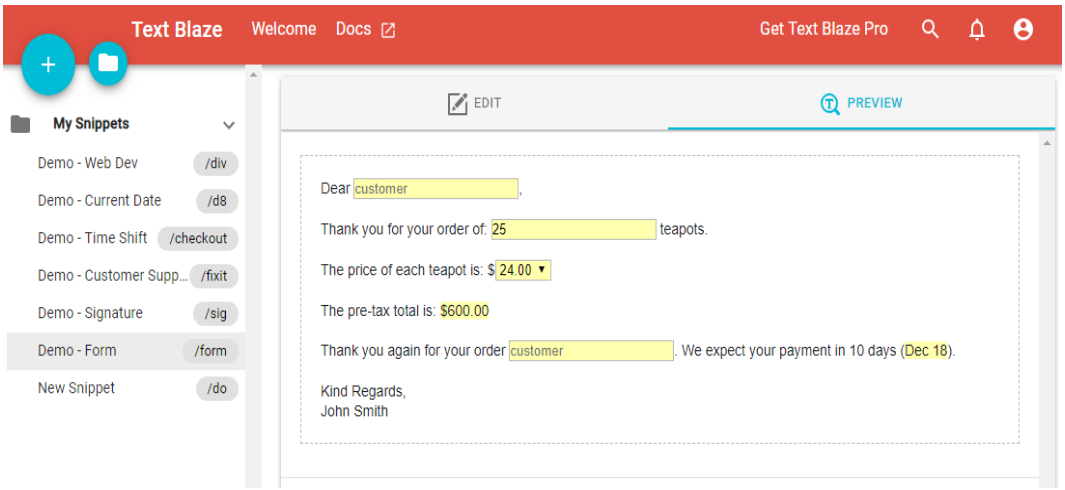

Рисунок 1.2 – Додаток для генерування тексту Text Blaze

Text Blaze – це інструмент для автоматизації роботи з текстом, який дозволяє користувачам створювати власні текстові фрагменти, які можна легко вставляти в будь-який текстовий контент, такий як документи, електронні листи, форми та інше.

Основна функція Text Blaze – допомога користувачам економити час та підвищувати продуктивність, автоматизуючи повторювані завдання введення тексту. За допомогою Text Blaze, користувачі можуть створювати власні фрагменти, які містять часто вживані фрази, шаблони тексту, підписи в електронній пошті, відповіді на форми та інше. Ці фрагменти можуть бути призначені для гарячих клавіш або скорочень, що дозволяє легко вставляти їх у будь-який документ або текстове поле.

Text Blaze також пропонує низку просунутих функцій, включаючи можливість створення вкладених фрагментів, використання змінних та умовної логіки та інтеграцію з зовнішніми джерелами даних. Користувачі також можуть утворювати фрагменти, які містять динамічний контент, такий як поточна дата і час, IP-адреса користувача, або вміст буфера обміну.

Загалом, Text Blaze – потужний інструмент для всіх, хто часто працює з текстовим контентом і хоче ефективно витрачати свій час та підвищувати

продуктивність. Його інтуїтивний інтерфейс та просунуті функції роблять його популярним вибором серед професіоналів.

Також, можна зазначити Copy.ai – це інструмент штучного інтелекту, який використовує алгоритми генерації тексту на основі моделей глибокого навчання, зокрема chat GPT. Цей додаток призначений для автоматизації процесу створення контенту для маркетингу, включаючи соціальні медіа, рекламу та веб-контент.

Основні функції Copy.ai включають:

– генерація заголовків: Користувач може ввести тему або ключові слова, і Copy.ai згенерує заголовки, які можна використовувати для блогів, статей та інших матеріалів;

– генерація описів: Користувач може ввести деталі про свій продукт або послугу, і Copy.ai створить опис, який можна використовувати на веб-сайтах, в рекламі та інших матеріалах, приклад (рисунок 1.3);

– генерація соціальних повідомлень: Copy.ai дозволяє користувачам згенерувати соціальні повідомлення для платформ, таких як Facebook, Twitter та Instagram, приклад (рисунок 1.4);

– переклад тексту: Copy.ai підтримує автоматичний переклад тексту на декілька мов;

– редагування та оптимізація тексту: Copy.ai дозволяє редагувати згенерований текст та оптимізувати його для пошукових систем, щоб забезпечити більшу видимість на веб-сайті.

Copy.ai забезпечує швидку та ефективну генерацію контенту, що дозволяє компаніям зосередитися на більш важливих завданнях. Крім того, Copy.ai пропонує кілька планів ціноутворення з різними можливостями та обсягами використання, що робить його доступним для широкої аудиторії користувачів з різними потребами та бюджетами.

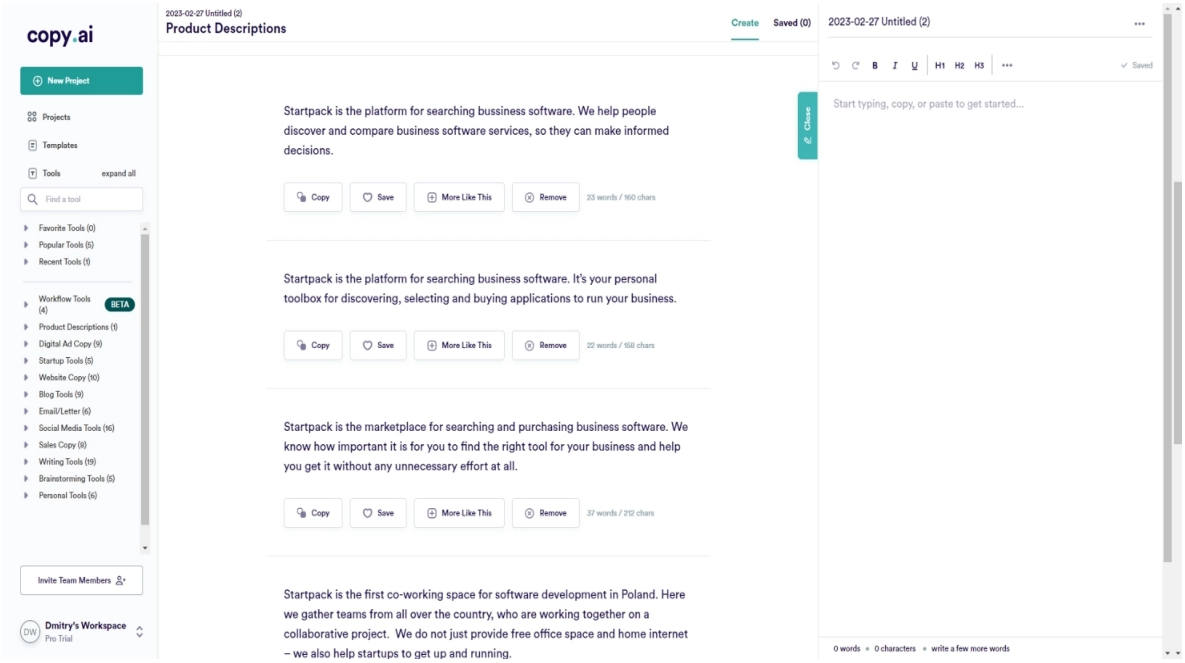

Рисунок 1.3 – Генерація описів Copy.ai

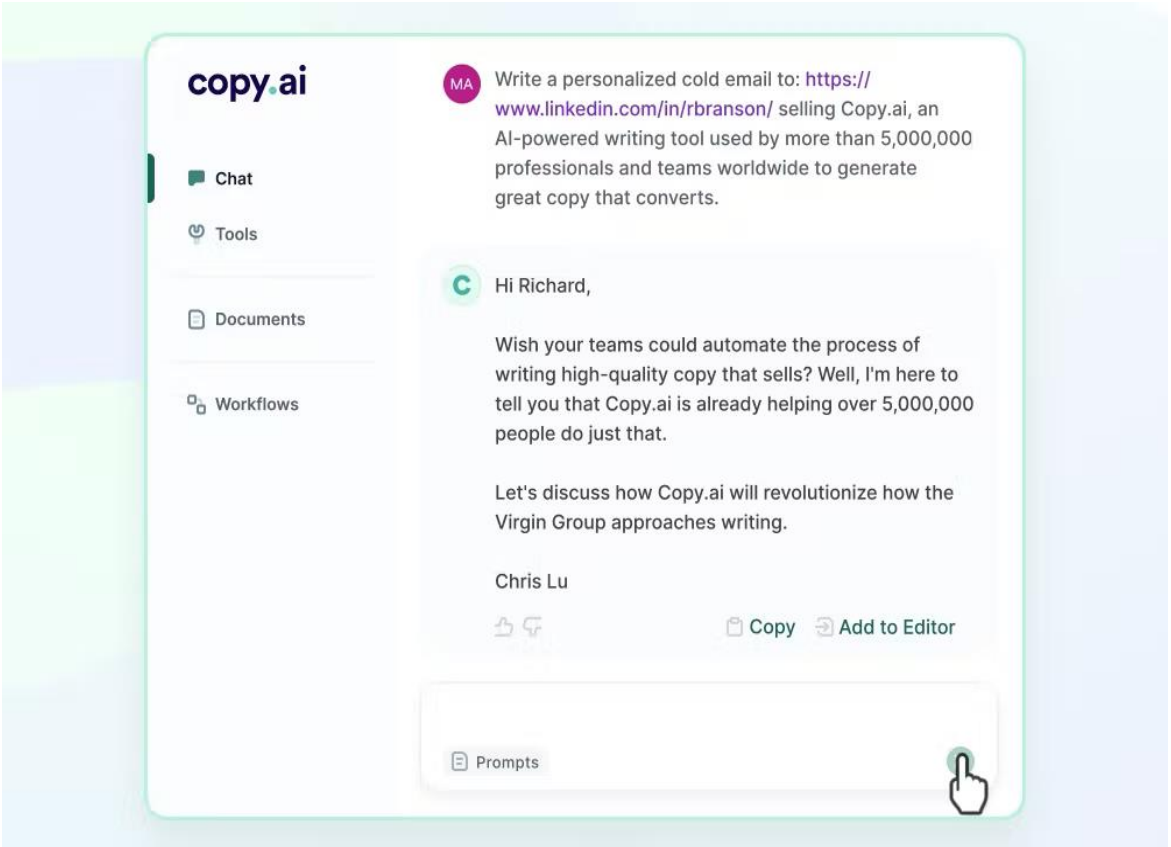

Рисунок 1.4 – Генерація соціальних повідомлень Copy.ai

1.3 Аналіз наборів для розробки

Існує багато наборів для розробки додатків, які можуть бути використані для реалізації різних функцій та завдань. Нижче я наведу кілька з найбільш популярних наборів та коротко опишу їх:

– Android SDK (Software Development Kit) – це набір інструментів та бібліотек для розробки додатків для Android-пристроїв. Він містить інструменти для створення інтерфейсу користувача, роботи зі звуком та відео, баз даних, мережевих запитів та багато іншого. Android SDK надає можливість розробляти додатки для різних пристроїв з різними розмірами екрану та версіями операційної системи;

– iOS SDK – це набір інструментів для розробки додатків для пристроїв Apple, зокрема iPhone, iPad та iPod Touch. Він містить інструменти для створення інтерфейсу користувача, роботи зі звуком та відео, баз даних, мережевих запитів та інших функцій. iOS SDK також надає можливість розробляти додатки для різних розмірів екрану та версій операційної системи;

– React Native – це відкрите програмне забезпечення для розробки мобільних додатків для iOS та Android з використанням JavaScript та React. Цей набір інструментів дозволяє розробляти мобільні додатки з використанням нативних елементів інтерфейсу користувача та мережевих запитів. React Native також надає можливість швидко розробляти та тестувати додатки, що дозволяє економити час та зусилля;

– Xamarin – це інструмент для розробки мобільних додатків для різних платформ, таких як iOS, Android та Windows. Він дозволяє розробляти додатки з використанням мов програмування C# та .NET Framework. Xamarin дозволяє створювати кросплатформні додатки з високою продуктивністю та максимальним повторним використанням коду. Також він має вбудований інструментарій для розробки інтерфейсу

користувача та можливість тестування додатків безпосередньо на пристроях з різних платформ.

Безпосередню увагу також потрібно приділити перевагам і недолікам всіх вище перечислених SDK.

Переваги Android SDK:

– Android SDK надає розробникам безкоштовні та легко доступні інструменти для розробки додатків для платформи Android;

– SDK містить велику кількість інструментів, таких як редактори ресурсів, емулятори, засоби відлагодження та інші, що дозволяє розробникам з легкістю створювати та тестувати додатки;

– наявність великої кількості документації (рисунок 1.5) та ресурсів, що допомагає розробникам швидко засвоїти необхідні знання та навички.

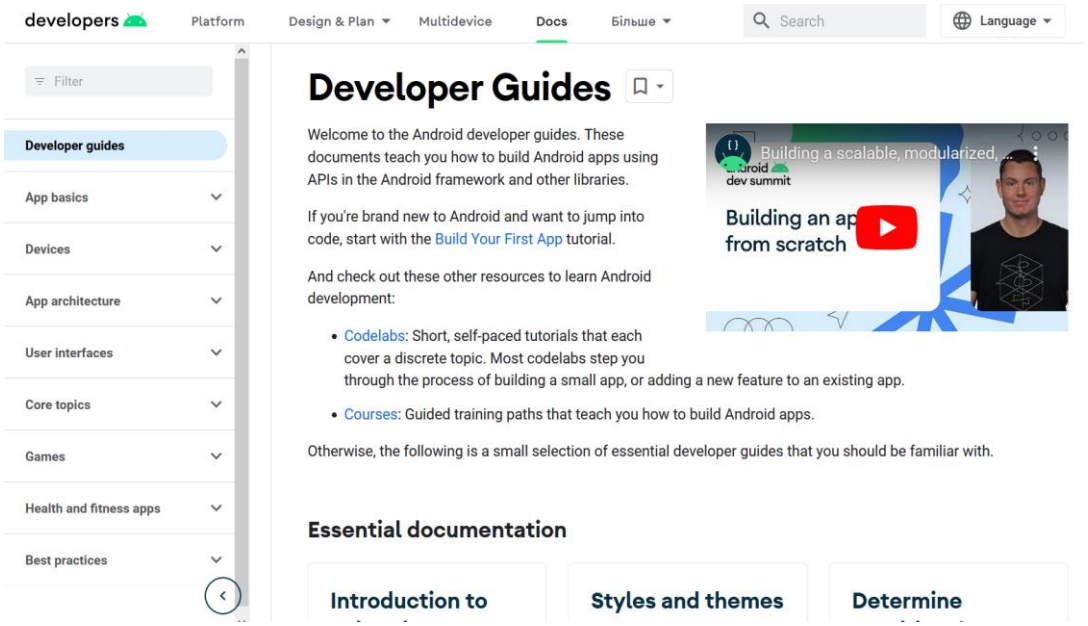

Рисунок 1.4 – Приклад документації Android Studio

Недоліки:

– розробка додатків для платформи Android може бути досить складною та часоємною задачею, особливо для початківців;

– низька єдність платформи та різні версії Android можуть створювати проблеми з сумісністю та тестуванням додатків;

– різноманітність пристроїв та їх характеристик можуть впливати на продуктивність додатків та їх сприйняття користувачами.

Основні переваги iOS SDK:

– висока якість та безпека додатків. iOS SDK дозволяє створювати додатки з високою продуктивністю та надійністю, завдяки обмеженому набору пристроїв та операційних систем, що підтримуються;

– широкий функціонал. iOS SDK має багатий набір функцій для розробки додатків, включаючи можливості роботи зі звуком, зображеннями, відео, сенсорами, GPS та іншими компонентами пристрою;

– простота розробки інтерфейсу. Інструментарій для розробки інтерфейсу користувача iOS SDK дозволяє створювати додатки зі стильним та зручним інтерфейсом;

– зручна інтеграція з іншими сервісами Apple. iOS SDK дозволяє легко інтегрувати додатки з іншими сервісами Apple, такими як iCloud, Apple Music, Apple Pay та інші.

Недоліки:

– обмежені можливості розробки кросплатформенних додатків. iOS SDK спеціалізований на розробці додатків для платформи iOS, що обмежує можливості кросплатформного розробки;

– високі вимоги до жорсткого та програмного забезпечення. Розробка додатків для iOS вимагає потужних комп'ютерів та високоякісного програмного забезпечення, що може бути великою витратою для початківців;

– суворі вимоги до додатків. Apple дотримується суворих вимог до додатків, які дозволяють публікуватися в App Store, що може бути важко для розробників, які не мають досвіду роботи з цією платформою;

– високі витрати на розробку та підтримку додатків. Розробка додатків для iOS може вимагати велики.

Переваги React Native SDK:

– React Native SDK дозволяє розробляти кросплатформні мобільні додатки з використанням JavaScript та React-підходу до розробки, що дозволяє забезпечити високу продуктивність та швидкість розробки;

– завдяки засобам гарячої заміни коду (hot reloading) можна побачити результат змін без необхідності перезапуску додатку (рисунок 1.5);

– React Native SDK має велику спільноту розробників, що забезпечує швидкий розвиток та підтримку інструменту;

– є можливість використання сторонніх модулів та компонентів, що дозволяє зменшити час розробки та забезпечити високу якість додатку.

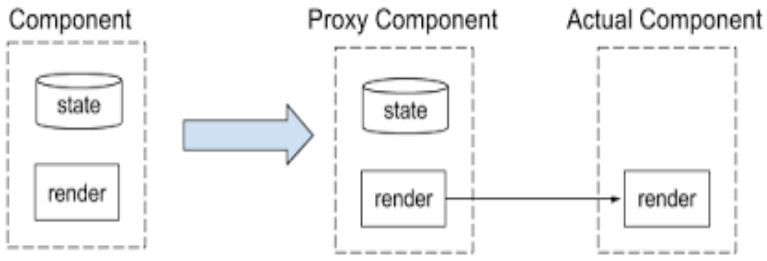

Рисунок 1.5 – Проста схема роботи hot reloading

Недоліки:

– хоча React Native SDK забезпечує високу продуктивність та швидкість розробки, в порівнянні з нативною розробкою, його продуктивність може бути меншою;

– навіть з гарячою заміною коду, процес розробки може бути повільнішим, ніж в нативній розробці;

– деякі функції можуть бути обмеженими або не доступними в React Native SDK, що може призвести до складнощів при розробці деяких додатків;

– розробка додатків за допомогою React Native SDK може вимагати певного рівня знань та навичок програмування;

– React Native SDK має певні обмеження щодо графічного інтерфейсу, що може призвести до складнощів при розробці деяких додатків.

Основні переваги Xamarin SDK:

– кросплатформність. Xamarin дозволяє створювати кросплатформні додатки, що працюють на різних платформах, включаючи iOS, Android та Windows;

– максимальна повторна використаність коду. Xamarin дозволяє розробляти додатки з використанням мови програмування C# та .NET Framework, що забезпечує максимальну повторну використовуваність коду та швидку розробку;

– швидкість розробки. Xamarin забезпечує швидку розробку додатків за рахунок вбудованого інструментарію для розробки інтерфейсу користувача та можливості тестування додатків безпосередньо на пристроях з різних платформ;

– можливість використання спільного коду. Використання спільного коду дозволяє зменшити час розробки та знизити витрати на створення додатків для кількох платформ;

– стабільність додатків. Xamarin забезпечує стабільність додатків та високу продуктивність.

Основні недоліки Xamarin SDK:

– платність. Для комерційних проектів необхідно придбати платну ліцензію, що може стати проблемою для невеликих компаній або розробників-фрілансерів;

– обмежена підтримка деяких платформ. Підтримка деяких платформ може бути обмеженою, що може стати проблемою для деяких проектів;

– вимоги до знань та навичок програмування. Розробка додатків за допомогою Xamarin може вимагати певного рівня знань та навичок програмування, що може бути складним для початківців.

1.4 Форми штучного інтелекту для генерування тексту

Штучний інтелект має багато форм, які можуть бути використані для генерації тексту. Деякі з них включають:

– глибинне навчання (deep learning) – це форма машинного навчання, що використовує нейронні мережі для знаходження складних зв'язків у даних. Глибинне навчання може бути використане для генерації тексту, якщо нейронна мережа навчена на великій кількості текстових даних;

– моделі марковських ланцюгів (Markov chain models) – це статистичні моделі, що використовуються для прогнозування майбутнього стану на основі поточного стану. Ці моделі можуть бути використані для генерації тексту, якщо вони навчені на текстових даних;

– генеративні віджети (Generative widgets) – це інтерактивні інтерфейси, що дозволяють користувачам взаємодіяти з моделями штучного інтелекту та генерувати текст. Генеративні віджети можуть бути навчені на текстових даних та використовуються для генерації тексту згідно з вказаними користувачем параметрами;

– рекурентні нейронні мережі (Recurrent neural networks) – це форма нейронних мереж, що використовуються для обробки послідовностей даних, таких як текст. Рекурентні нейронні мережі можуть бути навчені на текстових даних та використовуватися для генерації тексту на основі попереднього контексту;

– глибокі автокодувальні мережі (Deep autoencoder networks) – це форма нейронних мереж, що використовуються для зменшення розмірності

даних. Глибокі автокодувальні мережі можуть бути використані для зменшення розмірності текстових даних та використовуватися для генерації нового тексту;

– згорткові нейронні мережі (Convolutional neural networks) – це форма нейронних мереж, що використовуються для обробки зображень, але також можуть бути використані для обробки тексту. Згорткові нейронні мережі можуть бути навчені на текстових даних та використовуватися для генерації тексту на основі контексту;

– статистичні моделі (Statistical models) – це моделі, що використовуються для статистичного аналізу даних. Статистичні моделі можуть бути використані для генерації тексту на основі статистичного аналізу текстових даних;

– трансформерні мережі (Transformer networks) – це форма нейронних мереж, що використовуються для обробки послідовностей даних. Трансформерні мережі можуть бути навчені на текстових даних та використовуватися для генерації тексту згідно з вказаними параметрами.

Зважаючи на те, що кожна з форм штучного інтелекту має свої переваги і недоліки, нижче я перерахую недоліки деяких з них:

а) глибинні нейронні мережі:

– вимагають великої кількості даних для тренування, що може бути витратним та часомірним процесом;

– під час генерації тексту можуть створювати не стійкі та неповторні варіанти, які можуть бути важко контролювати;

– можуть мати складну структуру та велику кількість параметрів, що потребують високопродуктивного обладнання;

б) Марковські моделі:

– можуть створювати текст, що не має логічного зв'язку або змістовної цінності;

– залежно від ступеню складності можуть вимагати велику кількість пам'яті та обчислювальних ресурсів;

в) моделі на основі правил:

– вимагають великої кількості знань та експертизи у конкретній предметній галузі;

– можуть мати обмежені можливості генерації тексту, оскільки вони базуються на заданих правилах;

– під час використання в реальному часі можуть потребувати постійного оновлення правил;

г) генеративно-супер нечесні моделі:

– можуть створювати текст з помилками або неточностями через використання непередбачуваної логіки;

– можуть потребувати великої кількості обчислювальних ресурсів під час генерації великої кількості тексту.

Вибір пав на моделі Transformer, що здатні генерувати тексти на основі вихідних даних, які можуть включати в себе інформацію про питання, що ставляться (рисунок 1.6). Такі моделі зазвичай навчаються на великій кількості текстів, тому можуть бути досить точними в генерації текстів відповідно до питань, які Вам задають. Крім того, їх можна покращити за допомогою налаштування на конкретному наборі даних, що дозволить досягти кращих результатів у генерації текстів на основі запитань. Тому, варто враховувати, що навчання таких моделей може вимагати багато ресурсів та часу для досягнення якісних результатів.

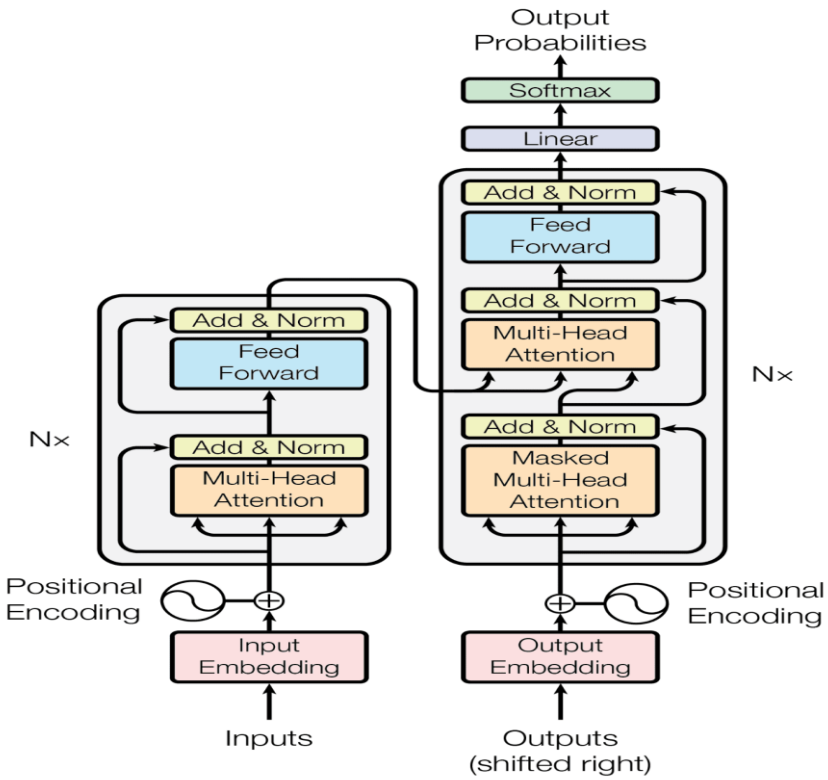

Рисунок 1.6 – Перегляд композиції відразу у грі

Проте, важливо мати на увазі, що генерація текстів на основі запитань може бути складним завданням, оскільки потрібно зрозуміти суть запитання та знати, яку саме відповідь потрібно генерувати.

2 ПОСТАНОВКА ЗАДАЧІ

2.1 Задача роботи

Після вивчення предметної галузі (розділ 1), було зроблено висновок про активність та актуальність ніші систем генерації тексту. Проаналізувавши аналогічні додатки, було встановлено, що багато проектів, представлених аудиторії, не є завершеними.

Відповідно, було прийнято рішення розробити систему допомоги генерації тексту, яка базується на введених даних. Це завдання є актуальним, оскільки наявні аналоги мають обмежені можливості і можуть бути поліпшені за допомогою впровадження інших моделей штучного інтелекту.

Метою даного проекту є – розробка мобільного додатку на платформі Android з використанням мови програмування Dart/Flutter та трасформерної моделі GPT для генерації дороблених запитань. Додаток буде мати наступні функції:

– вхід: Користувач вводить своє запитання в текстовому форматі;

– обробка запитання: Додаток використовує GPT-модель для обробки запитання та генерації до робленого запитання, що має на увазі користувача;

– відповідь: Додаток відображає дороблене запитання користувача на екрані.

Для реалізації цієї задачі необхідно виконати наступні кроки:

– збір та підготовка даних: Зібрати достатню кількість даних, які будуть використовуватися для тренування GPT-моделі. Для цього можна використовувати публічні датасети, такі як SQuAD, або скласти власний набір даних;

– розробка моделі: Натренувати модель GPT з використанням Tensorflow або PyTorch. Після тренування моделі, необхідно провести її тестування та налаштування;

– розробка мобільного додатку: Розробити мобільний додаток на платформі Android з використанням мови програмування Dart/Flutter. У додатку необхідно реалізувати функції вводу запитання та відображення дробленого запитання на екрані;

– інтеграція GPT-моделі в додаток: Після розробки мобільного додатку необхідно забезпечити інтеграцію тренованої GPT-моделі в додаток.

Основна мета додатка полягає в поліпшенні розуміння користувача у питаннях що він ставить. Для цього застосунок створює інформаційний хвилю, який формується за допомогою алгоритмічної частини додатка. Однак, додаток повинен також мати допоміжні можливості, наприклад, швидка переробка відповіді, що сприяє зручності користування застосунком.

При розробці додатка необхідно пам'ятати про створення інтуїтивної та приємної для користувача оболонки, щоб полегшити його користування додатком. Для цього, додаток повинен мати зрозумілу структуру, зроблену на основі потреб та очікувань користувачів, яка містить чіткі та зрозумілі елементи керування, такі як кнопки та посилання.

Оскільки створення повноцінної версії додатка з усіма додатковими можливостями в один цикл розробки є складною задачею, у цій роботі буде розглянуто перший цикл розробки, що включатиме запровадження генеративних моделей та першої версії додатка у вигляді прототипу. Також будуть розглянуті майбутні можливості розвитку та удосконалення додатка.

Отже, структура для дослідження та розробки додатка є чіткою та максимально зручною для його розробки, що дозволить ефективно використовувати час при створенні проекту.

Оскільки розробка даної системи з усіма додатковими можливостями потребує кількох циклів розробки та тестування серед користувачів, розробка у рамках одного циклу є складною та не доцільною задачею. Тому буде розглянуто перший цикл розробки, який включає вміщення

можливості генераційного тексту та першу версію додатка у вигляді прототипу. Майбутні можливості розвитку та удосконалення додатка також будуть розглянуті. Отже, маємо чітку структуру для дослідження та подальшої розробки додатка, що забезпечує максимально зручний цикл його розробки. Використовуючи це, можна максимально ефективно використовувати час для створення проекту.

2.2 Збір даних про запити користувача

Збір даних користувачів, що використовують генеративні моделі для отримання відповідей, є складним завданням, яке вимагає від розробників дбайливості та поваги до приватності користувачів. Для збору даних можна використовувати різноманітні методи, включаючи автоматичний збір даних, опитування та аналіз взаємодії користувачів з системою.

Один із способів збору даних – це автоматичний збір. Цей метод включає в себе використання технік моніторингу та відстеження поведінки користувачів. Наприклад, можна встановити спеціальний плагін у веб-браузері користувача, який збиратиме дані про взаємодію користувача з генеративною моделлю, такі як введені запити та отримані відповіді. Однак, цей метод збору даних може порушувати приватність користувача та призводити до конфіденційності даних.

Інший метод збору даних – це опитування. Цей метод включає в себе створення опитувальників для збору даних про користувачів, такі як їх вік, стать, мову та інші демографічні дані. Опитування можуть бути проведені онлайн і офлайн, та можуть містити як закриті, так і відкриті питання. Однак, важливо пам'ятати, що дані, отримані через опитування, можуть бути неповними або неправильними через неправильне розуміння запитань або бажання користувачів давати неправдиві відповіді.

Третій метод збору даних – це аналіз взаємодії користувачів з системою. Цей метод включає в себе аналіз взаємодії користувачів з

системою – це метод, який вимагає від розробників генеративної моделі збирати дані про взаємодію користувачів з системою. Для цього можна використовувати різноманітні інструменти, такі як системи аналітики веб-трафіку, системи відстеження поведінки користувачів та інші.

Одним зі способів аналізу взаємодії користувачів з системою є відстеження того, які запити користувачі роблять до генеративної моделі та які відповіді вони отримують. Цей метод дозволяє збирати дані про те, як користувачі використовують генеративну модель, та зробити висновки щодо ефективності моделі.

Інший спосіб аналізу взаємодії користувачів з системою – це використання систем аналітики веб-трафіку, таких як Google Analytics або інші аналітичні сервіси. Ці системи дозволяють збирати дані про поведінку користувачів на сайті, такі як час перебування на сайті, кількість переглядів сторінок та інші метрики. Ці дані можуть допомогти розробникам генеративної моделі зрозуміти, які частини сайту користувачі відвідують найчастіше та як вони взаємодіють з генеративною моделлю приклад роботи Google Analytics у якій є можливість переглядати різні типи графіків, що можуть свідчити про будь-яку інформацію можно побачити далі (рисунок 2.1).

Рисунок 2.1 – Приклад аналітики Google Analytics

Незалежно від методу збору даних, важливо пам'ятати про приватність користувачів та дотримуватися всіх необхідних правил та регуляцій щодо збору та обробки особистих даних. Для захисту приватності

користувачів можуть використовуватися такі методи, як анонімізація даних та шифрування. Також важливо повідомляти користувачів про те, які дані збираються та як вони будуть використовуватися. Для цього можна використовувати політики конфіденційності та угоди про використання, які містять інформацію про те, які дані збираються, як вони будуть використовуватися та які заходи захисту приватності будуть прийняті.

Нарешті, важливо пам'ятати про етичність збору даних користувачів. Розробники генеративних моделей повинні дотримуватися етичних принципів та не зловживати отриманою інформацією. Важливо забезпечити безпеку та конфіденційність даних користувачів та не використовувати їх для будь-яких незаконних або шкідливих цілей.

Для того, щоб почати користуватись можливостями генеративної системи типу GPT треба запросити API-ключ, що дозволяє стати розробником.

API-ключ OpenAI – це унікальний ідентифікатор, який дає вам доступ до можливостей OpenAI API. API-ключ дозволяє вам взаємодіяти з моделями глибокого навчання OpenAI і отримувати результати обробки наших алгоритмів штучного інтелекту. Важливість ключа розробника OpenAI полягає в тому, що він дає вам можливість використовувати потужні інструменти машинного навчання для розв'язання задач у вашому проекті. За допомогою API-ключа OpenAI ви можете запустити моделі, знайти відповіді на ваші запитання, створювати нові контенти, перекладати тексти та багато іншого взяття ключа можна побачити далі (рисунок 2.2).

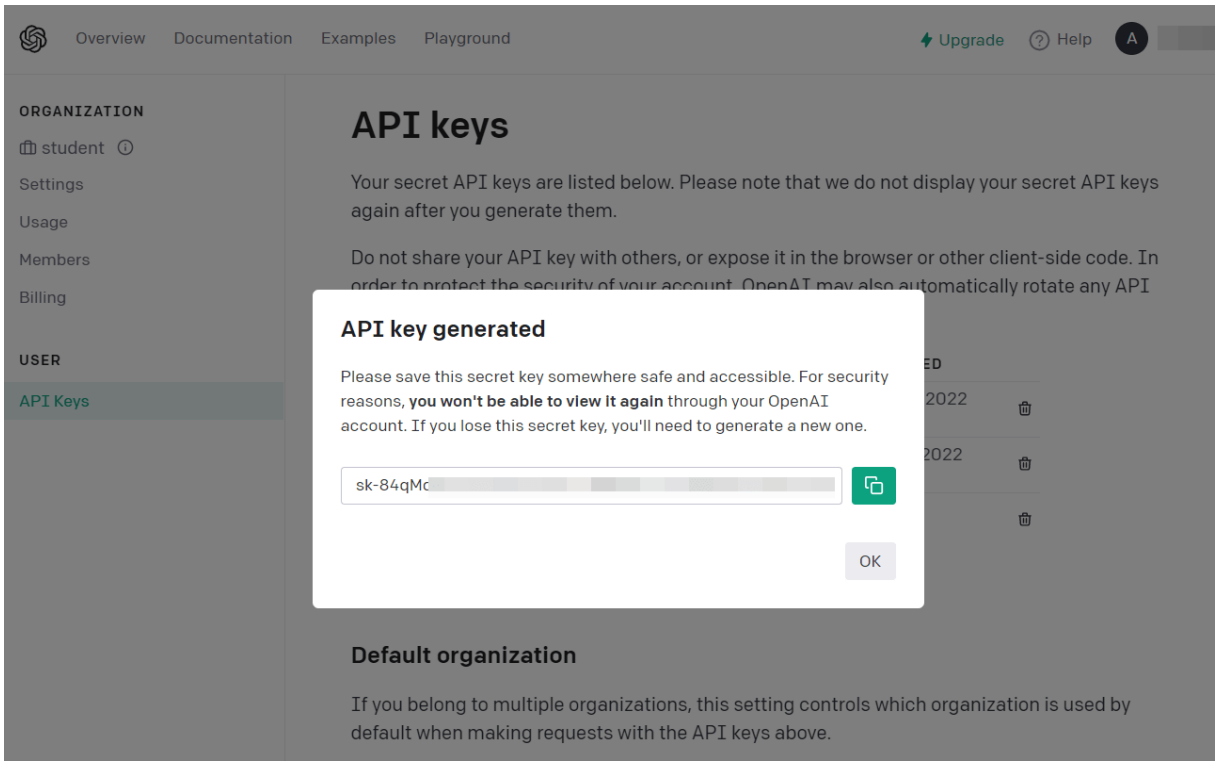

Рисунок 2.2 – Генерація ключа розробника

Отримання ключа розробника OpenAI є важливим кроком для використання моделей глибокого навчання та іншої функціональності OpenAI API. З ключем розробника ви отримуєте доступ до різноманітних інструментів машинного навчання, які можуть бути використані для створення нових продуктів, покращення існуючих та розв'язання бізнес-задач.

Отримання ключа розробника OpenAI дозволяє вам отримати доступ до різних функцій та можливостей OpenAI API. До чого саме можна отримувати доступ залежить від того, яку мету можна переслідувати. Ось деякі приклади того, до чого саме ви можете отримати доступ з ключем розробника OpenAI:

– моделі глибокого навчання: з ключем розробника ви можете запускати моделі глибокого навчання OpenAI, такі як GPT-3, для генерації текстів, перекладу мови, відповідей на запитання та інші;

– синтез тексту: з допомогою OpenAI API ви можете створювати унікальний текст для вашого веб-сайту, блогу, соціальної мережі та інші;

– зображення: OpenAI API дозволяє вам здійснювати обробку зображень та використовувати їх в вашому проекті;

– переклад: з ключем розробника ви можете використовувати OpenAI API для перекладу текстів на більше 50 мов;

– інші функції: OpenAI API також надає інші функції, такі як розпізнавання мови, класифікацію даних, рекомендації та інші.

В цілому, отримання ключа розробника OpenAI дає вам доступ до потужних інструментів машинного навчання, які можна використовувати для розв'язання багатьох завдань в вашому проекті. Варто зазначити, що під час використання OpenAI API необхідно дотримуватися правил та обмежень, встановлених OpenAI, та поважати права інтелектуальної власності. у цілому можна описати процес збору даних далі (рисунок 2.3).

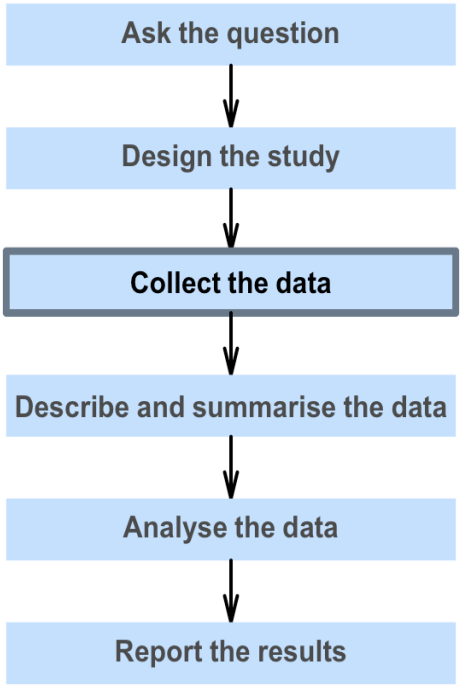

Рисунок 2.3 – Процес збору даних

2.3 Опис та правила аналізу тексту, контент-аналіз

Аналіз тексту – це процес визначення та розуміння смислу тексту, який може бути виконаний за допомогою різних методів та технологій. Дослідження текстів можуть включати в себе визначення ключових слів, тематичних елементів, структури, взаємозв'язків між різними частинами тексту, аналізу стилістичних особливостей та інші показники.

Основні правила аналізу тексту включають:

– читайте текст докладно. Ви повинні звернути увагу на кожне речення та кожне слово, щоб зрозуміти повний контекст;

– розумійте ціль тексту. Важливо зрозуміти, для чого було написано цей текст, яку мету він переслідує та якою мовою він написаний;

– аналізувати структуру тексту. Розгляньте заголовки, підрозділи, ключові слова, абзаци та їх послідовність, щоб зрозуміти, як вони взаємодіють та доповнюють один одного;

– визначте ключові слова та терміни. Визначте терміни та фрази, які використовуються автором, та спробуйте з'ясувати їхній смисл;

– розгляньте контекстуальні нюанси. Зверніть увагу на те, як слова та фрази використовуються в контексті, в якому вони з'являються, оскільки вони можуть мати різні значення в різних контекстах;

– встановіть зв'язки та відношення. Спробуйте встановити зв'язки між різними частинами тексту, щоб зрозуміти, як вони пов'язані між собою.

Для аналізу тексту необхідно мати чітке розуміння його структури та змісту. Перш за все, варто звернути увагу на заголовок та підзаголовки, які дають загальне уявлення про тему та структуру тексту. Далі, необхідно ретельно прочитати текст, намагаючись зрозуміти головну ідею та деталі, які підтримують цю ідею.

Один з важливих аспектів аналізу тексту - це розуміння контексту. Текст може містити кілька ідей та підтем, тому важливо розуміти, як вони пов'язані між собою та який є загальний контекст, що об'єднує їх.

Також варто звернути увагу на терміни та поняття, які використовуються в тексті. Якщо їх не зрозуміти, то можливо неправильно зрозуміти контекст або не зрозуміти загальну ідею. Якщо термін незнайомий, то його можна знайти в словнику або пошуковій системі.

При аналізі тексту також важливо звернути увагу на зв'язки між реченнями та абзацами. Вони допомагають розуміти, як ідеї пов'язані між собою та як вони розвиваються. Наприклад, ключові слова та фрази можуть повторюватися, що підкреслює їх важливість та пов'язаність з головною ідеєю тексту.

Нарешті, при аналізі тексту варто звернути увагу на тон та стиль. Текст може мати формальний або неформальний стиль, або бути спрямований на різні аудиторії. Тон тексту також може бути досить різним від об'єктивного до суб'єктивного, від наукового до гумористичного. Розуміння тону та стилю допоможе краще зрозуміти контекст Також важливо звертати увагу на контекст, в якому вживається слово чи фраза, оскільки він може змінювати їх значення. Наприклад, слово «банк» може мати різне значення в контексті розмови про фінансову установу та про прибережну частину річки.

При аналізі тексту також важливо звертати увагу на емоційний відтінок тексту. Наприклад, одна й та ж фраза може звучати різними тонами в залежності від інтонації, якою вона вимовляється.

Узагалі, аналіз тексту – це складний процес, що вимагає від аналітика вміння читати між рядків, розуміти мовні засоби, контекст, емоції та інші аспекти тексту. Це необхідна навичка для багатьох професій, пов'язаних із збором та аналізом даних, маркетингом, лінгвістикою, журналістикою та іншими сферами.

Контент-аналіз – це процес систематичного аналізу текстового, візуального та/або аудіо-матеріалу. Є декілька етапів в контент-аналізі, серед яких:

– визначення дослідницької проблеми: цей етап включає в себе визначення дослідницької проблеми або питання, яке дослідник прагне відповісти під час аналізу контенту;

– визначення вибірки: цей етап передбачає вибірку контенту для аналізу. Вибірка може бути випадковою, систематичною або за певними критеріями;

– кодування: цей етап включає процес встановлення кодів або категорій для опису відповідей на дослідницьке питання. Коди можуть бути кількісними або якісними, залежно від дослідницької проблеми;

– конструювання кодового каталогу: на цьому етапі вибір кодів або категорій для опису відповідей. Кодовий каталог визначається на основі дослідницької проблеми та попереднього досвіду дослідника;

– навчання кодерів: у випадку, якщо кілька людей будуть працювати з контентом, дослідник повинен забезпечити навчання кодерів. Це включає в себе пояснення дослідницької проблеми, опис кодів та інструкції для кодування;

– кодування: на цьому етапі дослідники здійснюють кодування відповідей на питання;

– перевірка достовірності: цей етап включає перевірку достовірності аналізу, щоб забезпечити точність дослідних результатів;

– аналіз результатів: на цьому етапі дослідники вивчають результати аналізу і намагаються зрозуміти, що вони означають.

Вони можуть порівнювати отримані дані зі знаннями, які вони вже мають про досліджувану тему, і шукати закономірності та зв'язки. Також можуть використовувати статистичні методи для обробки даних і отримання показників, які допоможуть зрозуміти результати дослідження етапи контент-аналізу (рисунок 2.4).

Рисунок 2.4 – Етапи контент-аналізу

У підсумку, контент аналіз є досить важливим інструментом у багатьох галузях, включаючи наукові дослідження, маркетинг та політичні кампанії. Цей процес дозволяє дослідникам збирати та аналізувати великі обсяги текстової інформації, здійснювати категоризацію та класифікацію даних, визначати ключові теми та тренди. Важливим етапом є розробка кодувальної схеми, що допомагає стандартизувати процес аналізу, забезпечує точність та надійність отриманих результатів. Результати контент аналізу можуть бути використані для розробки бізнес-стратегій, формулювання рекомендацій, а також для проведення подальших досліджень. Однак, необхідно бути обережним у виборі методів аналізу та інтерпретації результатів, щоб уникнути спотворення або невірного розуміння даних.

3 ВИБІР МОДЕЛІ ДЛЯ ВИКОРИСТАННЯ

3.1 Огляд

Як вже було сказано раніше, основними проблемами, які повинні бути вирішеними, є побудова рекомендаційного блоку та системи генерування відповіді користувачу з його поточним питанням у додатку. Користувач має змогу переписати відповідь за поточним згенерованим питанням та рекомендації того, які з перелічених слів можуть краще за все доповнити запитання. В додатку не завжди можна отримати відповідь, одразу та дуже часто потрібно перегенеровувати відповідь. тому додаток повинен звертати увагу не генерувати окремі відхилені рішення до колишньої відповіді.

Почнемо дослідження з генеративної системи та того, що для її подальшої побудови треба буде зробити.

У сучасному світі, де інформація є великою частиною нашого повсякденного життя, вміння розуміти та аналізувати її стає все важливішим. Один з найефективніших способів покращення розуміння оточуючого світу – це використання запитань. Запитання допомагають нам зосередитися на ключових аспектах інформації, з’ясувати деталі та переконатися, що ми розуміємо її належним чином. Використання запитань може бути корисним і в освітніх цілях, наприклад, у процесі вивчення нового матеріалу. Задавання запитань допомагає студентам зосередитися на ключових поняттях та допомагає їм згадати важливу інформацію під час екзамену. Крім того, запитання можуть бути корисним інструментом для покращення міжособистісних відносин. Задавання запитань допомагає нам краще зрозуміти інших людей, їхні переконання та мотивацію, що може підвищити нашу емпатію та зменшити конфлікти. Зазначимо, що сучасні технології, такі як GPT (Generative Pre-trained Transformer), базуються на аналізі великих обсягів текстової інформації та використовують запитання як один з ключових елементів при розумінні мови та генерації тексту. GPT

використовує глибокі нейронні мережі для розуміння контексту та здатності генерувати зрозумілі відповіді на запитання користувачів. Такі технології відкривають нові можливості в різних сферах, таких як автоматизація клієнтського сервісу, медична діагностика та багато іншого. Інформаційна система первинної підготовки запитів до GPT є програмним забезпеченням, яке допомагає користувачам ефективно підготувати запити до системи GPT [1]. Система може містити інтерфейс користувача та функціональність автоматичної обробки тексту, щоб забезпечити оптимальну підготовку запиту до передачі до GPT [2]. Інформаційна система може включати такі функції, як передбачення запиту, автоматичне заповнення запиту та виправлення помилок [3]. Використання інформаційної системи первинної підготовки запитів до GPT може бути корисним в різних сферах, де потрібні ефективні та якісні запити до системи GPT [4]. Розвиток та вдосконалення інформаційної системи первинної підготовки запитів до GPT може покращити роботу з системою GPT та забезпечити більш точні та продуктивні результати [5].

Для того, щоб задати правильний запитання GPT, варто врахувати кілька рекомендацій:

– сформулюйте своє запитання як можливо більш точно і конкретно. Чим більш детально ви описуєте своє запитання, тим більш вірогідно, що відповідь буде корисною та зрозумілою;

– спробуйте використовувати прості слова та фрази. Завдяки цьому, система краще зрозуміє, що саме ви хочете дізнатися;

– для уточнення запитання можна використовувати додаткові ключові слова та питальні слова, наприклад, «як», «що», «чому» і т.д;

– спробуйте від формулювати запитання таким чином, щоб його можна було відповісти одним словом або фразою. Наприклад, замість запитання «Можете дати мені детальний опис процесу виготовлення піци?» краще сформулювати «Як приготувати піцу?»;

– не перекладайте на машинний переклад питання з інших мов. Якщо ви не впевнені в своєму знанні мови, краще скористатися допомогою перекладача або запитати у співрозмовника, який володіє мовою краще вас.

Враховуючи ці рекомендації, ви можете задавати більш точні та корисні запитання GPT, що дозволяє отримувати більш якісні та зрозумілі відповіді.

3.2 Детальний опис GPT

Generative Pre-trained Transformer (GPT) є однією з найбільш відомих та ефективних архітектур для генерації текстів. Принцип роботи GPT полягає у тому, що модель навчається на великому обсязі текстів, після чого вона може генерувати текст відповідно до заданого контексту.

Основою GPT є трансформер, який дозволяє моделі ефективно працювати з довгими текстами та зберігати контекстну інформацію. Трансформер складається з двох основних частин: єнкодера та декодера. Энкодер отримує на вхід послідовність векторів слів та перетворює їх у приховані вектори. Декодер отримує на вхід приховані вектори та генерує нову послідовність векторів слів.

Архітектура GPT має лише декодер і працює за принципом авторегресивної моделі. Це означає, що модель генерує текст по одному слову за раз, з урахуванням попереднього згенерованого слова. При цьому модель намагається знайти таку послідовність слів, щоб максимізувати ймовірність з'явлення наступного слова в тексті.

Одна з головних особливостей GPT – це те, що модель навчається на великому обсязі текстів, а саме, на т.з. корпусах текстів. При цьому, GPT може використовувати різні джерела текстів, наприклад, Вікіпедію, новини, книги, блоги та інші джерела. Навчання моделі GPT зазвичай проводиться на великих кластерах серверів з використанням різних технік, таких як паралельне навчання та дистрибутивне навчання.

Отже, GPT може виконувати різноманітні завдання, такі як генерація тексту, переклад мови, відповіді на запитання та інше. Для цього модель отримує на вхід послідовність слів (текст) та намагається передбачити наступне слово в послідовності. Щоб зробити це, модель використовує декілька шарів нейронів, які обробляють вхідні дані та генерують вихідну послідовність слів.

Кожен шар має свій власний набір вагів, які використовуються для обчислення значень наступного шару. Крім того, кожен шар використовує функцію активації, яка допомагає моделі вирішувати нелінійні завдання та передавати інформацію між шарами.

Під час тренування моделі використовуються тисячі текстових даних, які модель навчається передбачати. Процес тренування може займати від декількох днів до кількох тижнів, залежно від розміру моделі та обсягу даних.

Після тренування, модель може бути використана для генерації тексту, перекладу мови, відповідей на запитання та інші завдання. Крім того, модель може бути доопрацьована на конкретному наборі даних, щоб поліпшити її роботу у певному контексті.

Важливо зауважити, що GPT є лише однією з багатьох архітектур нейронних мереж, які можуть виконувати завдання генерації тексту. Кожна з цих архітектур має свої переваги та недоліки та може бути більш чи менш ефективною в різних задачах.

Продовжуючи розповідь про принцип роботи GPT, можна сказати, що модель працює за допомогою глибокого навчання (deep learning) з використанням нейронних мереж. Для навчання моделі використовують великі обсяги тексту, які проходять через набір функцій, що називається трансформером (transformer).

Трансформер розбиває текст на послідовності токенів (слів або символів), які потім кодуються у векторному форматі. Застосовуючи механізм само зауважень (self-attention), трансформер обробляє ці вектори

та визначає, які токени мають більший вплив на виведення наступного токену у послідовності. Таким чином, трансформер може враховувати залежності між словами у тексті та вирішувати завдання згідно з контекстом.

GPT використовує так звану «маскування з майбутнім» (masking of future tokens), що означає, що модель може мати доступ тільки до попередніх токенів у тексті, а не до наступних. Це допомагає моделі під час генерації тексту вирішувати завдання за контекстом та уникати повторень.

Крім того, GPT має декілька шарів з різною кількістю нейронів, які дозволяють моделі адаптуватися до складніших завдань та збільшувати точність виведення тексту.

Узагалі, принцип роботи GPT досить складний та базується на великому обсязі даних, що використовуються для навчання моделі. Однак, завдяки цим особливостям, GPT дозволяє генерувати високоякісний текст, який може використовуватися в багатьох галузях, від розробки програмного забезпечення до медіа та маркетингу.

GPT використовує метод глибокого навчання, який дозволяє йому автоматично вивчати складні залежності в даних. Це забезпечує високу точність та якість генерованого тексту, особливо у порівнянні з іншими методами генерації тексту.

Однак, налаштування та тренування GPT може бути досить складним процесом, який вимагає великої кількості даних та обчислювальних ресурсів. Хоч і може здатися на перший взгляд, що це не така велика проблема, але потрібно враховувати бюджет та фахівців, деякі з них можуть бути недостатньо компетентні тобто без досвіду або ті хто може вдавати це. Тому, багато компаній та організацій звертаються до послуг з налаштування та тренування GPT від провідних фахівців у цій галузі приклад зі збору даних у трьох етапах, можна побачити далі (рисунок 3.1).

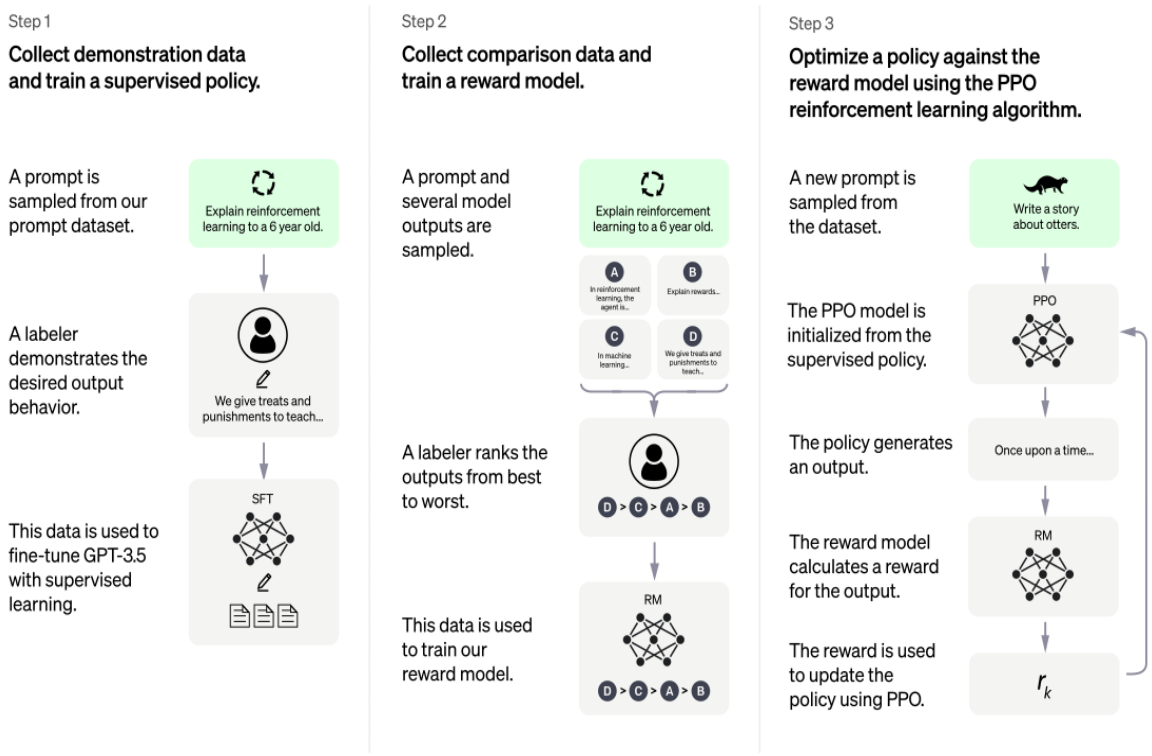

Рисунок 3.1 – Етапи збору даних моделі GPT

Після цього GPT може використовуватися для генерації тексту на основі вхідного контексту. Зокрема, вона може продовжувати незавершені речення або генерувати новий текст на основі податкового запиту.

3.3 Різниця між версіями GPT

Перша версія GPT була представлена в 2018 році і мала 117 мільйонів параметрів. Вона була натренована на великому корпусі текстових даних і могла генерувати текст, відповідний контексту. Однак, вона не могла працювати з довшими текстами і мала тенденцію до збурення на високих рівнях абстракції.

Друга версія GPT, яка була випущена в 2019 році, мала 1,5 мільярда параметрів. Вона була тренована на ще більшому корпусі текстових даних, що дозволило їй генерувати текст, більш зв'язний і правдоподібний, а також забезпечувати кращу здатність розуміти довші контексти.

Третя версія GPT була випущена в 2020 році з назвою GPT-3 і має 175 мільярдів параметрів. Це найбільша модель GPT до цього часу. Вона була натренована на величезному обсязі текстових даних і може генерувати довільні тексти, включаючи новини, вірші, інструкції, відгуки, статті тощо. GPT-3 також може виконувати різні завдання, такі як переклад текстів, заповнення пропущених слів, створення програмного коду і багато іншого.

Одна з ключових відмінностей між різними версіями GPT – це кількість параметрів. Кожна нова версія містить більше параметрів, що дозволяє їй краще розуміти текстові дані і генерувати більш складні тексти. Крім того, кожна нова версія має покращену архітектура.

Основною різницею між різними версіями GPT є їхні характеристики, які визначають їх ефективність і можливості. Давайте розглянемо кілька ключових різниць між GPT, GPT-2 та GPT-3:Кожна версія GPT пройшла покращення в порівнянні з попередньою версією.

Різницю можна побачити далі:

– кількість параметрів: GPT мав близько 117 мільйонів параметрів, тоді як GPT-2 має близько 1,5 мільярдів параметрів, а GPT-3 має нейромережу з 175 мільярдами параметрів. Це означає, що GPT-3 має в 10 разів більше параметрів, ніж GPT-2 і в 1500 разів більше параметрів, ніж оригінальний GPT. Таким чином, більші кількості параметрів в GPT-2 та GPT-3 дозволяють їм вирішувати більш складні завдання та генерувати більш точні та різноманітні результати;

– розмір корпусу даних: GPT-2 та GPT-3 були треновані на набагато більшому обсязі текстових даних, ніж оригінальний GPT. GPT-2 був тренований на 40GB текстових даних, в той час як GPT-3 був тренований на 570GB текстових даних. Це дозволило їм вивчати більшій діапазон тем і мов;

– різноманітність завдань: GPT-2 та GPT-3 були натреновані на більшій кількості завдань, ніж GPT. Наприклад, GPT-2 був тренований на 7 різних завданнях, в той час як GPT-3 був тренований на 70 різних завданнях.

Це дозволяє GPT-2 та GPT-3 генерувати більш різноманітний і корисний контент для різних завдань;

– виконання завдань: GPT-3 має декілька додаткових можливостей в порівнянні з GPT-2 та GPT.

GPT-4, який став наступним кроком у розвитку нейронних мереж і може забезпечити значний прорив у генерації тексту та розумінні мови.

Проте, на даний момент використання GPT-4 є платним і доступним лише обмеженій кількості компаній та організацій. Це створює певні обмеження для багатьох дослідників та розробників, які б хотіли використовувати цей інструмент у своїх проектах.

Незважаючи на це, багато дослідників та розробників продовжують працювати з попередніми версіями GPT та іншими інструментами штучного інтелекту, які є безкоштовними та відкритими для використання. Це дозволяє їм займатися важливими дослідженнями та створювати корисні додатки для різних галузей, хоча і з певними обмеженнями.

Звичайно, сподіваємося, що в майбутньому GPT-4 буде доступним більш широкому колу дослідників та розробників, щоб забезпечити максимальний прорив у галузі штучного інтелекту та покращення якості нашого життя в цілому.

3.4 Підводні камені використання Open AI API

OpenAI API є потужним інструментом для розробки інтелектуальних додатків, проте його використання пов'язане з деякими підводними каменями. По-перше, використання API є платним, що може бути проблемою для певних компаній з обмеженим бюджетом. Крім того, є деякі обмеження на кількість запитів до API в місяць та на кількість результатів, що можуть бути отримані з кожного запиту (рисунок 3.2).

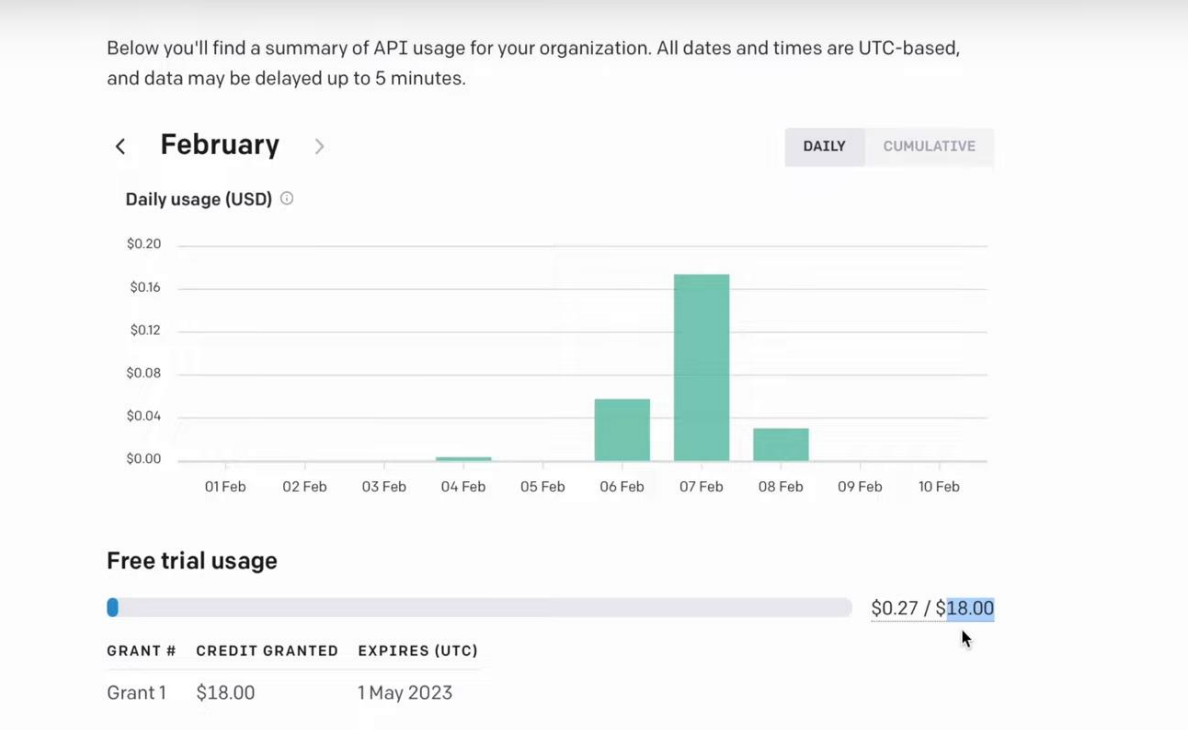

Рисунок 3.2 – Колаборативний фільтр

Крім того, OpenAI API має певні обмеження щодо тематики та контексту введеного тексту. Це може призвести до недостатньої точності результатів або неправильного розуміння тексту. Також існує ризик зловживання API шляхом створення дезінформаційних або шкідливих додатків.

Незважаючи на ці підводні камені, використання OpenAI API може принести значну користь для розробників додатків. Він забезпечує доступ до потужних інструментів машинного навчання та нейронних мереж, що дозволяє створювати інноваційні продукти, які можуть допомогти вирішувати складні проблеми в різних галузях, таких як медицина, фінанси, наука і технології.

4 МЕТОД ПОБУДОВИ ДОДАТКУ

4.1 Android Studio для розробки додатка

Блок архітектура – це підхід до розробки програмного забезпечення, який забезпечує легку і швидку розширюваність, тестування та підтримку коду. Вона забезпечує чітку організацію коду, що розділяє його на незалежні компоненти, які взаємодіють між собою за допомогою певного механізму передачі даних.

У Android Studio Flutter, для використання блок архітектури, рекомендовано використовувати пакет flutter_bloc. Цей пакет надає засоби для реалізації таких архітектур як BLoC (Business Logic Component) та Cubit, приклад блок архітектури можна побачити далі (рисунок4.1).

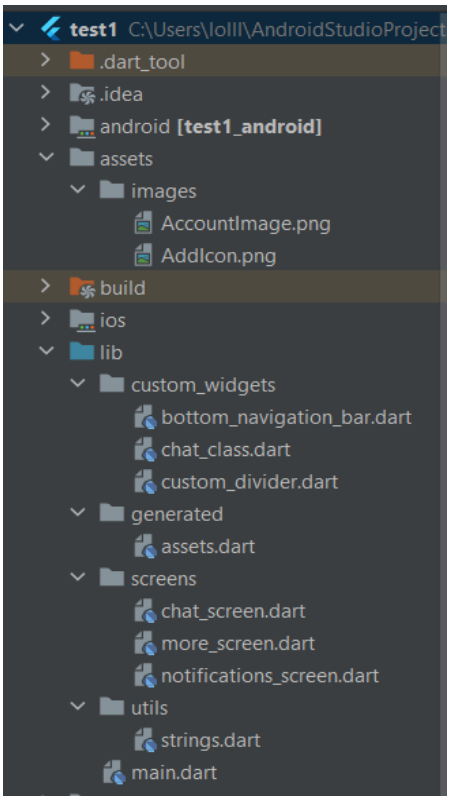

Рисунок 4.1 – Блок Архітектура додатка

BLoC – це підхід, при якому всі дані та події проходять через об'єкт BLoC, який відповідає за логіку бізнес-процесів в додатку. Крім того, BLoC відповідає за відображення даних в користувацькому інтерфейсі та реакцію на події, що виникають у додатку.

Cubit – це спрощена версія BLoC, яка забезпечує лише механізм управління станом додатку.

Використання блок архітектури у Android Studio Flutter дозволяє забезпечити високу якість коду, спрощення тестування та підтримки, що робить проект більш масштабованим та стійким до змін.

Main.dart є головним файлом у додатку Flutter. Він містить точку входу у додаток та описує структуру його інтерфейсу користувача. Код у файлі Main.dart виконується при запуску додатку і він містить клас, що наслідується від класу StateLessWidget, який є базовим класом для всіх віджетів у Flutter.

У файлі Main.dart можна виконувати різні завдання, наприклад, ініціалізацію додатку, створення та конфігурацію макету інтерфейсу, налаштування роутів, додавання різних пакетів та бібліотек, і т.д. Основна мета Main.dart – це ініціалізація додатку та створення дерева віджетів, які будуть використовуватися у програмі.

Main.dart є одним з найважливіших файлів у додатку Flutter, оскільки він визначає основну логіку та структуру додатку. Він є точкою входу, де Flutter стартує додаток та відображає головний екран додатку.

Крім того, у Main.dart можна встановлювати тему та стилі для додатку, додавати інші файли та ресурси, налаштовувати різні параметри та виконувати різні операції на початку запуску додатку.

Узагалі, файл Main.dart відповідає за створення та налаштування основної структури додатку і його функціональності також є приклад наступного файлу (рисунок 4.2).

```dart
void main() {
  runApp(const App());
}

class App extends StatelessWidget {
  const App({super.key});

  @override
  Widget build(BuildContext context) {
    return Sizer(builder: (context, orientation, deviceType) {
      return const MaterialApp(

        home: BottomNavigationBarLib(),
      );  // MaterialApp
    });  // Sizer
  }
}
```

Рисунок 4.2 – Приклад Main.dart

Бібліотеки в Flutter – це набір готових до використання функцій, класів та інших компонентів, що дозволяють розробникам зосередитись на головній меті своєї роботи, а не витрачати час на написання всього з нуля.

Важливість бібліотек в Flutter полягає в тому, що вони дозволяють значно прискорити розробку додатків, зменшити кількість коду, що потрібно написати, а також забезпечують зв'язок з різноманітними API та сервісами. Наявність багатофункціональних бібліотек дозволяє розробникам створювати додатки швидко та з меншими витратами. У Flutter існує величезна кількість різних бібліотек, що дозволяють з легкістю додавати в додатки різноманітні функції, такі як робота з базами даних, мережеві запити, робота з графікою, інтерфейсами користувача та багато іншого. Для кожної задачі можна знайти відповідну бібліотеку, що дозволить зекономити час та зусилля на розробку додатків.

Бібліотеки є невід'ємною складовою Flutter, що робить цей фреймворк привабливим для розробників. Вони дозволяють швидко створювати

функціональні додатки та зосередитись на головному – розробці якісного програмного забезпечення приклад введення бібліотек нижче (рисунок 4.3).

Рисунок 4.3 – Введення бібліотек у Flutter

4.2 Інструменти у внедрення API

У OpenAI API для генерації тексту є кілька важливих параметрів. Давайте розглянемо кожен з них детальніше:

– sufix – це фрагмент тексту, який буде доданий до початкового тексту перед генерацією наступного речення. Він допомагає контролювати тему та зміст тексту, що генерується;

– temperature – це параметр, який контролює креативність та непередбачуваність згенерованого тексту. Ви можете налаштувати його від 0 до 1, де більші значення означають більш випадкову та креативну генерацію тексту;

– top_p – це параметр, який контролює кількість найбільш ймовірних варіантів наступного слова, які API використовує для генерації тексту. Значення top_p також можна налаштувати від 0 до 1, де більші значення означають більш різноманітну та менш повторювану генерацію тексту;

– n – це параметр, який визначає, скільки речень буде згенеровано

API. За замовчуванням n = 1, але ви можете налаштувати його на більші значення, якщо вам потрібно більше тексту;

– max_tokens – це параметр, який визначає максимальну кількість токенів, які можуть бути згенеровані API для кожного речення. За замовчуванням max_tokens = 2048, але ви можете зменшити його, якщо вам потрібен коротший текст, або збільшити, якщо вам потрібно більше деталей.

Наявні команди також представлені далі (рисунок 4.4).

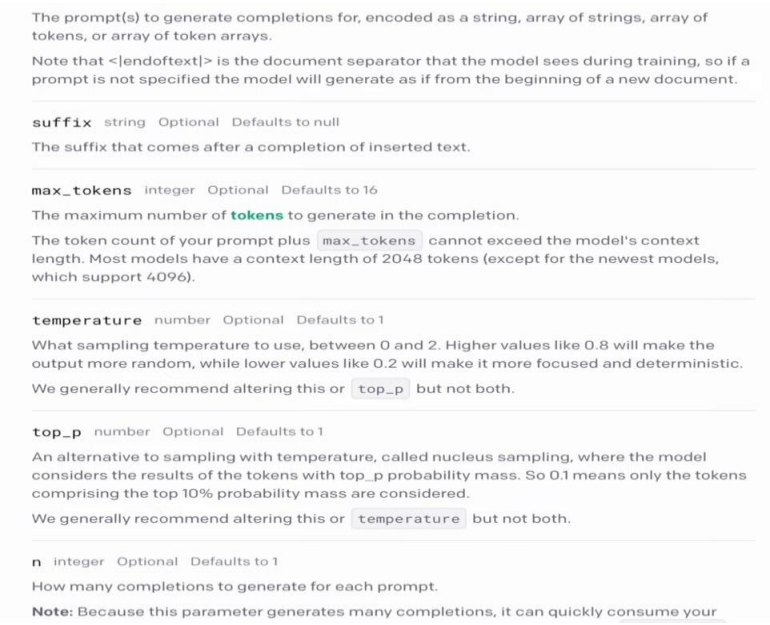

Рисунок 4.4 – Команд для використання API

Для запису параметрів запиту до OpenAI API зазвичай використовують формат JSON. JSON (або JavaScript Object Notation) є легким форматом обміну даними, який використовується для передачі структурованих даних між серверами та клієнтами. В даному випадку, при використанні OpenAI API, JSON дозволяє передавати різноманітні параметри запиту, такі як текст для генерації, кількість згенерованих речень або слів, встановлення температури генерації, налаштування режиму підвищення різноманітності результатів та інші. Структуру побудови можна

побачити далі (рисунок 4.5).

Рисунок 4.5 – Оболонка з запитами до GPT

У відповідь на запит сервер OpenAI API повертає відповідь у форматі JSON, який містить згенерований текст або інші запитані дані. Клієнтська програма може оброблювати ці дані, розпарсюючи JSON та використовуючи інформацію для своїх потреб. Отже, використання JSON для передачі параметрів запиту та отримання відповіді є важливою складовою у взаємодії з OpenAI API та іншими веб-сервісами.

У OpenAI API POST Completion використовується для генерації тексту, що продовжує певний контекст. Користувач може ввести початковий текст або контекст, і задати різні параметри генерації, такі як кількість виведених слів, температуру, top_p, n, max_tokens тощо. Після відправки запиту на сервер OpenAI, API повертає згенерований текст, який може бути використаний для різних цілей, таких як автоматичне завершення тексту, генерація новин, статей, письмових робіт тощо приклад посилання (рисунок 4.6).

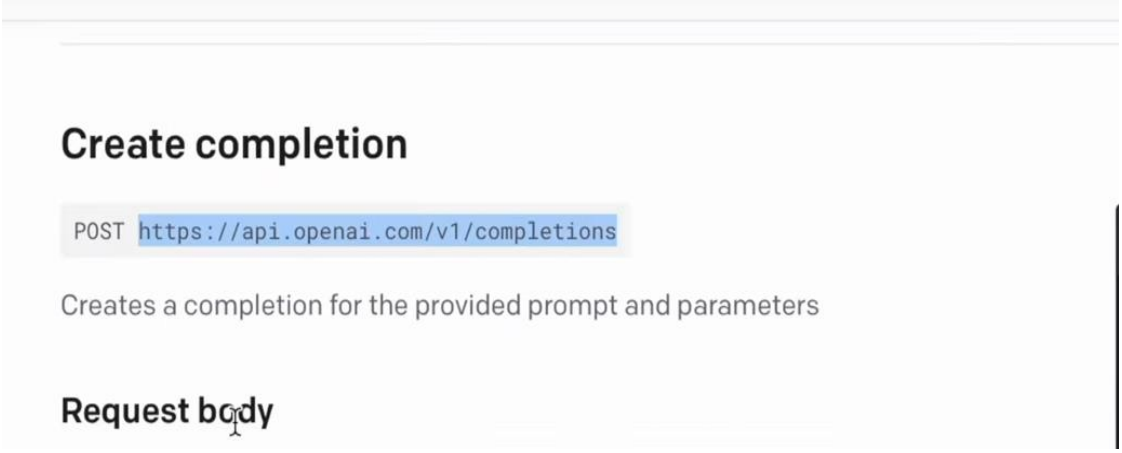

Рисунок 4.6 – Взяття посилання для адресації

Цей метод POST Completion є одним з найбільш потужних інструментів у OpenAI API, оскільки він дозволяє генерувати якісний та змістовний текст на основі заданого контексту та параметрів генерації.

Створення запиту (request) для відправлення повідомлення до OpenAI можна здійснити за допомогою HTTP-запиту методом POST. Щоб скористатися OpenAI API, необхідно створити HTTP-запит до адреси приклад запиту можна побачити далі (рисунок 4.7).

```
RequestBody body = RequestBody.create(jsonBody.toString(),JSON);
Request request = new Request.Builder()
        .url("https://api.openai.com/v1/completions")
        .header()
        .post(body)
        .build();
```

Рисунок 4.7 – Внедрення посилання для адресації

У заголовку (Header) запиту до OpenAI API необхідно вказати ключ авторизації (Authorization), тип контенту (Content-Type) та деякі інші параметри, що залежать від конкретної операції, яку ви намагаєтеся виконати за допомогою API, внедрення ключа, можна побачити нижче (рисунок 4.8).

```
RequestBody body = RequestBody.create(jsonBody.toString(),JSON);
Request request = new Request.Builder()
        .url("https://api.openai.com/v1/completions")
        .header( name: "Authorization", value: "Bearer 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")
        .post(body)
        .build();
```

Рисунок 4.8 – Внедрення ключа розробника

У заголовку реквесту можна також вказати додаткову інформацію, таку як інформацію про клієнта, яка дозволяє серверу розпізнати ваш додаток і забезпечити кращу підтримку. Заголовок реквесту має бути детально відпрацьованим та документованим, щоб сервер міг правильно обробляти запит та повернути необхідні дані.

Після відправлення запиту до сервера OpenAI API і отримання відповіді, отриманий текст може бути виведений на екран або оброблений далі в програмі. Зазвичай, текст виводиться на екран за допомогою інтерфейсу користувача, такого як текстове поле, мітка або діалогове вікно.

У випадку, якщо сервер повертає позитивну відповідь, тобто запит успішно оброблений, можна перевірити наявність отриманого тексту та продовжувати роботу з ним в програмі. Проте, під час взаємодії з сервером OpenAI API можуть виникати помилки, такі як невірний формат запиту, неправильний ключ API або проблеми зі з'єднанням. У такому випадку сервер може повернути відповідь з інформацією про помилку. Цю інформацію можна обробити в програмі та вивести користувачеві на екран, або виконати інші дії, щоб запобігти помилкам у подальшій роботі програми на прикладі знизу можна побачити таку структуру (рисунок 4.9).

```
@Override
public void onResponse(@NonNull Call call, @NonNull Response response) throws IOException {
    if(response.isSuccessful()){
        JSONObject  jsonObject = null;
        try {
            jsonObject = new JSONObject(response.body().toString());
        } catch (JSONException e) {
            e.printStackTrace();
        }
        JSONArray jsonArray = jsonObject.getJSONArray( name: "choices");
        String result = jsonArray.getJSONObject( index: 0).getString( name: "text");
        addResponse(result.trim());
```

Рисунок 4.9 – Обробка помилок та виведення тексту

Для обробки помилок у відповіді від сервера OpenAI API можна використовувати різноманітні механізми, такі як обробники винятків (exception handlers) або структури даних для збереження відповіді та статусу запиту також на прикладі знизу можна це побачити (рисунок 4.10).

```
client.newCall(request).enqueue(new Callback() {
    @Override
    public void onFailure(@NonNull Call call, @NonNull IOException e) {
        addResponse("Failed to load response due to "+e.getMessage());
    }

    @Override
    public void onResponse(@NonNull Call call, @NonNull Response response) throws IOException {
        if(response.isSuccessful()){

        }else{
            addResponse("Failed to load response due to "+response.);
        }
    }
});
```

```
isSuccessful()
body()
toString()
header(String name)
message()
```

Рисунок 4.10 – Оцінка на тестовому наборі

5 РОЗРОБКА ДОДАТКУ

За допомогою Android Studio, Flutter та ChatGPT було розроблено додаток для заповнення питань.

Android Studio – це інтегроване середовище розробки, яке дозволяє розробникам створювати Android-додатки за допомогою Java або Kotlin. Це середовище надає багато корисних інструментів, таких як емулятор Android-пристроїв, редактор макетів та інші.

Flutter – це відкрите середовище для розробки мобільних додатків, яке створено компанією Google. Flutter дозволяє створювати високоякісні додатки для iOS та Android за допомогою одного коду. Використання Flutter дозволяє розробникам ефективно використовувати ресурси та створювати додатки з великою кількістю функцій та можливостей.

GPT – це інструмент штучного інтелекту, який дозволяє створювати текстові моделі з використанням глибокого навчання. Цей інструмент надає можливість розробникам створювати різноманітні додатки, які здатні генерувати текст на основі введеного користувачем запиту.

За допомогою цих компонентів було створено додаток, який може допомогти користувачам заповнити свої питання, використовуючи інтелектуальну систему генерації тексту. Додаток отримує запитання від користувача та використовуючи GPT, генерує відповідь на запитання. Крім того, за допомогою Flutter було розроблено зручний та привабливий інтерфейс додатку, який дозволяє користувачам взаємодіяти з системою легко та ефективно. Таким чином, за допомогою цих компонентів було створено додаток, який допомагає користувачам отримувати відповіді на свої запитання за допомогою інтелектуальної системи генерації тексту.

Розроблений додаток доповнення тексту з інтегрованими моделями наведено на рисунку 5.1.

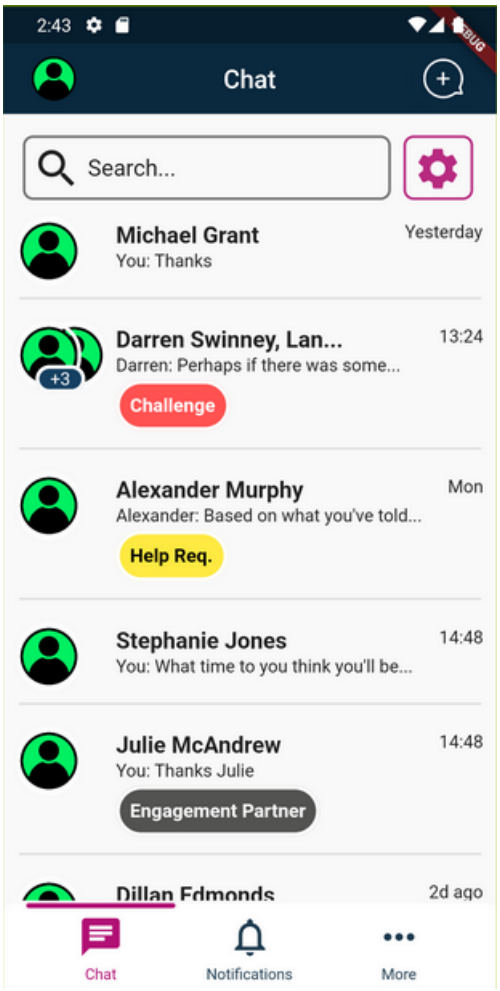

Рисунок 5.1 – Додаток для генерування тексту.

ВИСНОВКИ

У цій кваліфікаційній роботі була розглянута тема «Інформаційна система первинної підготовки запитів до GPT». Було досліджено сутність інформаційної системи та її роль у вирішенні проблеми підготовки запитів до GPT.

Було проведено аналіз предметної області та досліджено актуальність GPT для вирішення проблем, таких як генерація текстів, автопереклад, підготовка відповідей та інші. Також були розглянуті різні додатки для Android, які використовують GPT для генерації тексту.

Було детально описано два з таких додатків – GPT Playground та Text Blaze, а також Copy.ai. За допомогою GPT Playground користувач може ввести текст та отримати його продовження, змінити налаштування генерації тексту, а також виконувати інші функції. Text Blaze є додатком, що дозволяє користувачам зберігати та використовувати текстові шаблони, що генеруються за допомогою GPT.

Також було проведено аналіз того, як інформаційна система первинної підготовки запитів може бути корисною для підприємств та організацій, що працюють з великим обсягом текстової інформації, такої як документація, звіти, повідомлення та інше.

У загальному, було показано, що додаток інформаційної системи первинної підготовки запитів є важливою складовою для використання GPT та інших штучних інтелектуальних систем для роботи з текстовою інформацією. Більшість додатків, що використовують GPT, дозволяють користувачам більш ефективно взаємодіяти з текстовою інформацією.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Mobile App Development with Xamarin. URL: https://dotnet.microsoft.com/apps/xamarin (дата звернення 13.05.2023).

2. Flutter. URL: https://flutter.dev/ (дата звернення 13.05.2023).

3. Android Developers – Getting Started with Android App Development. URL: https://developer.android.com/guide (дата звернення 13.05.2023).

4. iOS Developer Library. URL: https://developer.apple.com/documentation/ (дата звернення 13.05.2023).

5. TensorFlow – An end-to-end open source machine learning platform. URL: https://www.tensorflow.org/ (дата звернення 13.05.2023).

6. PyTorch – An open source machine learning framework. URL: https://pytorch.org/ (дата звернення 13.05.2023).

7. OpenAI. URL: https://openai.com/ (дата звернення 13.05.2023).

8. Microsoft Azure Cognitive Services. URL: https://azure.microsoft.com/en-us/services/cognitive-services/ (дата звернення 13.05.2023).

9. IBM Watson Services. URL: https://www.ibm.com/cloud/watson-services (дата звернення 13.05.2023).

10. Google Cloud AI. URL: https://cloud.google.com/ai (дата звернення 13.05.2023).

11. Amazon Web Services – Artificial Intelligence. URL: https://aws.amazon.com/ai/ (дата звернення 13.05.2023).

12. Dialogflow – Conversational AI Platform. URL: https://cloud.google.com/dialogflow/ (дата звернення 13.05.2023).

13. Wit.ai – Natural Language for Developers. URL: https://wit.ai/ (дата звернення 13.05.2023).

14. Rasa – Open source conversational AI. URL: https://rasa.com/ (дата звернення 13.05.2023).

15. Botpress – The open-source conversational platform. URL: https://botpress.com/ (дата звернення 13.05.2023).

16. Microsoft Bot Framework. URL: https://dev.botframework.com/ (дата звернення 13.05.2023).

17. Apple Core ML. URL: https://developer.apple.com/documentation/coreml (дата звернення 13.05.2023).

18. Google ML Kit. URL: https://developers.google.com/ml-kit (дата звернення 13.05.2023).

19. Scikit-learn – Machine Learning in Python. URL: https://scikit-learn.org/stable/ (дата звернення 13.05.2023).

20. OpenAI API 19. OpenAI API Documentation. URL: https://openai.com/docs/api-reference/ (дата звернення 13.05.2023).

ДОДАТОК А

Відомість кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	59 с.
	Інші документи	
2.	Презентаційні матеріали	__ плакатів

.	.	Прізвище та ініціали.	Підп.	Дата		
Розробив		Масловський М.Ю.			Шифр групи	Код напр./спец.
Перевірив		Магдаліна І.В.			СШІм-21-2	122
Н.контр.		Малєєва І.А.			ХНУРЕ кафедра ШІ	
Затв.		Філатов В.О.				

Інформаційна система первинної підготовки запитів до chatGPT