НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ
кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

“До захисту допущено”
Завідувач кафедри ЦТЕ
_____ Наталія АУШЕВА

“___”_____2024 р.

Дипломна робота

на здобуття ступеня бакалавра
за освітньо-професійною програмою
“Цифрові технології в енергетиці”
зі спеціальності *122 “Комп’ютерні науки”*

на тему: **Веб система аналізу тональності тексту**

Виконавла:
студентка IV курсу, групи ТР-02

Носкова Дар’я Максимівна _____
(прізвище, ім’я, по батькові) (підпис)

Керівник: *асистент каф. цифрових технологій в енергетиці*
асистент, Костянтин ЗДОР _____
(посада, вчене звання ,науковий ступінь, ім’я, ПРІЗВИЩЕ) (підпис)

Рецензент: _____
_____
(посада, вчене звання, науковий ступінь, ім’я, ПРІЗВИЩЕ) (підпис)

Н.контроль: *асистент, Андрій ПАСІЧНЮК* _____
(посада, ім’я, ПРІЗВИЩЕ) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____
(підпис)

Київ – 2024

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ

Кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

Рівень вищої освіти – перший (бакалаврський)
спеціальність 122 “Комп’ютерні науки”
Освітньо-професійна програма “Цифрові технології в енергетиці”

ЗАТВЕРДЖУЮ
Завідувач кафедри ЦТЕ
_____ Наталія АУШЕВА_

«__»_____2024 р.

ЗАВДАННЯ
на дипломну роботу студенту

НОСКОВІЙ Дар’ї Максимівні
(прізвище, ім’я, по батькові)

1. Тема роботи **Веб-система аналізу тональності тексту**

керівник роботи ***Здор Костянтин Андрійович, асистент***
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «___»_____ 202__ р. №_____

2. Термін подання роботи студентом **07.06.2024 року**

3. Вихідні дані до роботи мова програмування Python, фреймворк Streamlit, СКБД Redis, середовище розробки PyCharm, інструмент для тестування Docker

4. Перелік питань, які потрібно розробити
1) провести аналіз серед наявних аналогів
2) веб-систем для надання та пошуку наставницьких послуг в сфері ІТ
3) аргументувати вибрані інструменти, технології та алгоритми для реалізації програмного забезпечення

4) побудувати архітектури програмного забезпечення, баз даних та сховища файлів

5) створити прикладний програмний веб-інтерфейс

6) представити процес застосування веб-інтерфейсу

5. Орієнтований перелік ілюстративного матеріалу діаграми, що демонструють роботу алгоритмів веб-інтерфейсу, архітектуру системи, бази даних та сховища файлів, взаємодію користувачів із системою, класи програмного забезпечення; приклади роботи з програмним продуктом.

6. Дата видачі завдання «15» вересня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1.	Затвердження теми роботи		Виконано
2.	Вивчення та аналіз задачі	17-20.04.24	Виконано
3.	Розробка архітектури та загальної структури системи	21-25.04.24	Виконано
4.	Розробка частин окремих підсистем	25.04-02.05.24	Виконано
5.	Програмна реалізація системи	03-07.05.24	Виконано
6.	Оформлення пояснювальної записки	08-12.05.24	Виконано
7.	Захист програмного продукту	13-17.05.24	Виконано
8.	Передзахист	03-06.06.24	Виконано
9.	Подання готової роботи на кафедру	07.06.2024	Виконано
10.	Захист	17-21.06.2024	Виконано

Студент _____ **Дар'я НОСКОВА**
(підпис) (ім'я, ПРІЗВИЩЕ)

Керівник роботи _____ **Костянтин ЗДОР**
(підпис) (ім'я, ПРІЗВИЩЕ)

АНОТАЦІЯ

Дипломна робота виконана на 58 сторінках, містить 13 ілюстрацій, 1 додаток, 18 джерел в переліку посилань.

Метою проекту є створення веб-системи, яка аналізує настрої тексту за допомогою моделі RoBERTa для виявлення емоційних відтінків у тексті та моделі GPT-3.5 для переписування тексту. Робота реалізує обробку природної мови разом із методами машинного навчання. Основні інструменти, що використовуються в розробці, включають мову програмування Python, фреймворк Streamlit для створення програм даних, базу даних Redis для кешування даних, PyCharm IDE як середовище розробки та інструмент тестування Docker для автоматизації тестів у контейнерах.

Ключові слова: АНАЛІЗ ТОНАЛЬНОСТІ ТЕКСТУ, ВЕБ-СИСТЕМА, ROBERTA, GPT-3.5, STREAMLIT, REDIS.

ABSTRACT

The thesis consists of 58 pages, includes 13 illustrations, 1 appendice, and 18 sources in the list of references.

The project's goal is to create a web system that analyzes text sentiment using the RoBERTa model to detect emotional nuances in the text and the GPT-3.5 model for rewriting text. The work implements natural language processing along with machine learning methods. The main tools used in the development include the Python programming language, the Streamlit framework for creating data applications, the Redis database for data caching, PyCharm IDE as the development environment, and Docker as a testing tool for automating tests in containers.

Keywords: SENTIMENT ANALYSIS, WEB SYSTEM, ROBERTA, GPT-3.5, STREAMLIT, REDIS

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API — інтерфейс програмування застосунків (англ. Application Programming Interface).

БД — база даних.

HTTP — протокол передачі гіпертексту (англ. HyperText Transfer Protocol).

HTTPS — захищений протокол передачі гіпертексту (англ. HyperText Transfer Protocol Secure).

Back-end — серверна частина застосунку.

IDE — інтегроване середовище розробки (англ. Integrated Development Environment).

RoBERTa — модифікація моделі BERT для покращеного розпізнавання тексту (англ. Robustly Optimized BERT Pretraining Approach).

Redis — система кешування даних у пам'яті (англ. Remote Dictionary Server).

GPT-3.5 — генеративний переднавчений трансформер, версія 3.5 (англ. Generative Pre-trained Transformer 3.5).

ВСТУП

Аналіз настрою тексту став однією з ключових галузей у світі сучасних систем обробки природної мови (NLP). Заради щоденного нагромадження інформації, важливим стає використання автоматизованих систем для ефективного розпізнавання емоцій у текстах. Це питання набирає актуальності серед компаній, оскільки розуміння настроїв клієнтів може значно впливати на репутацію компанії або бренду, що сприяє розв'язанню конфліктних ситуацій та уникненню непорозумінь між замовниками послуг. Наприклад, оперативна відповідь компанії на критичне сприйняття користувачами продуктів та послуг дозволяє вирішувати проблеми за пріоритетністю, коригуючи продукти та покращуючи задоволення клієнтів. Усе це можливості завдяки цінній іінформації, яку подає аналіз відгуків. Крім цього, робота з великим обсягом даних не є повноцінною без аналізу настроїв тексту, що дає новий напрямок для наукових дослiджень і комерцiйного застосування.

Незважаючu на значниu успіх у цій галузі, багатьма сучасними рішеннями характеризують двi основні проблеми: недостатня точність і флексибільність. Точність означає здатність адаптувати аналіз до конкретних потреб користувача. На жаль, більшість доступних систем або загальні, або потребують складних налаштувань для специфичних сценаріїв використання. Це стає гальмом на шляху до широкого поширення таких систем у різних галузях, від маркетингу до медицини. Таким чином, разробка веб-системи з модернізованими моделями машиного навчання, такими як RoBERTa і GPT-3.5, здобуває особливі значення, оскільки забезпечує посилену якість і зручність аналізу і переписуванню тексу. Використання таких моделей забезпечує високу точність виявлення та генерації нових сентиментів, а також сприяє простоті адаптації до різноманітних завдань.

Основна ідея дипломної роботи полягає у створенні веб-системи, яка об'єднує сучасні технології із моделями машинного навчання для аналізу та переписування тональності тексту. Система побудована з урахуванням сучасних стандартів щодо зручності користувацького інтерфейсу та ефективності обробки даних. Вона може легко пристосовуватися до різних потреб користувачів завдяки

свою гнучкості. Отримані результати мають значний потенціал для застосування в різних галузях, будь то маркетинг, обслуговуючий сервис чи дослідження соцмереж.

При розробці використовувлась модель RoBERTa через її високу точність у визначенні тональності тексту. Окрім того, під час розробки також застосовувалась модель GPT-3.5, оскільки вона ефективно генерує текст на основІ заданих параметрів із збереженням цілісності оригиналу. Така комбінація покращує сприйняття емоційних нюансів у текстах і дозволяє адаптувати їх за індивідуальними потребами.

Аналіз і переписування тексту з урахуванням тональності є надзвичайно актуальними, оскільки дозволяють поліпшити комунікацію мІж клієнтами та ефективно взаємодіяти з різними типами аудиторії. Правильний сентимент допомагаж точно передати повідомлення, сприяючи кращому порозумінню та зміцненню іміджу фірми чи організації. Для маркетингових компаній вибір відповідної тональності підвище взаємодію з цільовою аудиторією, роблячи текст більш приваблимим і релевантним. У бізнес комунікаціях правильний тон допомагає розв'язувати напружені конфлікти та підтримувати конструктивний діалог.

Аналіз та вдосконалення виразності текстів стають ключовими засобами для досягнення ефективного спілкування та успішної реалізації комунікаційної стратегії.

1 ПОСТАНОВКА ЗАДАЧІ

Розробити веб-додаток, який використовує модель для аналізу емоційного забарвлення текстів. Додаток повинен дозволяти користувачам реєструватися, входити в систему та використовувати основний функціонал для аналізу текстів. Реалізувати базу даних для зберігання інформації про користувачів та їх активності.

Основний інтерфейс веб-додатку розроблено чітким та зрозумілим, де користувачі можуть легко навігувати між-функціями реєстрації, входу та аналізу тексту.

Бічна панель для швидкого доступу до різних секцій додатку, включаючи особистий кабінет, історію запитів, документацію та інструкції користувача.

Функціонал реєстрації та входу:

- форма реєстрації з обов’язковими полями (ім’я користувача, електронна адреса, пароль), включаючи перевірку даних на валідність;

- форма входу для існуючих користувачів з аутентифікацією за допомогою імені користувача та пароля.

Кабінет користувача:

- Особиста сторінка з інформацією про користувача, історією його запитів та результатами аналізу.

- Можливість переглядати історію аналізів.

- Аналіз тональності:

- Форма для вводу тексту для аналізу.

- Використання моделі для аналізу тексту на позитивний, нейтральний, або негативний сентименти.

- Показ відкласифікованого результату у зрозумілій формі з можливістю отримання тексту з іншим сентиментом, відмінним від відкласифікованого.

Інтеграція функціоналу для пропозицій щодо переписування тексту на основі отриманих результатів, використовуючи моделі для генерації альтернативних формулювань.

Використання бази даних для зберігання інформації про користувачів, а також для зберігання результатів аналізу, що забезпечує швидкий доступ та високу продуктивність системи.

Веб-додаток для аналізу емоційного забарвлення текстів буде надавати користувачам можливість реєструватися, входити в систему, а також користуватися основним функціоналом для аналізу текстів. Основний інтерфейс веб-додатку буде розроблений чітким та зрозумілим, що дозволить користувачам легко навігувати між різними функціями, такими як реєстрація, вхід та аналіз тексту. Додаток буде оснащений бічною панеллю для швидкого доступу до різних секцій, включаючи особистий кабінет, історію запитів та інструкції користувача.

Веб-додаток буде масштабованим та продуктивним, завдяки використанню кешування для тимчасового зберігання результатів частих запитів. Документація додатку надасть користувачам детальні пояснення функціоналу, опис технологічного стеку та архітектури. Інструкції користувача включатимуть покрокові інструкції щодо реєстрації, входу та використання основного функціоналу, а також часті запитання для вирішення поширених проблем, що зробить користування додатком максимально комфортним і простим.

2 АНАЛІЗ ТОНАЛЬНОСТІ ТЕКСТУ ЗА ДОПОМОГОЮ ШТУЧНОГО ІНТЕЛЕКТУ

Сьогодні штучний інтелект (ШІ) став однією з найбільш інноваційних та динамічних технологій, яка впливає на різні сфери нашого життя, включаючи медицину, фінанси, розваги та освіту. Прогрес у галузі машинного навчання, глибокого навчання та обробки природної мови (NLP) сприяє розвитку ШІ. Ця категорія технологій дозволяє створювати системи, яким попередньо потребувалася людська інтелектуальна присутність для аналізу великих обсягів даних, розпізнаванню зображень та прийманню рішень. Також це може бути використано для взаємодії з людьми на природній мові.

Однією з ключових сфер, де застосовується ШІ, є аналіз настроїв у текстах. Ця технологія дозволяє автоматично визначати емоційний відтінок тексту - чи він позитивний, негативний або нейтральний. Аналіз настроїв у текстах має широке застосування у різних галузях, включаючи маркетинг. У цьому контексті компанії можуть використовувати дану технологію для оцінки сприйняття своїх продуктів і послуг клієнтами на основі їх відгуків та коментарів у соцмережах та інших джерелах текстових даних. У політичному житті аналітика тональності також допомагає слідкувати за громадською думкою та реакціями на політичні події чи заяви. В області обслуговування клієнтів ця технологія може автоматично класифікувати звернення клієнтів за ступенем їх задоволеності, що дасть можливість швидше реагувати на негативний фідбек и покращувати якість обслуговування.

ШІ - однин з найдинамичниших і найперспективніших напрямків сучасної науки і техніки, який пророкує значні зміни у багатьох аспектах життя. Один із напрямків, де ШІ досягав значних результатів і досягнень - це аналітіка тональності тексту. Цей напрямок досліджує як комп’ютери можуть визначати емоційне забарвлення тексту, що має численні застосування в маркетингу, соціології, політиці та багатьох інших сферах.

2.1 Штучний інтелект

Штучний інтелект - це комплекс наукових і технічних дисциплін, що займається створенням інтелектуальних систем, здатних виконувати завдання, які зазвичай потребують людського інтелекту. До таких завдань належать розпізнавання мовлення, ухвалення рішень, переклад мови, гра у шахи та інші види діяльності. ШІ можна поділити на два основних типи: вузький (слабкий) ШІ і вагальний (сильний) ШІ [18].

Вузький (слабкий) ШІ являє собою системи, які виконують конкретні завдання. Приклади включають розпізнавання мови або облич, а також рекомендаційні системи.

Загальний (сильний) ШІ включає гіпотетичні системи, що мають загальні інтелектуальні здібності, подібні до людських. Вони здатні навчатися та застосовувати знання у різних контекстах без спеціальної підготовки.

До основних методів ШІ належать: машинне навчання (ML), глибинне навчання (DL), обробка природної мови (NLP), експертні системи.

Машинне навчання (ML) представляє собою підгалузь ШІ, що дозволяє системам навчатися на основі даних. Машинне навчання включає різноманітні алгоритми, такі як регресія, класифікація та кластеризація. Системи машинного навчання можуть навчатися на великих обсягах даних і виявляти закономірності, які використовуються для ухвалення рішень.

Глибинне навчання (DL) є підвидом машинного навчання, що використовує нейронні мережі з багатьма шарами для аналізу даних. Глибинні нейронні мережі (DNN) здатні навчатися з необроблених даних, витягуючи з них складні особливості та взаємозв'язки. Приклади DL включають згорткові нейронні мережі (CNN) для обробки зображень і рекурентні нейронні мережі (RNN) для обробки послідовностей даних.

Обробка природної мови (NLP) - це галузь ШІ, що займається взаємодією між комп'ютерами та людською мовою. Вона включає розуміння, інтерпретацію та генерування тексту на природних мовах. NLP використовує як традиційні методи машинного навчання, так і новітні підходи глибинного навчання [11].

Експертні системи використовують знання та правила для вирішення складних задач у певних галузях знань. Вони імітують процес ухвалення рішень експертами в конкретних областях.

2.2 Аналіз тональності тексту

Вивчення емоційного відтінку тексту (Sentiment Analysis) є одним з ключових аспектів обробки природної мови (NLP). Його метою є визначення емоційного спектру тексту - чи є він позитивним, негативним чи нейтральним. Крім цього, аналіз тональності може розпізнавати багато складних емоційних станів, таких як радощі, сум, гнів та інше [5].

Існують різні підходи до аналізу тональності тексту, серед яких виділяються: лексиконні методи, машинне навчання та глибинне навчання.

Лексикографічні підходи використовують словники, що містять слова з відомим емоційним забарвленням, наприклад SentiWordNet та AFINN. Кожне слово оцінюється на основі його присутності у словнику, а загальна емоційна окраска тексту визначається як сума оцінок кожного слова. Це дозволяє отримати зрозумілі результати, але такий метод обмежений у роботі з складними мовними конструкціями та контекстом тексту.

Методи машинного навчання базуються на алгоритмах, які вивчаються за допомогою позначених даних (тексти з відомою емоційною окрасою) і можуть передбачати емоційну окрасу нових текстів. Серед популярних методів - наївний баєсовий класифікатор, метод опорних векторів (SVM), логістична регресія [15]. Машинне навчання може ураховувати контекст і пристосовуватися до нових даних, але потребує значної кількості позначених даних для тренування моделей [6].

Глибоке навчання використовує рекурентні нейронні мережі, такі як LSTM (модель довгострокової пам'яті) та GRU (гейтовані рекурентні блоки), що добре підходять для обробки послідовних даних, таких як текст. Також варто зазначити про трансформери, такі як BERT і GPT, які продемонстрували високу ефективність у завданнях аналізу тексту завдяки їх здатності розуміти контекст слів у широкому

спектрі [16]. Глибоке навчання характеризується високою точністю й можливістю розпізнавати складні мовні структури, але для успішного навчання потребує значних обчислювальних ресурсів та обсяru даних.

Аналіз настрою тексту має широке застосування у різних галузях. У сфері бізнесу та маркетингу важливо вивчати відгуки клієнтів, аналізувати соцмережі для оцінки сприйняття бренду, керувати репутацією, а також досліджувати враження про продукти та послуги. У політичних справах проводиться аналіз громадської думки стосовно політичних подій або кандидатов, мониторинг ЗМІ для виявлення суспильного настрою. В галузях соціальних наук досліджують емоційні реакції на різноманітні соціальні явища та проводять аналітику соцмереж для виявлення ставлення у суспльствіі. Також кольоровий спектр тексту може мати значення у медичних питаннях, наприклад для визначення емоцыйных станiv хворих з метою надання психологичної пІдтримки чи моні́торингу настроях серед населеня пІд час епі́демі́й чи кризових ситиаций.

Незважаючи на значний прогрес у сфері аналізу емоційного забарвлення тексту, існує кілька проблем, які потребують подальших досліджень. Вживання ідіом, сарказму та контекстних виразів може бути викликом для автоматичних систем. Наприклад, фраза “Це чудово!” може мати як позитивне, так і негативне значення в залежності від ситуації. Розробка моделей, які однаково ефективно працюють з різними мовами, є складною задачею. Багато сучасних моделей добре функціонують для англомовного контенту, але можуть стикається з труднощами при роботі з іншими мовами. Ефективність моделей може значно вар'юватися у розмаїтих галузях. Наприклад, оцінка тональності фінансових текстів може значно відрізнятися від оцінки тональності у коментарях до фільмiv. Використання аналізу емоціі може порушити конфиденцііальность та приватність і потребую полегшеного додержання етичних норм і законодавства.

Аналіз настрою тексту є чудовим прикладом практичного застосування інноваційних технологій, зокрема методів машинного та глибинного навчання. Використання штучного інтелекту для аналізу настрою тексту дозволяє автоматизувати процеси, які раніше вимагали значних людських зусиль, та забезпечує високу точність і ефективність аналізу.

Використання методів машинного навчання дозволяє створювати моделі, яким навчаються на великих обсягах текстових даних і можуть пристосовуватися до нових даних, що робить їх універсальними та гнучкими у використанні. Глибоке навчання, зокрема застосування трансформерних моделей, таких як RoBERTa, BERT та GPT, дозволяє враховувати контекст слів у тексті, що значно покращує точність аналізу настрою.

Інтеграція методів штучного інтелекту в аналіз настрою тексту розширює можливості для бізнесу, науки та суспільства. Вона не лише автоматизує аналіз текстових даних, а й робить це на більш високому рівні, забезпечуючи точніші та глибші інсайти. Це сприяє прийняттю обґрунтованих рішень, підвищенню ефективності комунікацій та поліпшенню якості послуг и продуктів.

Отже, штучний інтелект та аналіз тональності тексту взаємопов'язані галузі, де просування в одній з них сприяє розвитку іншої. Взаємодія між ними створює нові можливості для аналізу та розуміння текстових даних, що має велике значення для сучасного суспільства. Цими технологіями можна не лише ефективно оцінювати громадську думку, але й передбачати поведінку користувачів, виявляти потенційні проблеми у взаємодії з кліентами та навіть прогнозувати тенденції ринку. Удосконалення штучного інтелекту та методів аналізу емоційного забарвлення текстів сприяють створенню більш точних і надійних систем, які можуть обробляти та розуміти складну мовну структуру. Це в свою чергу покращує ефективність та точність у прийнятті рішень у різних сферах.

2.3 Модель RoBERTa

Удосконаленням моделі BERT (Bidirectional Encoder Representations from Transformers) є RoBERTa (Phonely optimized BERT approach), розроблена Facebook AI. Вони представили цю модель у 2019 році в статті «RoBERTa: надійно оптимізований підхід до попереднього навчання BERT». Метою RoBERTa є покращення результатів BERT шляхом оптимізації попереднього навчання, що веде до отримання більшої кількості даних [17].

Архітектура RoBERTa залишається схожою на архітектуру BERT і базується на енкодері трансформера з багатошаровим механізмом самоваги (self-attention). Проте, RoBERTa вносить значні зміни у методологію переднього навчання. По-перше, вона використовує значно більший обсяг даних. Якщо BERT тренувався на 16GB тексту з англомовної Вікіпедії та BooksCorpus, то RoBERTa користується 160GB текстом, що охоплює англомовну Вікіпедію, BooksCorpus, OpenWebText, Stories та інші джерела. Це дає можливість моделі краще захоплювати контекст і значення слів.

Процес навчання RoBERTa має інший шлях. Він використовує більші розміри партій і заглиблюється в більше ітерацій. З партіями, які напружують свої м’язи до 8000 прикладів, і кількість ітерацій, що стрімко зростають за 500 000 кроків до попереднього навчання, цей підхід малює яскраву картину для моделі. Він отримує віртуальний погляд на більшу кількість прикладів — вправа, спрямована на отримання кращого узагальнення інформації. Важлива зміна полягає в тому, що RoBERTa використовує динамічне маскування слів на відміну від статичного підходу BERT, де слова маскуються однаково в різних уривках. Вдаючись до цього особливого методу, який забезпечує різні маски для слів при кожній зустрічі з набором даних, RoBERTa чудово налаштовується на те, щоб зрозуміти різноманітні контексти навколо різних слів — дія, яка, отже, підвищує ефективність навчання на північ.

Ще однією важливою зміною є усунення завдання NSP (Next Sentence Prediction). BERT застосував завдання прогнозування наступного речення, яке допомогло моделі зрозуміти зв’язки між реченнями. Однак дослідження показали, що це конкретне завдання не значно впливає на продуктивність моделі; тому RoBERTa відкидає його та акцентує увагу на завданні маскування слів (модель маскування мови). Цей підхід вважається кращим для покращення можливостей навчання, оскільки більше деталей можна легше зрозуміти з меншим когнітивним навантаженням.

Оптимальна ефективність також залежить від правильного налаштування гіперпараметрів. RoBERTa проводить ретельні експерименти, що передбачають

налаштування таких параметрів, як швидкість навчання або розмір партії тощо, завдяки чому модель досягає оптимальних результатів у всіх різних завданнях.

RoBERTa продемонструвала значно кращу продуктивність у порівнянні з базовою моделлю BERT на багатьох важливих наборах тестувань, що використовуються для оцінки обробки природної мови (NLP), таких як GLUE (Загальне оцінювання розуміння мови), SQuAD (Стенфордський набір даних відповідей на запитання) та RACE (Розуміння прочитаного з іспитів) серед інших. Високий результат RoBERTa за цими тестами свідчить про його покращену здатність розуміти та декодувати природну мову, що робить його цікавим інструментом у численних завдань НЛП, таких як класифікації тексту, розпізнавання сутностей та відповідь на запитання, серед купи інших [13].

Поміж інших переваг моделі RoBERTa варто відзначити її гнучкість та розширюваність, додатково до високої продуктивності. Ця система підтримує різноманітні архітектури та налаштування, що дозволяє адаптувати її для конкретних завдань. Наприклад, модель RoBERTa може бути налаштована для роботи з різними мовами та завданнями, ставши універсальним інструментом для розв'язку широкого спектру текстових завдань.

Крім цього, RoBERTa має широке застосування у багатьох галузях промисловості та наукових сферах. Вона лежить в основі багатьох програм у сфері обробки природної мови - чи то автоматизованих систем перепланування чи систем рекомендацій, аналіз настроїв чи чат-ботів. Її ефективна використовувальна потужність разом з імпонуючою точністю забезпечують їй популярність серед організацій, які оперують з великими обсягами текстової інформації.

RoBERTa перебудувала ландшафт моделей обробки природної мови завдяки іноваціям у методиках навчаня та оптимізації гіпермараметрів. Ця модель подала значний прогрес у продуктивності, видкривши нові направлення досліджень і розробок у сегменті обробки природної мови. Досягнення PoBERTa не означають кінцеву точку, а вони стали початком нової ери, яка призводить до створення ще потужніших моделей, побудованих на концепціяї трансформаційних архітектур.

2.4 Модель GPT-3.5

Поява моделі OpenAI GPT-3.5 стала важливою подією в розвитку генеративних трансформаторів, побудованих на основі архітектури трансформаторів. Завдяки механізму самоконтролю, ця модель може покращити обробку послідовностей даних, встановлюючи зв'язки між будь-якими двома точками у послідовностях незалежно від їхнього розташування.

Модель GPT-3.5 була представлена після попередньої версії GPT, зокрема GPT-3, яка була найбільшою та потужною моделлю обробки природної мови на момент свого запуску у 2020 році. GPT-3.5 продовжує цей шлях, пропонуючи поліпшення продуктивності завдяки ряду суттєвих змін.

По-перше, GPT-3.5 має значно більшу кількість параметрів порівняно з попередніми моделями. Навіть якщо точний розмір моделі не завжди виявляється, вона може містити сотні мільярдів параметрів, що дозволяє обробляти та аналізувати величезну кількостю даних. Це дозволяє моделі генерувати якісний текст, розуміючи контекстні нюанси, що забезпечують високий рівень деталей та точності у результатах.

Важливою особливістю GPT-3.5 є те, що вона навчалася на широкому спектрі текстових джерел. Серед них - книги, статті, інтернет-сторінки, форуми тощо. Це дає моделі доступ до різноманітних областей знань і дає можливість генерації тексту на будь-яку тему.

Процес навчання складається з двох етапів: попереднього навчання для основного ознайомлення з великою кількістю неструктурованих даних та точного налаштування для специфічного налаштуваиня на конкретні дані для покращення його продуктивності у конкретних завданнях.

GPT-3.5 працює на основі методу автоматичного мовлення, де слова генеруються послідовно одне за одним з урахуванням вже створених. Це дозволяє моделі створювати логічний та змістовний текст у контексті, оскільки кожне попереднє слово впливає на наступне. Сама модель має складну структуру з багатьма шарами; кожен шар спрямований на поліпшення когнітивного розуміння

та синтезу для обробки текстових даних, що призводить до якісного результату через цей поетапний процес поглиблення та розширення.

Одним з ключових особливостей GPT-3.5 є можливість навчання на малих обсягах даних або навіть без нього, що дозволяє моделі виконувати завдання навіть у рази, якщо вони не були частинами початкового тренувального процесу або отримали лише обмежений обсяг навчання. Модель може розв'язувати задачі лише за описом або прикладами, поданими безпосередньо у запиті, не потребуючи інформації у тренувальних даних, що робить її легко адаптованою до нових завдань та ситуацій.

Модель GPT-3.5 проявляє високу ефективність у створенні контенту, який поєднує в собі креативність і інформативність. Це робить її надзвичайно корисною у різних галузях, таких як автоматизоване написання статей, формування публікацій у соцмережах, програмування та обслуговування клієнтської бази, а також задоволення освітніх потреб. Текст, який генерується цим штучним інтелектом, настільки якісний, що важко розрозуміти його від текстів людей. Це дає можливості для автоматизації процесів та покращення продуктивності, показуючи значущий вплив на сучасне програмне забезпечення.

Використання GPT-3.5 неминуче супроводжується різними викликами та обмеженнями. Однією з головних проблем є боротьба з упередженнями та етичними питаннями. Модель навчається на даних, які містять певні упередження, що може призвести до їх виявлення у створеному контенті. Це може проявлятися у випадковому відображенні соціальних стереотипів, гендерних ролей або расових упереджень. Для вирішення цих проблем необхідно застосовувати комплексний підхід, який охоплює ретельний вибір навчального матеріалу, настройку алгоритмів та впровадження механізмів контролю та стримувань для забезпечення етичного застосування цієї технології.

Ще однією серйозною перешкодою є високі вимоги до обчислювальних ресурсів, необхідних для навчання та запуску моделі GPT-3.5. Для реалізації таких моделей потрібні потужні обчислювальні можливості, яким можуть бути важко доступними фінансово та технологічно для багатьох організацій. Це призводить до додаткових проблем, пов'язаних із доступністю технологій.

Основним принципом моделі GPT є її здатність генерувати текст на основі вхідних даних. За допомогою архітектури Transformer, ця модель може аналізувати контекст тексту та створювати відповідні реакції або продовження тексту. GPT-3.5 є значним досягненням у розвитку генеративних моделей обробки природної мови, показуючи надзвичайну продуктивність у різноманітних завданнях і сценаріях застосування, які позначають значний прогрес у сфері штучного інтелекту.

Проте ефективне та етичне використання таких потужних моделей вимагає обережного погляду і усвідомленості можливих викликів та обмежень, пов'язаних з їх застосуванням. GPT-3.5 представляє собою покращену версію попередників, яка має полегшений точний результат і здатна генерувати більш складений та важливий текст. Ця модель може братися до роботи для рядка завдань, таких як створення текстів, відповіді на запитання, переклад тексту, аналіз мовлення та безліч інших напрямків діяльності, забезпечуючи як високий результат, так і широкий спектр застосунків.

3 ЗАСОБИ РОЗРОБКИ

Розробка програмних систем вимагає знань та використання різних технологій, які працюють у взаємодії для створення функціональних рішень. У цьому розділі описані ключові технології, які були використані для розробки веб системи аналізу тональності тексту.

3.1 Мова програмування Python

Python - це відома мова програмування з відкритим кодом, яка широко використовується для розробки як окремих програм, так і скриптів у різних галузях. Її безкоштовність, портативність, потужність та простота використання роблять її привабливою для програмістів усього світу. Python прагне до покращення продуктивності розробника та якості програмного забезпечення, що стало стратегічною перевагою як для великих, так і для невеликих проектів. [7].

Для багатьох людей важливість мови Python у зручності читання, послідовності та загальному якості програмного забезпечення в цілому робить її відмінною від інших інструментів у світі сценарного програмування. Код на Python розроблено так, щоб його можна було легко читати, а отже, його можна було ефективно використовувати та підтримувати набагато краще, ніж у традиційних мовах сценарного програмування. Однаковий стиль коду Python допомагає в його сприйнятті.

Безумовно, Python підтримує більше застосування механізмів перевикористання програмного забезпечення, таких як об'єктно-орієнтоване програмування (ООП), що також розширює його гнучкість. Продуктивність розробника значно підбадьорюється при застосуванні Python порівняно з компільованими або статично типізованими мовами, такими як C, C++ і Java. Зазвичай код на Python складає лише третину або навіть одну п'яту обсягу еквівалентного коду C++ або Java. Це означає, що чим менше написаних рядків коду, тим менше налагоджень та послуг по упровадженню [9]. Програми на Python

також запускаються миттєво без тривалих процесів компіляції та компонування, необхідних для деяких інших інструментів для подальшого пришвидшення роботи програміста.

Більшість програм, написаних на Python, працюють без проблем на різних операційних системах. Переміщення коду між Linux та Windows, наприклад, зазвичай вимагає лише копіювання скрипту з однієї машини на іншу. Крім цього, Python надає кілька можливостей для створення переносних графічних інтерфейсів користувача, доступу до баз даних, веб-систем тощо. Навіть інтерфейси операційної системи, такі як запуск програм або обробка каталогів, дуже портативні у Python.

Python має обширну колекцію готових та переносних функцій - стандартну бібліотеку. Ця бібліотека покриває ряд завдань програмування на різних рівнях - від роботи з текстовими шаблонами до мережевих сценаріїв. Будучи дуже гнучким, Python можна легко розширити за допомогою власних бібліотек чи стороннього програмного забезпечення для покращення функціональності програм. Стороння модель для Python пропонує інструменти для створеня сайтів, численного програмування, доступу до послідовного порту та навчання у галузі ігрового програмування та багато іншого. Розширення NumPy називали безкоштовною та потужною альтернативою системи числового програмування Matlab.

Python скрипти можуть легко взаємодіяти з іншими частинами програми за допомогою різноманітних механізмів інтеграції. Це дозволяє використовувати Python для налаштування та розширення продукту. На сьогоднішній день код Python може спілкуватися з бібліотеками C та C++, викликати програми на мовах C і C++, інтегруватися з компонентами Java та .NET, використовувати фреймворки, такі як COM, сполучатися з пристроями через послідовні порти та комунікувати через мережу за допомогою таких інтерфейсів, як SOAP, XML-RPC і CORBA [8].

Python виділяється серед інших мов програмування своєю легкістю використання та широким набором вбудованих інструментів. Це може перетворити процес програмування на задоволення, а не на клопіт. Ця нематеріальна перевага справді є дуже значущою, оскільки позитивно впливає на продуктивність розробників.

Серед усіх переваг Python особлива увага — це якість коду і висока продуктивність. Вони є найбільш переконливими аргументами на користь цієї мови для багатьох користувачів.

3.2 Бібліотека NumPy

NumPy, яке є скороченням від Numerical Python, вже давно займає важливе місце у сфері числових обчислень на мові програмування Python. Ця бібліотека має розгалуджену колекцію структур даних, алгоритмів і інструментів, які необхідні для багатьох наукових програм, які працюють з числовими даними у Python. NumPy включає швидкий і ефективний багатовимірний масив об'єкта ndarray для виконання поелементних операцій з масивами та математичних операцій над ними. Крім цього, ця біблтотека пропонує функціонал для читання та запису масивних даних на диск, підтримує операції лінійної алгебри, перетворення Фур'є та генерацію псевдовипадкових чисел. Завдяки зрілому та функціональному API для мови C забезпечений доступ до структур даних та обчислювальних можливостей NumPy як розширення Python так і інтеграції з кодом на C або C++. Окрім можливостей швидкої обробки масивів, NumPy є основним контейнером для передачі даних між алгоритмами та бібліотеками в аналізі даних. Масиви NumPy є більш ефективними для зберігання та обробки числових даних, ніж інші вбудовані структури даних Python. Бібліотеки, написані на мовах нижчого рівня, таких як C або Fortran, можуть працювати з даними, що зберігаються в масивах NumPy, без необхідності копіювання даних в інше представлення пам’яті.

Завдяки цьому, багато наборів числових обчислювальних інструментів для Python використовують масиви NumPy як основну структуру даних або забезпечують повну сумісність з NumPy. NumPy становить один з ключових базових пакетів для числових обчислень у Python. Багато обчислювальних пакетів, які надають науковий функціонал, користуються масивами NumPy як основною формою обміну даними.

NumPy пропонує ефективний багатомірний масив ndarray, який підтримує швидкі масивно-орієнтовані арифметичні операції та гнучкость у трансляціях. Крім того, він включає математичні функції для швидких операції над цілими массивами даних без необхідності у циклах. Додатковими можливостями є інструменти для читання/запису даних массиву на диск і роботи з файлами, відображеними в пам'яті, а також функції лінійної алгебри, генерації випадкових чисел і перетворення Фур'є. NumPy представляє C API для з’єднування до бібліотек написаних на C, C++ або Fortran. Завдяки простому у використанні C API, дані можуть легко передаватися зовнішнім бібліотекам, написаним на мовах нижчого рівня, а також можуть повертатися до Python у вигляді масивів NumPy. Це зробило Python мовою вибору для обгортання застарілих кодових, надаючи їм динамічний та простий у використанні інтерфейс.

NumPy відіграє ключову роль у чисельних обчисленнях у Python завдяки його ефективності при маніпулюванні великими обсягами даних. Це досягається за допомогою кількох методів. NumPy зберігає дані в неперервному блоку пам'яті, незалежно від інших вбудованих об'єктів Python. Алгоритми NumPy, написані на мові C, працюють з цим блоком пам'яті без необхідності перевірки типу чи інших операцій. Масиви NumPy потребують значно менше пам'яті, ніж вбудована послідовність Python. Операції NumPy забезпечують складні обчислення з цими масивами без застосування циклiv Python for.

3.3 Бібліотека Pandas

Бібліотека pandas пропонує високорівневі структури даних і функції, розроблені для того, щоб забезпечити простоту та гнучкість у роботі з структурованими або табличними даними [2]. З моменту свого запуску у 2010 році вона допомогла зробити Python потужним і ефективним середовищем для аналізу даних. Основними об'єктами в pandas є DataFrame, який представляє собою табличну структуру даних з орієнтацією на колонки із мітками як для рядків, так і для колонок, і Series, який є одномірним об'єктом масиву з мітками.

Бібліотека pandas об’єднує поняття обчислення масивами NumPy і можливостей роботи з даними, які характерні для електронних таблиць та реляційних баз даних (таких як SQL). Вона надає зручну функціональність індексування, що дозволяє змінювати формат даних, розділяти їх на частини, виконувати агрегацію та виділяти підмножини даних.

Бібліотека pandas містить інструменти обробки даних, які призначені для швидкого й простого очищення та аналізу даних у Python. Зазвичай вона використовується разом з числовими обчисленнями, такими як NumPy і SciPy, аналітичними бібліотеками типу statsmodel і scikit-learn, а також бібліотеками візуалізації даних, наприклад matplotlib. Багато функцій pandas ґрунтуються на стилевих розв’язках NumPy для масивних операцій та намагаються уникати циклів for під час обробки даних.

Хоча pandas використовує багато конструкцій кодування з NumPy, його основне призначення - робота з таблицями чи розношерстю даними. У той же час як NumPy краще підходить для однорідних чисельних масивів. Починаючи з 2010 року Pandas став проектом з відкритим кодом, який перетворився на значну бібліотеку для широкого спектра застосунків у реальних проектах. Розмір спільноти розробників Pandas перевалив за 800 учасників, що активно сприяли його подальшому розвитку завдяки застосуванням у повсякденному житті.

3.4 Платформа Google Colab

Google Colab (походження від Google Collaboratory) - це хмарний сервіс, розроблений компанією Google, який дозволяє користувачам створювати та запускати блокноти Jupyter через веб-браузер. Ця платформа надає можливість окремим людям програмувати на мові Python без необхідності налаштування локальних ресурсів. Безкоштовний сервіс Colab дозволяє вам користуватися високопродуктивним апаратним забезпеченням, таким як графічні процесори та процесори TPU, що прискорює завдання, які потребують значної обчислювальної потужності, без будь-яких обмежень.

На рисунку 3.1 зображено ознайомлюючу сторінку платформи Google Colab.

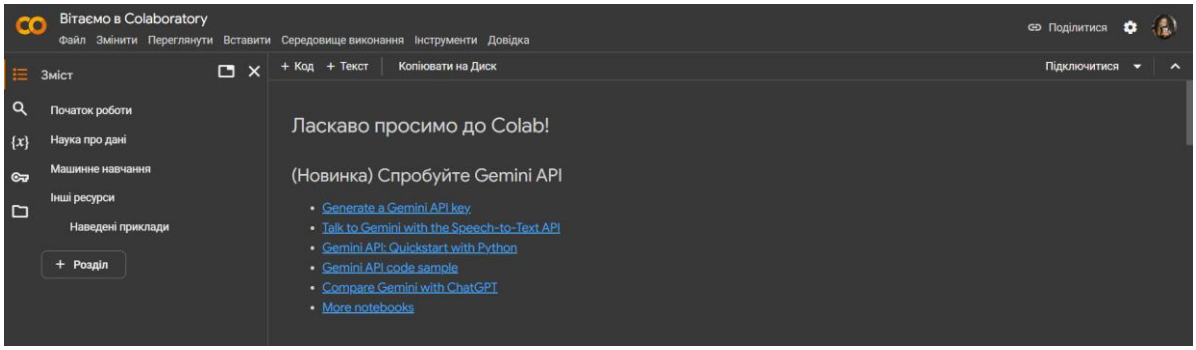

Рисунок 3.1 - Google Colab

Google Colab має один важливий аспект: просте підключення до Google Drive. Це дозволяє користувачам легко зберігати свою роботу та демонструвати її іншим. Людям надається можливість зберігати свої нотатки безпосередньо у своєму обліковому записі Google Drive та отримувати до них доступ з будь-якого місця за допомогою будь-якого пристрою, який підключений до Інтернету. Нотатки можна експортувати у форматах PDF або HTML, що робить їх універсальними для різних завдань, таких як презентації чи звіти.

Подання інтерактивної візуалізації даних є ще одним важливим компонентом. Завдяки Google Colab доступна можливість створення графіків та діаграм за допомогою багатьох бібліотек, таких як Matplotlib, Seaborn, Plotly тощо. Це особливо корисно у сфері аналітики даних і машинного навчання: коли наочне представлення може значно спростити розуміння складних наборів даних.

Google Colab - це зручний інструмент для науковців та інженерів у галузі машинного навчання. Ця платформа надає готове середовище, де вони можуть експериментувати з різними моделями та алгоритмами, оскільки вона вже містить передвстановлені бібліотеки, такі як TensorFlow, Keras, PyTorch тощо. Користувачам не потрібно витрачати час на налаштування середовища - вони можуть почати працювати безпосередньо. Крім того, інфраструктура дозволяє обробляти великі обсяги даних, що сприяє навчанню складних моделей.

Ще одна значуща особливість полягає у можливостях спільної роботи. Користувачам доступна функція спільного доступу до своїх нотаток з іншими

користувачами, що дозволяє декільком людям працюватии над одним проектом одночасно. Це особливо корисно для колективних проектів Вчителї або спіільна робота над науковим дослідженнями.

Крім того, він також забезпечує підтримку виконання команд оболонки: таким чином ви можете запускати системні команди безпосередньо з блокнота. Це, у свою чергу, відкриває шлях для додаткової гнучкості — дозволяє встановлювати пакети, завантажувати дані або виконувати будь-які інші системні операції безпосередньо з середовища Colab.

Однією з ключових причин популярності Google Colab серед багатьох користувачів є можливість безкоштовного доступу до високопродуктивних апаратних ресурсів. Користувачам дозволяється використовувати GPU та TPU без будь-яких додаткових витрат, що значно покращує продуктивність, особливо при обчисленнях великого обсягу. Хоча існують обмеження на тривалість сесій та споживання ресурсів, ці обмеження можна вважати досить гнучкими для багатьох сценаріїв використання. Загалом, Google Colab є потужним інструментом, який бажають мати розробники, дослідники та студенти, яким потрібен доступ до надійних обчислювальних ресурсів у середовищі Python без необхідності у налаштуваання локального середовища. Його інтеграція з Google Drive, можливості спільної роботи над проектами, засоби для визначення та доступ до GPU/TPU роблять його незамінною платформою для користувачіv по всьому світу.

3.5 Фреймворк Streamlit

Streamlit спеціально розроблений для розробників, які займаються даними, інженерів з машинного навчання та науковців. Він з’явився у 2019 році та швидко здобув популярність завдяки своїй простоті, яка дозволяє створювати повноцінні веб-програми за допомогою Python без необхідності мати спеціальні навички веб-розробки. Це досягається з мінімальними зусиллями, що робить його ідеальним інструментом для швидкої та ефективної реалізації ідей у вигляді інтерактивних веб-додатків [1].

В основі Streamlit лежить проста, але потужна ідея: зробити процес створення веб-додатків легким і простим. Розробники можуть використовувати свої навички Python для створення додатків, а Streamlit безперешкодно впорається з презентацією та аспектами взаємодії з користувачем від їхнього імені. Це дозволяє зосередитися на тонкощах логіки програми та маніпуляції даними, звільняючи від необхідності займатися HTML, CSS або JavaScript.

Інтерактивність є головною перевагою Streamlit. Оскільки фреймворк дозволяє використовувати такі елементи інтерфейсу користувача, як повзунки, кнопки та текстові поля в додатку в реальному часі, користувачі можуть взаємодіяти з ним. Це робить його ідеальним інструментом для створення інтерактивних панелей управління, аналітичних інструментів та демонстрацій моделей машинного навчання.

Також варто відзначити простоту Streamlit. Створення додатків з його використанням не потребує багато часу та енергії; розробнику просто потрібно прийняти декларативний підхід, коли він описує, що має бути на сторінці, а фреймворк автоматично створює інтерфейс. Це призводить до скорочення значної кількості часу як для прототипів, так і для кінцевих продуктів.

Крім того, Streamlit легко інтегрується з іншими інструментами для роботи з даними та наборами машинного навчання. Наприклад, ви можете використовувати Pandas для обробки даних, Matplotlib або Seaborn для створення візуалізацій, а TensorFlow або PyTorch для побудови моделей машинного навчання. Відображення графіків і візуальних елементів у Streamlit не вимагає додаткових зусиль — вони автоматично відображаються в інтерфейсі веб-додатку, що значно спрощує роботу порівняно з традиційним підходом, що вимагає знання HTML, CSS або JavaScript. Це особливо корисно для швидкого розуміння даних, отриманих під час аналізу.

Ще одна чудова характеристика — здатність швидко розгортати програми. Streamlit-додатки можуть бути легко розгорнуті на різних платформах, таких як Heroku або власні сервери. Це робить їх доступними для ширшої аудиторії без значних витрат часу та ресурсів. Можливість швидкого розгортання та поширення

додатків дозволяє розробникам швидко ділитися своїми проектами та результатами з колегами або клієнтами.

Streamlit також підтримує ефективну співпрацю, що є важливою характеристикою для розробників і дослідницьких груп. Розробники можуть легко обмінюватися своїми програмами з іншими членами команди, які можуть використовувати програму, змінювати її, вносити зміни та пропонувати вдосконалення. Це робить співпрацю більш ефективною та прискорює процес розробки.

Крім того, Streamlit є відкритим проєктом із широкою базою користувачів і активною спільнотою розробників, які постійно вдосконалюють фреймворк, додаючи нові функції. Спільнота є багатим джерелом матеріалів, включаючи документацію, приклади коду та форуми підтримки, що робить початок роботи зі Streamlit простішим і приємнішим.

Загалом, Streamlit є ефективним інструментом для швидкої та легкої розробки інтерактивних веб-додатків для аналізу даних, машинного навчання та інших задач. Його простота, інтерактивність, можливості інтеграції та підтримка співпраці роблять його ідеальним вибором для розробників і дослідників, які хочуть швидко й ефективно візуалізувати та аналізувати дані.

3.6 База даних Redis

Redis — це віддалена база даних у пам’яті, яка пропонує високу продуктивність, реплікацію та унікальну модель даних для створення платформи для вирішення проблем [3]. Підтримуючи п’ять різних типів структур даних, Redis вміщує широкий спектр проблем, які можна природним чином відобразити в тому, що пропонує Redis, дозволяючи вирішувати проблеми без виконання концептуальної гімнастики, необхідної для інших баз даних. Додаткові функції, такі як реплікація, постійність і сегментування на стороні клієнта, дозволяють Redis масштабуватись від зручного способу створення прототипу системи до сотень гігабайт даних і мільйонів запитів на секунду.

Redis є дуже швидка нереляційна база даних, яка зберігає відображення ключів на п’ять різних типів значень. Redis підтримує постійне зберігання в пам’яті на диску, реплікацію для масштабування продуктивності читання та сегментування на стороні клієнта для масштабування продуктивності запису.

Redis зазвичай називають NoSQL або нереляційною. У Redis немає таблиць, і немає визначеного базою даних або примусового способу зв’язування даних у Redis з іншими даними в Redis.

Нерідко можна почути порівняння Redis з memcached, який є дуже високопродуктивним сервером кешу типу "ключ-значення". Подібно до memcached, Redis також може зберігати відображення ключів у значення та навіть може досягати таких же рівнів продуктивності, як і memcached. Але подібності швидко закінчуються — Redis підтримує автоматичний запис своїх даних на диск двома різними способами та може зберігати дані в чотирьох структурах на додаток до простих рядкових ключів, як це робить memcached. Ці та інші відмінності дозволяють Redis вирішувати ширший спектр проблем і дозволяють Redis використовувати як основну базу даних або як допоміжну базу даних з іншими системами зберігання.

Під час використання баз даних у пам'яті, таких як Redis, важливо враховувати питання збереження даних у випадку вимкнення сервера. Redis пропонує два різні методи збереження даних у пам'яті на диск у компактному форматі.

Перший метод передбачає створення дампа на певний момент часу або при виконанні певних умов, таких як кількість записів за певний період, або при виклику однієї з двох команд для створення дампу на диск. Інший метод використовує файл лише для додавання, який записує кожну команду, що змінює дані в Redis, на диск у міру її виконання. В залежності від необхідного рівня надійності, запис у файл лише для додавання можна налаштувати таким чином, щоб він ніколи не синхронізувався, синхронізувався раз на секунду або синхронізувався після завершення кожної операції.

Незважаючи на ефективність Redis, через його архітектуру в пам'яті можуть виникати ситуації, коли потрібна обробка більшої кількості запитів на читання, ніж

може впоратися один сервер Redis. Для забезпечення вищої продуктивності читання та відновлення після збоїв, Redis підтримує реплікацію за схемою «головний/підлеглий». У цій конфігурації підлеглі сервери підключаються до головного сервера та отримують початкову копію повної бази даних. Після цього всі зміни, що відбуваються на головному сервері, надсилаються всім підключеним підлеглим у режимі реального часу для оновлення їхніх наборів даних. Завдяки постійному оновленню даних на підлеглих серверах, клієнти можуть підключатися до будь-якого з них для читання, замість того щоб надсилати запити до головного сервера. Таким чином, використання реплікації в Redis дозволяє не лише збільшити продуктивність читання, але й забезпечує надійність та відмовостійкість системи в цілому.

3.7 Програмне забезпечення Docker

Docker — це командний інтерфейс і набір віддалених служб, що використовують логістичний підхід для вирішення типових проблем із програмним забезпеченням та спрощують процеси встановлення, запуску, публікації й видалення програмного забезпечення. Це досягається за допомогою технології контейнерів UNIX.

Історично склалося так, що операційні системи типу UNIX використовували термін "в'язниця" для опису модифікованого середовища виконання для програми, яке обмежувало доступ до захищених ресурсів. Після випуску Sun Solaris 10 у 2005 році та впровадження Solaris Containers, термін "контейнер" став загальновживаним для такого середовища виконання. Мета контейнерів полягає не лише у запобіганні доступу до захищених ресурсів, а й у повній ізоляції процесів від усіх ресурсів, за винятком випадків, коли це явно дозволено.

Використання контейнерів стало найкращою практикою у програмній інженерії. Проте створення контейнерів вручну може бути складним і схильним до помилок процесом. Це обмежувало доступність цієї технології для деяких користувачів, тоді як неправильно налаштовані контейнери створювали хибне

відчуття безпеки для інших. Будь-яке програмне забезпечення, що запускається з Docker, працює всередині контейнера. Docker використовує існуючі механізми контейнерів для забезпечення узгоджених контейнерів, створених відповідно до найкращих практик, що робить надійну безпеку доступною для всіх.

Використання Docker дозволяє отримувати контейнери за значно нижчою ціною. У міру вдосконалення Docker і його механізмів контейнерів користувачі отримують найновіші та найкращі функції. Замість того, щоб слідкувати за швидко змінюваним і високотехнологічним світом створення потужних контейнерів додатків, користувачі можуть дозволити Docker виконувати основну частину цієї роботи, що заощаджує час і ресурси.

Керування програмним забезпеченням є складним процесом. Під час інсталяції необхідно враховувати операційну систему, необхідні ресурси, наявне програмне забезпечення та його залежності. Потрібно визначити місце встановлення та знати, як його виконати. Процеси встановлення часто значно відрізняються між собою, що робить їх непослідовними і складними. Чим більше програмного забезпечення використовується, тим складніше ним керувати, а також зростає ризик вразливостей до атак.

Усі ці проблеми можна вирішити за допомогою ретельного обліку, управління ресурсами та логістики, проте це часто є рутинною роботою. Розробники Docker врахували ці виклики і створили інструмент, який дозволяє пройти через ці процеси з мінімальними зусиллями та у короткий термін.

Docker є сучасним інструментом, що дозволяє розробникам і системним адміністраторам автоматизувати процеси розгортання, масштабування та управління програмами. Docker вирішує проблеми сумісності та залежностей програмного забезпечення шляхом створення ізольованих контейнерів, які можуть працювати незалежно від основної операційної системи. Це значно спрощує розробку та розгортання додатків, роблячи процеси швидшими і менш схильними до помилок. Його здатність забезпечити ізольовані, відтворювані та безпечні середовища для додатків робить його важливим інструментом для будь-якої організації, що прагне досягти високої ефективності та надійності у своїй IT-інфраструктурі. Docker не тільки спрощує процеси розробки та розгортання, але й

забезпечує високу ступінь гнучкості, дозволяючи швидко адаптуватися до змінних вимог і умов.

Крім того, Docker активно підтримує та розвиває екосистему навколо своїх інструментів, включаючи Docker Hub — репозиторій образів контейнерів. Docker Hub дозволяє розробникам ділитися своїми образами, що сприяє спільному використанню і повторному використанню компонентів програмного забезпечення. Це значно спрощує процес розробки, оскільки розробники можуть легко знайти та використовувати готові рішення для своїх додатків.

Основною перевагою Docker є його здатність забезпечити стабільність і відтворюваність середовища розробки. Розробники можуть бути впевнені, що код, який працює в їхньому середовищі, буде працювати так само у виробничому середовищі. Це досягається шляхом контейнеризації додатків разом з усіма їхніми залежностями і конфігураціями.

4 ОПИС ПРОГРАМНОЇ РЕАЛІЗАЦІЇ

Цей розділ присвячений докладному опису реалізації системи, яка виконує аналіз тональності тексту та змінює його емоційне забарвлення. Описані тут кроки є основоположними для розробки системи та впровадження необхідних функцій. Будуть представлені діаграма прецедентів та діаграма розгортання веб-системи, що демонструє її фізичну архітектуру та основні компоненти. Опис процесу розробки почнемо з налаштування моделі для аналізу тональності, яка базується на моделі RoBERTa. Буде показано як налаштувати модель для класифікації тексту за тональністю, а також як використовувати токенізатор для підготовки текстових даних.

Далі розлянуто, як система інтегрується з базою даних Redis для зберігання результатів аналізу.

Окрему увагу приділено використанню API OpenAI для переписування тексту з іншою тональністю. Буде описано, як формувати запити до моделі та обробляти відповіді для генерації нового тексту.

4.1 Функціональна декомпозиція

Діаграма прецедентів (англ. Use Case Diagram) — це тип діаграми в моделюванні систем, яка описує функціональні вимоги до системи з точки зору користувача [12]. Вона є одним з основних засобів візуалізації в рамках методології об'єктно-орієнтованого аналізу та проєктування UML (Unified Modeling Language).

Діаграма прецедентів, яка зображена на рисунку 4.1, була створена для моделювання функціональних можливостей системи з точки зору її користувачів, а також для визначення взаємодії між акторами (користувачами, системами або зовнішніми компонентами) і системою.

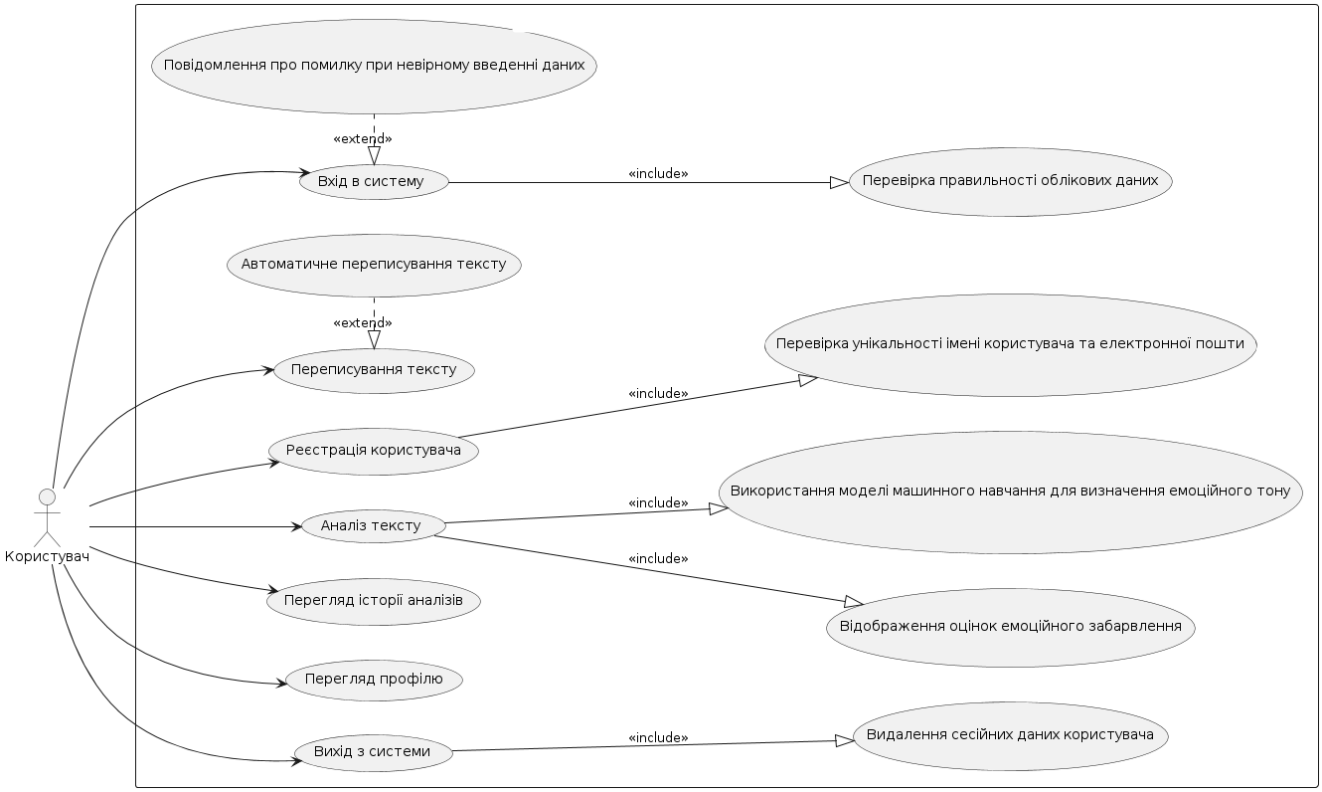

Рисунок 4.1 – Діаграма прецедентів системи

Актор та його можливості в діаграмі прецедентів:

Актор «Користувач»: Це зареєстрований користувач програми для аналізу емоційного забарвлення текстів. Програма надає йому можливості взаємодії з системою для отримання результатів аналізу та управління своїм акаунтом. Нижче описано всі можливості користувача:

«Реєстрація користувача»: Користувач може зареєструватися в системі, надавши необхідну інформацію, таку як ім'я користувача, електронна пошта і пароль. Процес реєстрації включає перевірку унікальності наданого імені користувача та електронної пошти. Після успішної реєстрації користувач отримує підтвердження у вигляді повідомлення на екрані.

«Вхід в систему»: Після реєстрації користувач може увійти в систему, використовуючи свої облікові дані (ім'я користувача та пароль). Система перевіряє правильність наданих облікових даних і надає доступ до функціоналу програми в разі успішної автентифікації. У випадку невірного введення даних користувач отримує повідомлення про помилку.

Аналіз тексту: Користувач може ввести текст для аналізу і отримати результати аналізу емоційного забарвлення. Система використовує модель машинного навчання для визначення емоційного тону тексту (позитивний, нейтральний або негативний). Результати аналізу відображаються у вигляді оцінок емоційного забарвлення та загального висновку.

Переписування тексту: Користувач може переписати текст залежно від результатів аналізу, щоб змінити його емоційне забарвлення. Система надає рекомендації або автоматично переписує текст для досягнення бажаного емоційного тону. Користувач може порівняти оригінальний текст та переписаний варіант.

Перегляд історії аналізів: Користувач може переглядати свою історію попередніх аналізів. Історія включає збережені тексти, результати аналізу, результати переписування. дату та час проведення аналізу та переписування. Користувач може сортувати та фільтрувати історію для зручності пошуку потрібних записів.

Перегляд профілю: Користувач може переглядати та редагувати свій профіль. Профіль містить інформацію, таку як ім'я користувача, електронна пошта та інші персональні дані.

Вихід з системи: Користувач може вийти зі свого акаунту, завершуючи поточну сесію. Після виходу користувач повинен буде повторно увійти, щоб знову отримати доступ до функціоналу програми. Система забезпечує безпечний вихід, видаляючи всі сесійні дані користувача.

4.2 Діаграма розгортання веб-системи

Для реалізації веб-сервісу була використана клієнт-серверна архітектура. Вона складається з клієнтської частини (Python, Streamlit) та серверної (Redis) і використовує клієнтсько-серверний протокол для забезпечення комунікації між клієнтом і сервером.

Діаграма розгортання, яка зображена на рисунку 4.2, відображає фізичну конфігурацію та розгортання компонентів системи на апаратному та програмному забезпеченні.

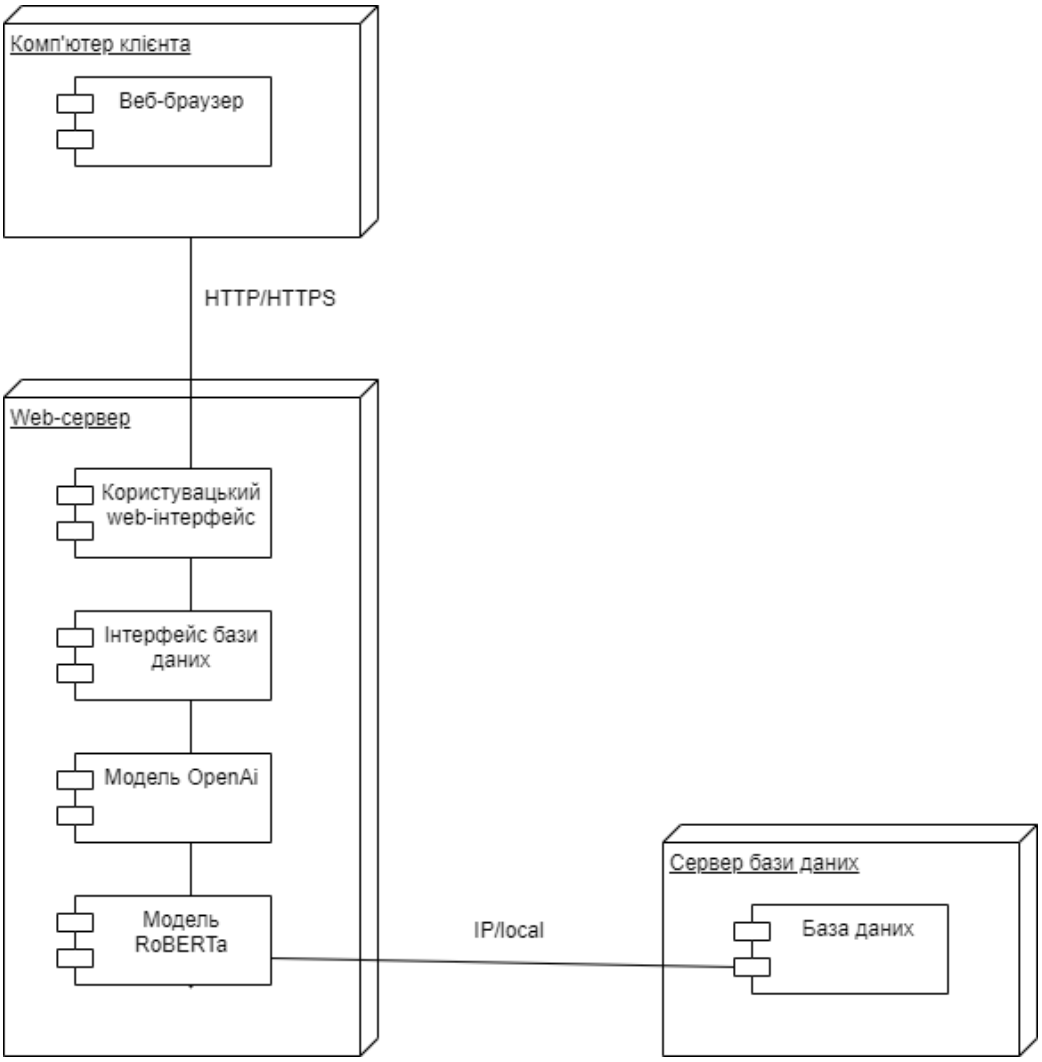

Рисунок 4.2 – діаграма розгортання веб-системи

Система складається з трьох основних компонентів: клієнтської частини, веб-сервера та сервера бази даних. Ця архітектура забезпечує гнучкість, масштабованість та надійність системи.

Розглянемо комп’ютер клієнта. На ньому знаходиться веб-браузер, за допомогою якого взаємодіє з системою. Веб-браузер відправляє HTTP/HTTPS запити до веб-сервера і отримує відповіді, які відображаються користувачеві.

Розглянемо веб-сервер. На ньому розташовані такі модулі: користувацький веб-інтерфейс, інтерфейс бази даних, модель OpenAI, модель RoBERTa.

Користувацький веб-інтерфейс обробляє запити від клієнтського веб-браузера і надає користувацький інтерфейс, з яким взаємодіє користувач. Він забезпечує функції реєстрації, входу, аналізу тексту, перегляду історії та інші функції.

Інтерфейс бази даних забезпечую взаємодію з сервером бази даних. Відправляє запити на зчитування і запис даних, необхідних для роботи додатку.

Модель OpenAI використовується для генерації переписаних текстів на основі результатів аналізу емоційного забарвлення.

Модель RoBERTa використовується для аналізу емоційного забарвлення тексту. Ця модель завантажується на сервері і обробляє введені користувачем тексти.

Сервер бази даних зберігає всі дані, необхідні для функціонування системи, включаючи дані користувачів, результати аналізу та інші метадані. База даних реалізована на основі Redis, що забезпечує швидкий доступ до даних.

Ця архітектура вирізняється гнучкістю, масштабованістю та високою продуктивністю. Вона дає змогу користувачам зручно взаємодіяти з системою через веб-інтерфейс, де вони можуть аналізувати тексти, зберігати результати та отримувати переписані текстові варіанти. Веб-сервер здійснює всі необхідні обчислення та взаємодіє з базою даних, забезпечуючи надійне зберігання інформації

4.3 Налаштування моделі для аналізу тональності тексту

Використання мовних моделей, таких як RoBERTa (Robustly optimized BERT approach), стало революційним кроком у сфері обробки природної мови (NLP). Ці моделі здатні розуміти та аналізувати текст з винятковою точністю, що робить їх незамінними в багатьох додатках, включаючи аналіз емоційного забарвлення текстів. У цій дипломній роботі детально описуються всі процеси налаштування моделі RoBERTa для аналізу твітів, починаючи від встановлення необхідних бібліотек і закінчуючи розгортанням готової моделі.

Першим і основним кроком у підготовці до роботи з будь-якою мовною моделлю є встановлення відповідного програмного забезпечення та бібліотек. Це важливий етап, який створює основу для подальшої роботи з даними та моделями.

Бібліотека transformers, розроблена компанією Hugging Face, є однією з найбільш популярних і широко використовуваних у сфері NLP. Вона містить великий набір моделей, таких як BERT, GPT, RoBERTa та інші, які можна легко використовувати для різноманітних завдань, включаючи класифікацію тексту, генерацію тексту, переклад та інше. Встановлення цієї бібліотеки забезпечує доступ до потужних попередньо натренованих моделей, що значно знижує час та ресурси, необхідні для розробки NLP-додатків.

PyTorch — це бібліотека для глибинного навчання, розроблена Facebook AI Research (FAIR). Вона забезпечує гнучкість і швидкість у розробці нейронних мереж і є основою для багатьох сучасних досліджень і розробок у галузі штучного інтелекту. PyTorch підтримує автоматичне диференціювання, що робить процес навчання моделей більш ефективним і зручним.

Scikit-learn є ключовою бібліотекою для машинного навчання в Python. Вона включає численні інструменти для класифікації, регресії, кластеризації та оцінки моделей. Ця бібліотека також забезпечує широкий спектр функцій для попередньої обробки даних, таких як нормалізація, вибір ознак та розділення даних на тренувальні та тестові набори.

NumPy і Pandas є базовими бібліотеками для наукових обчислень та аналізу даних. NumPy забезпечує підтримку багатовимірних масивів і матриць, а також математичних функцій для їх обробки. Pandas дозволяє ефективно маніпулювати табличними даними та виконувати різноманітні операції з ними, такі як фільтрація, групування та агрегація даних.

Matplotlib та Seaborn є бібліотеками для візуалізації даних. Matplotlib надає широкий набір інструментів для створення графіків і діаграм, тоді як Seaborn забезпечує додаткові можливості для статистичної візуалізації та спрощує процес створення складних графіків.

Після встановлення необхідних бібліотек наступним кроком є завантаження та попередня обробка даних. Це один з найважливіших етапів у підготовці до

навчання моделі, оскільки якість даних безпосередньо впливає на результати навчання.

Дані можуть бути отримані з різних джерел, таких як файли CSV, бази даних або API. Важливо забезпечити правильне завантаження даних у датафрейм, що дозволяє зручно маніпулювати ними та виконувати необхідні операції з обробки.

Процес очищення даних включає кілька етапів. Спочатку необхідно видалити дублікатні записи, які можуть спотворити результати навчання. Далі слід очистити текст від непотрібних символів, таких як HTML-теги, URL-адреси, спеціальні символи тощо. Це покращує розуміння тексту моделлю, оскільки видаляє шум та залишає лише корисну інформацію.

За цим відбувається процес токенізації. Це процес розбиття тексту на окремі елементи, які називаються токенами. Токени можуть бути словами, фразами, символами або іншими значущими частинами тексту. У випадку з RoBERTa використовується спеціальний токенайзер, який переводить текст у числові значення, зрозумілі моделі. Це важливий етап, оскільки модель працює саме з числовими представленнями тексту.

Після очищення даних необхідно забезпечити їх правильне балансування та розподіл на тренувальні, валідаційні та тестові набори.

Балансування даних є критично важливим для задачі класифікації, особливо коли класи нерівномірно представлені. Наприклад, у наборі даних може бути значно більше негативних твітів, ніж позитивних або нейтральних. У таких випадках модель може навчитися віддавати перевагу більш представленим класам, що призведе до поганих результатів для менш представлених класів. Балансування даних можна досягти за допомогою довибірки (oversampling) менш представлених класів або відбору частини даних з найбільш представленого класу.

Балансування даних ділиться на три основні набори: тренувальний, валідаційний і тестовий.

Тренувальний набір використовується для навчання моделі. Цей набір даних повинен бути найбільшим, щоб забезпечити моделі достатньо інформації для навчання.

Валідаційний набір використовується для оцінки моделі під час навчання та налаштування гіперпараметрів. Цей набір дозволяє виявити переобучення (overfitting) та підобучення (underfitting).

Тестовий набір використовується для остаточної оцінки моделі після навчання. Цей набір даних повинен бути абсолютно незалежним від тренувального та валідаційного наборів, щоб забезпечити об'єктивну оцінку ефективності моделі.

Для роботи з категорійними мітками, такими як емоційне забарвлення твітів, використано метод One-Hot Encoding. Він перетворює категорійні змінні у числові вектори, які можуть бути використані моделлю для навчання. Цей метод запобігає впливу відносних масштабів, тобто кожен категорійний клас представлений окремим бінарним вектором, що виключає можливість того, що модель буде інтерпретувати певні значення як маючі більше значення, ніж інші.

Наступним кроком було створення та компіляція моделі.

Завантажено попередньо натренеровану модель RoBERTa на платформі Google Colab. Базова модель вже навчена на великому корпусі текстів , що дозволяє їй розуміти загальні закономірності мови.

Для задачі класифікації додано спеціалізований вихідний шар, який відповідає за передбачення класів. Цей шар приймає виходи базової моделі RoBERTa і перетворює їх на ймовірності для кожного з класів (позитивний, нейтральний, негативний).

Після створення архітектури моделі її було скомпільовано, тобто вибрано оптимізатор та функцію втрат. Оптимізатор AdamW є популярним вибором для тренування моделей глибинного навчання завдяки своїй ефективності та стабільності. Функція втрат (наприклад, крос-ентропія) визначає, як оцінювати різницю між передбаченими та фактичними мітками під час навчання.

Для навчання моделі було підготовлено дані у вигляді датасетів, які легко подаються у модель. Для цього використовуються спеціальні класи, що забезпечують зручне завантаження даних у пакетах (батчах), оптимізуючи процес навчання.

Після навчання моделі використано матрицю конфузій та точність для оцінки її ефективності. Вона показує загальну ефективність моделі, вимірюючи відсоток

правильних передбачень. Висока точність свідчить про те, що модель добре справляється зі своїм завданням. На рисунку 4.3 зображено матрицю класифікації.

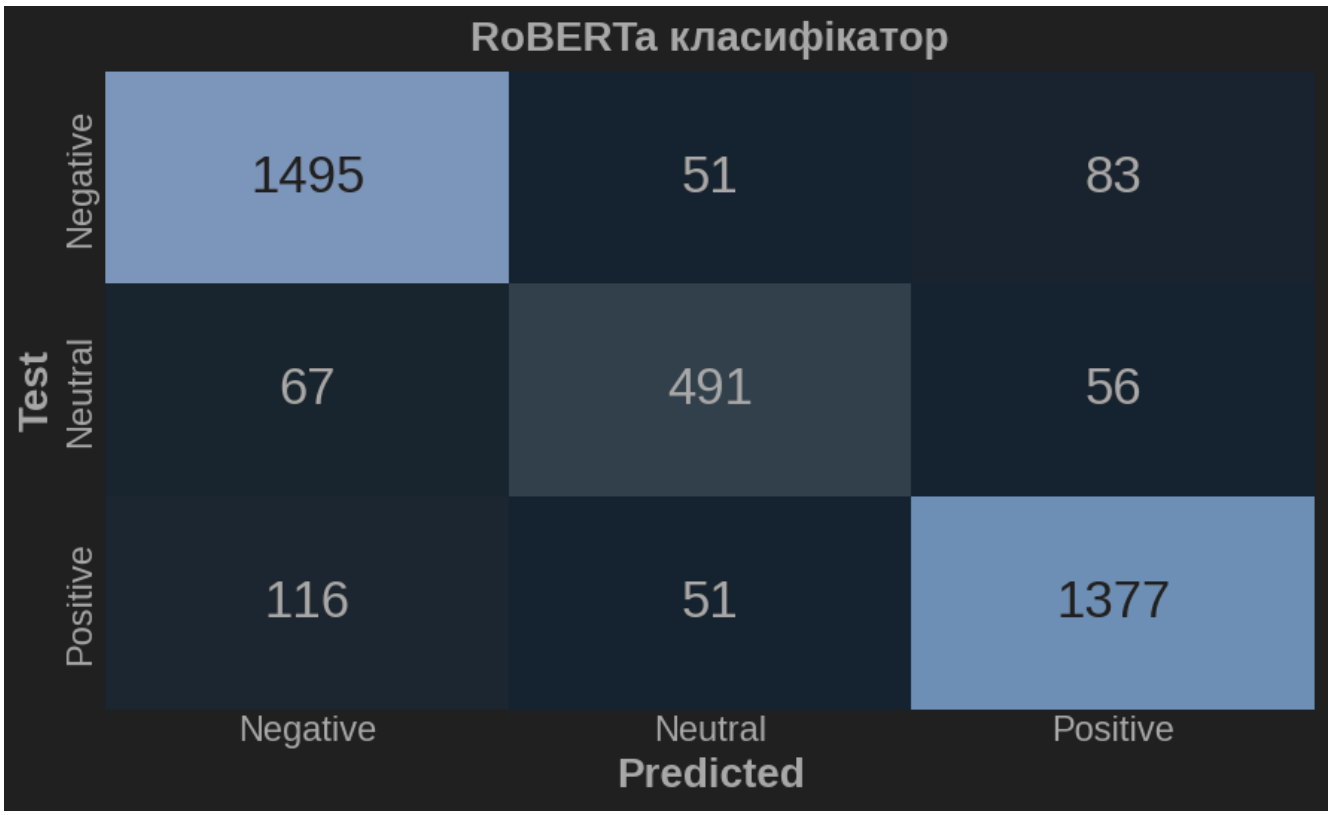

Рисунок 4.3 – матриця класифікації

На рисунку 4.3 видно, що алгоритм добре впорався із завданням класифікації, з результатами ефективності близько 90%.

4.4 Процес інтеграції Redis з системою

Інтеграція Redis з системою аналізу емоційного забарвлення текстів включає кілька основних кроків: підключення до бази даних, зберігання результатів аналізу та отримання збережених даних.

Першим кроком є налаштування підключення до бази даних Redis. Це передбачає встановлення відповідного клієнта для Python, наприклад redis-py, і

налаштування параметрів підключення. Зазвичай Redis працює на порту 6379, але ці налаштування можуть бути змінені залежно від конфігурації сервера.

Після підключення до бази даних реалізовано механізм зберігання результатів аналізу. Кожен аналізований текст може мати унікальний ідентифікатор (ID), за яким зберігаються результати аналізу. Це дозволяє легко отримувати результати для конкретного тексту в майбутньому.

Коли користувач запитує результати аналізу для певного тексту, система повинна мати можливість швидко отримати ці дані з Redis. Завдяки високій продуктивності Redis, цей процес займає мінімальний час, забезпечуючи миттєву відповідь на запити користувачів.

4.5 API OpenAI для переписування тексту

Інтеграція OpenAI API у програму для переписування тексту з іншою тональністю є важливою складовою функціоналу. Ця частина системи використовує передові моделі на основі GPT-3.5 для генерації тексту з різним емоційним забарвленням.

На першому етапі відбувається підготовка для роботи з OpenAI API. Це включає завантаження необхідних налаштувань та створення клієнта для взаємодії з API. Основним компонентом цього етапу є завантаження змінних середовища з файлу .env, де зберігається API-ключ для доступу до OpenAI API. Цей ключ є критично важливим, оскільки він забезпечує авторизацію та дозволяє використовувати сервіси OpenAI.

Після ініціалізації клієнта OpenAI підготовується запит до моделі. Формування запиту включає створення спеціального текстового повідомлення, яке буде відправлено моделі GPT-3.5. Це повідомлення або prompt сформульоване таким чином, щоб модель чітко розуміла завдання. Залежно від емоційного забарвлення вихідного тексту, яке було визначено раніше (негативне, нейтральне або позитивне), формулюється відповідний запит.

Після формування запиту відбувається взаємодія з OpenAI API. Для цього використовується метод, який відправляє запит до моделі GPT-3.5 та отримує відповідь. Цей етап забезпечує виконання запиту та отримання результату від моделі. Важливо правильно налаштувати параметри запиту, щоб отримати максимально релевантну відповідь.

Після отримання відповіді від моделі GPT-3.5 оброблює результат. Відповідь зазвичай містить згенерований текст, який є переписаним варіантом вихідного тексту з іншим емоційним забарвленням. Цей текст витягується із структури відповіді та використовується для подальшого аналізу або збереження.

Важливим аспектом роботи з OpenAI API є також моніторинг та логування всіх запитів та відповідей. Це дозволяє аналізувати роботу моделі, відслідковувати можливі помилки або недоліки у формулюванні запитів, а також вдосконалювати алгоритми генерації тексту на основі отриманих даних.

Інтеграція OpenAI API у програму для переписування тексту з іншою тональністю відкриває широкі можливості для автоматизації процесів редагування та адаптації текстів. Завдяки використанню передових моделей на основі GPT-3.5, програма забезпечує високу точність та релевантність результатів, що є критично важливим для професійного використання у різних сферах, включаючи маркетинг, журналістику, навчання та багато інших.

5 ОПИС ВЗАЄМОДІЇ КОРИСТУВАЧА ІЗ СИСТЕМОЮ

Для доступу до веб-застосунку достатньо перейти за посиланням, за умови навності доступу до інтернету та встановленого веб-браузеру.

При першому запуску системи, користувача зустрічає вікно входу (рисунок 5.1), що містить два поля для введення даних: «username» та «password», а також кнопку для підтвердження входу. Користувач повинен ввести свої облікові дані та натиснути кнопку для входу до системи.

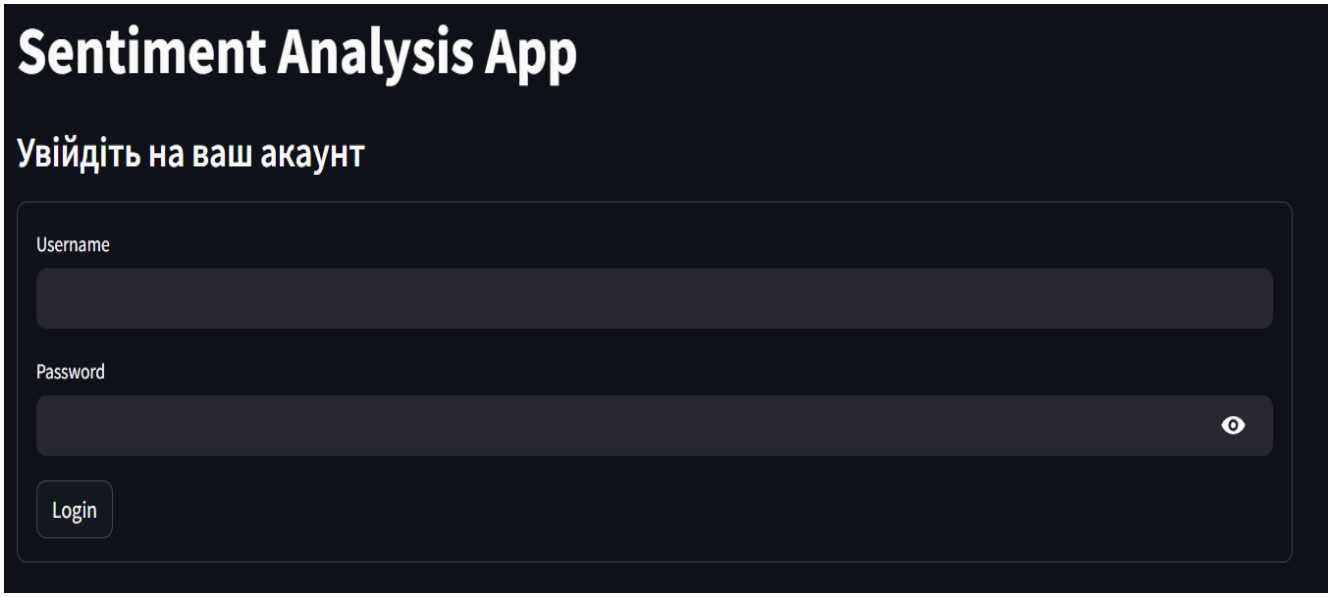

Рисунок 5.1 – Форма входу

Якщо користувач не має облікового запису, він може зареєструватися, перейшовши за посиланням "Register" (рисунок 5.2). На цій сторінці представлена форма реєстрації нового користувача, яка містить поля для введення імені користувача, електронної пошти та паролю. Після заповнення всіх полів валідними даними і натискання кнопки "Register", обліковий запис користувача буде створено, і він зможе увійти до системи. При введені некоректних даних на екрані з’являєтья повідомлення про помилку реєстрації.

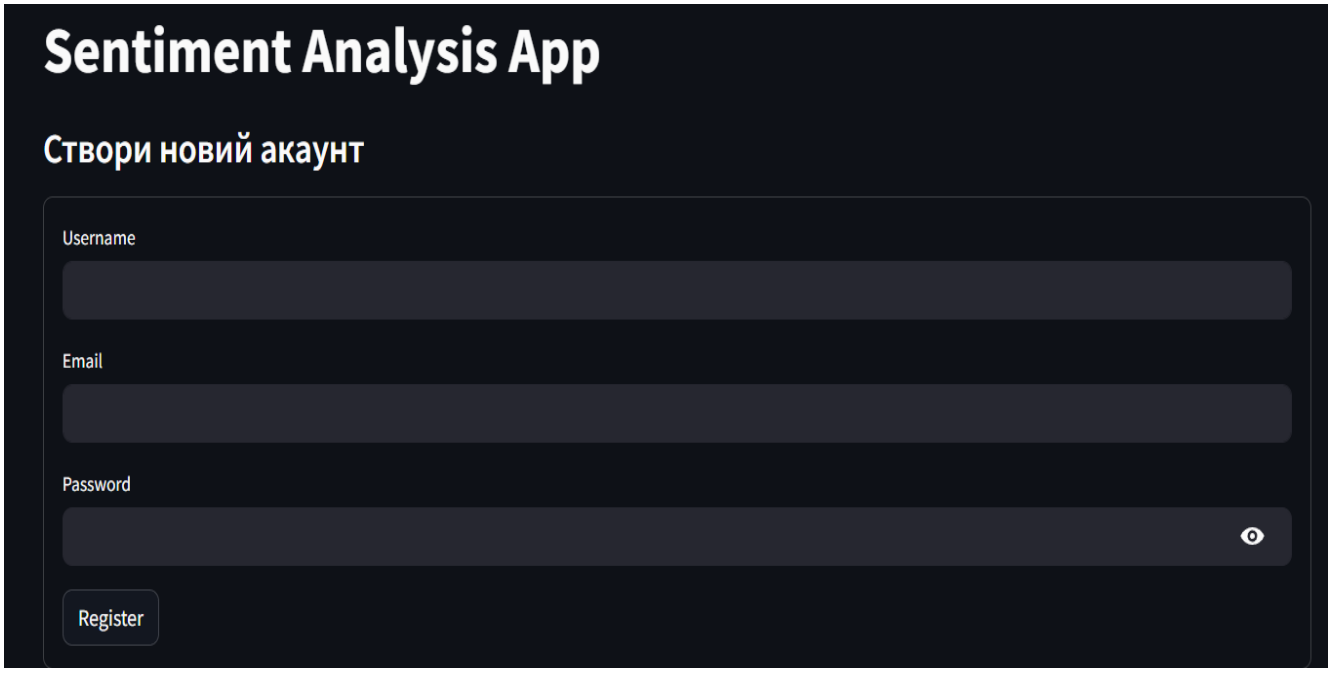

Рисунок 5.2 – Форма реєстрації

Форма реєстрації на рисунку 5.2 відзначається простотою та зрозумілістю. Користувач легко може заповнити необхідні поля, натиснути кнопку "Register" і створити свій обліковий запис. Це важливий крок для входу до системи, де користувач отримує доступ до різноманітних функцій та може взаємодіяти з іншими користувачами.

Після успішного входу до системи користувач потрапляє на головну сторінку додатку (рисунок 5.3). На цій сторінці відображається привітання та загальна інформація про додаток, його функціональність та переваги використання аналізу сентименту для бізнесу та особистих цілей.

Щоб зробити навігацію ще зручнішою, в додатку передбачена бічна панель, яка містить швидкі посилання на основні розділи та функції системи. Ця панель дозволяє користувачам швидко перемикатися між різними частинами додатку, такими як розділи аналітики, налаштування облікового запису, повідомлення та інструменти звітності. Бічна панель завжди доступна незалежно від того, на якій сторінці додатку знаходиться користувач, що забезпечує безперервний доступ до всіх важливих функцій.

Рисунок 5.3 – Головна сторінка

Для аналізу тексту користувач може перейти на сторінку аналізу тексту (рисунок 5.4).

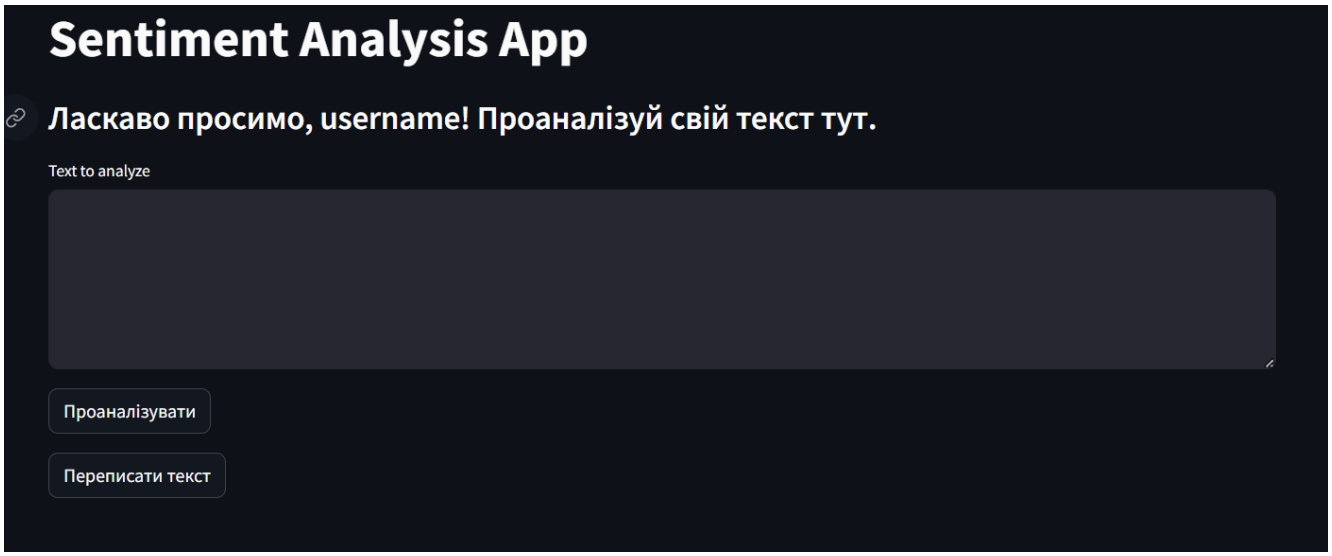

Рисунок 5.4 –Аналіз тональності тексту

У вікні аналізу користувач може ввести текст у відповідне поле та натиснути кнопку "Проаналізувати" (рисунок 5.4).

Після натискання кнопки, система проведе аналіз сентименту тексту (рисунок 5.5), використовуючи модель Roberta, і відобразить результати аналізу, вказуючи емоційне забарвлення тексту (негативне, нейтральне або позитивне).

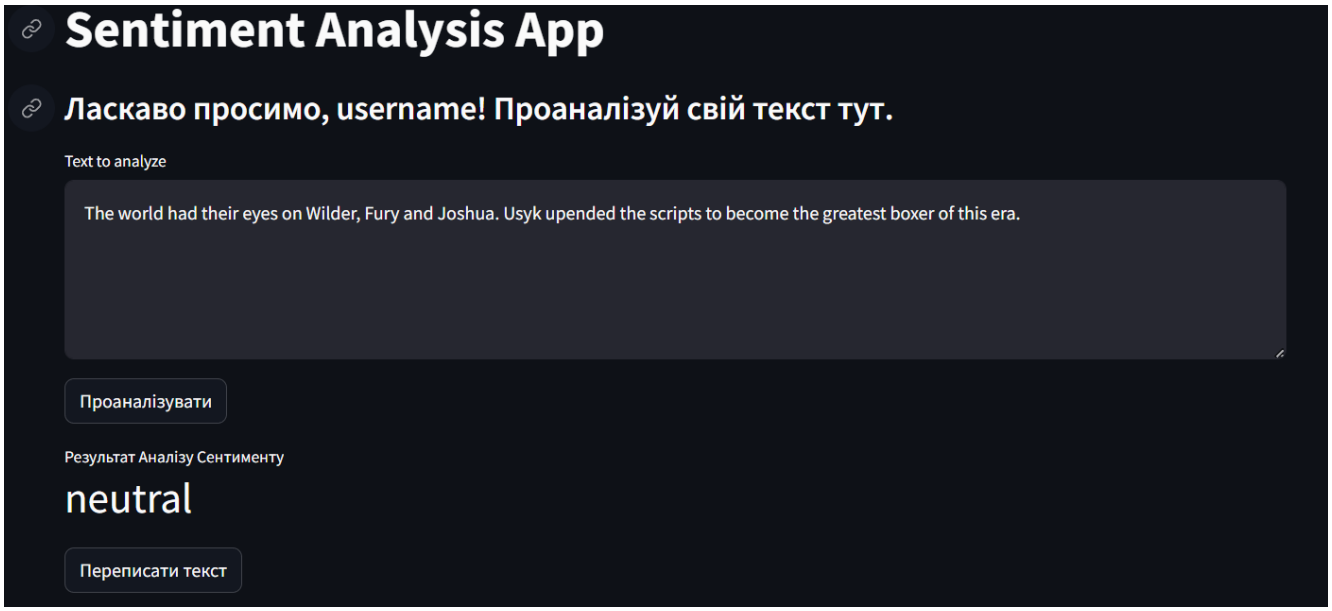

Рисунок 5.5 – Результат аналізу тональності тексту

Після отримання результатів аналізу користувач може скористатися функцією переписування тексту (рисунок 5.6). Натиснувши кнопку "Переписати текст", користувач отримає переписаний текст зі зміненим сентиментом за допомогою моделі GPT 3-5. Новий текст відобразиться на сторінці разом із результатами нового аналізу сентименту.

Цей новий текст може мати змінений сентимент, що дає користувачеві можливість порівняти його з оригінальним текстом і оцінити, наскільки ефективно були досягнуті бажані зміни. Функція переписування тексту є надзвичайно корисною для тих, хто хоче оптимізувати свої повідомлення для певної аудиторії або контексту. Наприклад, маркетологи можуть використовувати цю функцію для створення більш привабливих рекламних повідомлень, а представники служб підтримки клієнтів можуть використовувати її для написання більш дружніх відповідей на запити клієнтів.

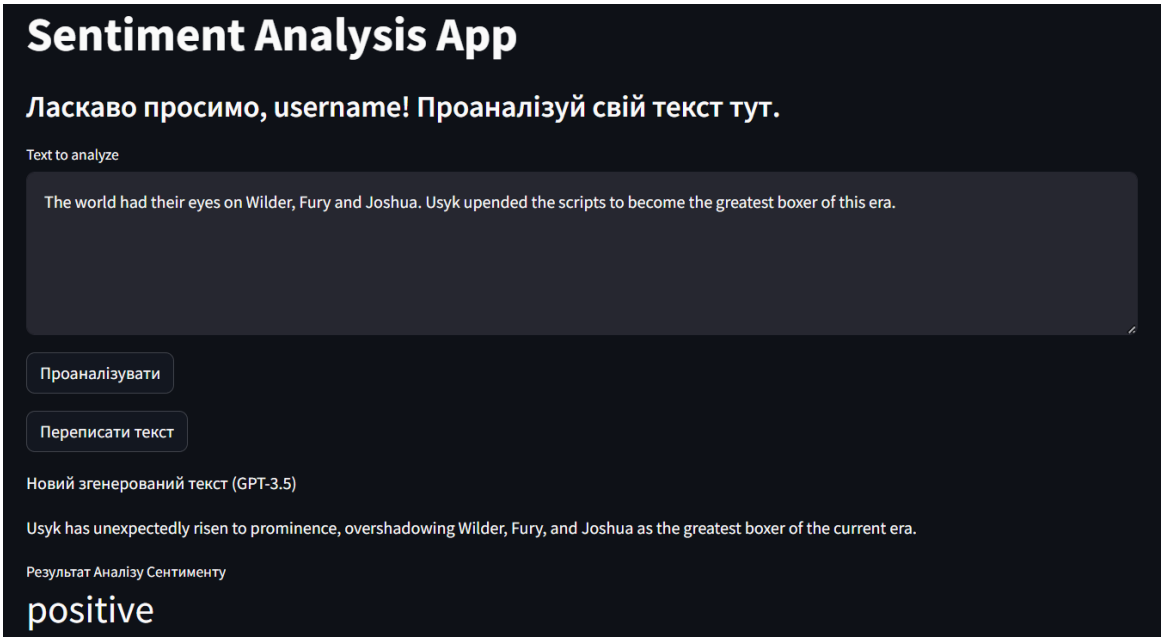

Рисунок 5.6 – Результат переписування тексту на інший сентимент

На сторінці історії аналізів (рисунок 5.7) користувач може переглядати історію всіх проведених аналізів текстів.

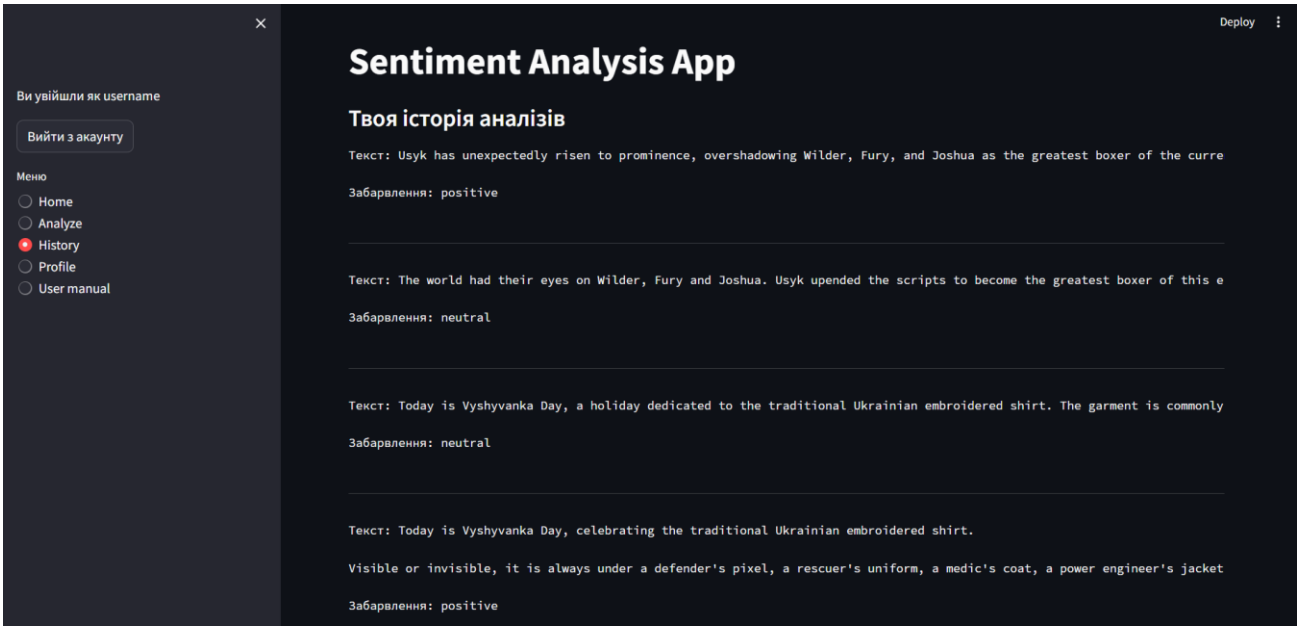

Рисунок 5.7 – Історія аналізів і переписувань текстів

Відображаються оригінальний текст, переписаний текст (якщо є), результати аналізу сентименту (рисунок 5.7).

На сторінці профілю користувача (рисунок 5.8) відображається інформація про користувача, така як ім'я користувача та електронна пошта.

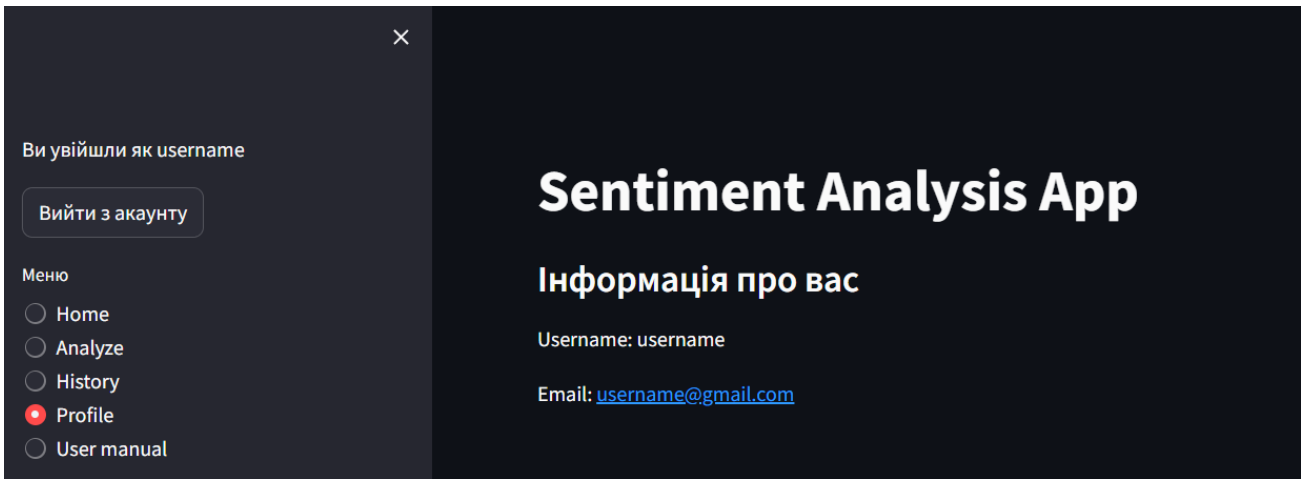

Рисунок 5.8 – Профіль користувача

На сторінці інструкцій (рисунок 5.9) містяться докладні інструкції про те, як користуватися додатком.

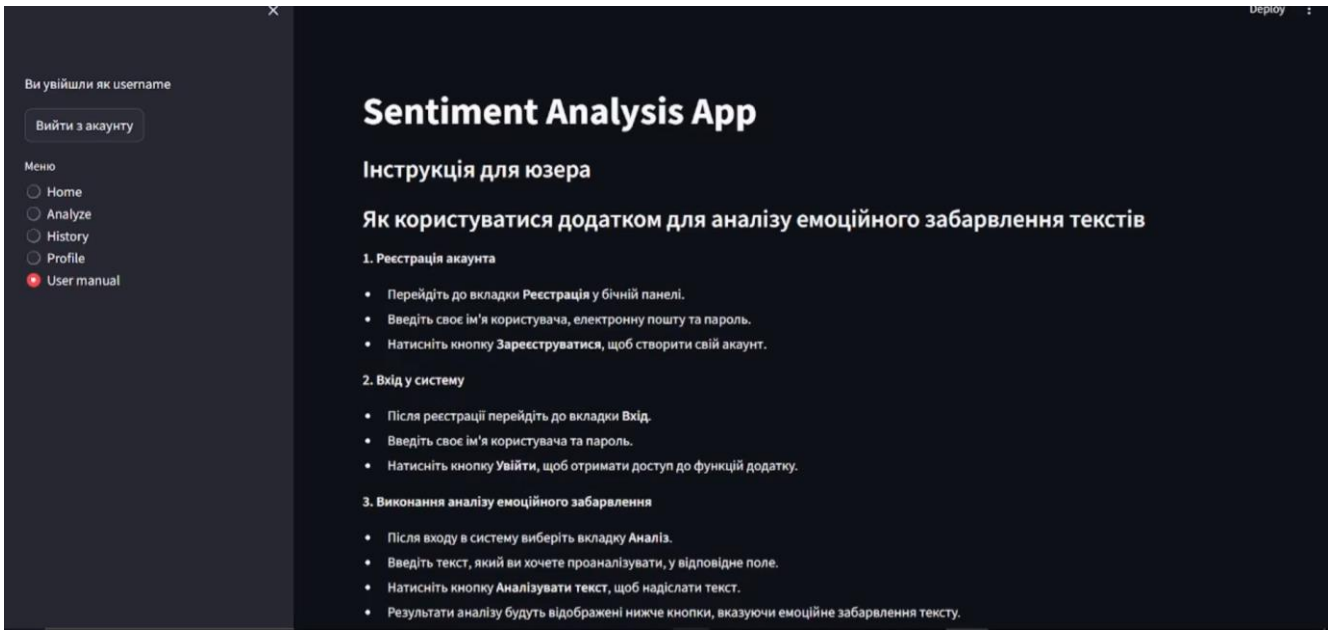

Рисунок 5.9 – Інструкція користування додатком

Інструкція для користувача розписана детально та зрозуміло.

ВИСНОВКИ

У ході даної дипломної роботи було створено онлайн-платформу для аналізу тексту з можливістю переписання на іншу емоційну спрямованість. Основною метою проекту було розробити програму, що дозволить користувачам аналізувати емоційне забарвлення текстів та переписувати їх у відповідності до потреб, включаючи заміну негативних висловлювань на позитивні або нейтральні і навпаки. Результатом роботи стала інтерактивна веб-система, яка дає можливості користувачам: реєструватися та входити в систему, проводити аналіз емоційного наповнення текстових матеріалів, переписуючи їх на інший сентимент, переглядати історію аналітики та мати доступ до особистого профілю.

Система пройшла успішне тестування та продемонструвала свою ефективність у виконанні різноманітних завдань. Вона може бути корисною для широкого кола користувачів, таких як маркетологи, фахівці з обслуговування клієнтів, контент-менеджери та інші професіонали, які потребують інструменти для аналізу та зміни тональності тексту.

Веб-система, що була створена у рамках даної роботи, пропонує значні переваги порівняно з наявними рішеннями завдяки застосуванню передових моделей машинного навчання та обробки природної мови. Це дозволяє отримувати точні й актуальні результати емоційного аналізу тексту, що у свою чергу сприяє кращому усвідомленю текстових даних і поверненню якості комунікаціїї.

У майбутньому можливим напрямком розвитку системи може стати інтеграція додаткових мовних моделей, розширення функціоналу для підтримки ширшого спектру емоційних відтінків та поліпшення користувацького інтерфейсу для забезпечення більшої зручності та ефективності використання. Розвиток системи може передбачати введення нових можливостей, таких як підтримка мультимовного аналізу, розширене налаштування аналізу для різних галузей та контекстів, а також інтеграцію з іншими інструментами та платформами для багатофункціональності та оптимального використання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Moppicon H. Streamlit for Data Science. – Manning Publications, 2022. – 384 с. ISBN 978-1617297995.

2. Маккінні Уес. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. – 2-е видання. – O'Reilly Media, 2017. – 544 с. ISBN 978-1491957660.

3. Карлсон Д., Андерсон П. Redis in Action. – Manning Publications, 2013. – 288 с. ISBN 978-1617290859.

4. Свейгарт А. Automate the Boring Stuff with Python: Practical Programming for Total Beginners. – 2-е видання. – No Starch Press, 2019. – 592 с. ISBN 978-1593279929.

5. Джурафскі Д., Мартін Дж. Х. Speech and Language Processing. Sentiment Analysis and Opinion Mining. – 3-е видання. – Pearson, 2019. – 1024 с. ISBN 978-0131873216.

6. Лю Бінг. Sentiment Analysis and Opinion Mining. – Morgan & Claypool Publishers, 2012. – 167 с. ISBN 978-1608458844.

7. Берслей Д., Джонс Б., Майерс Д. Е. Python Cookbook, 3rd Edition: Recipes for Mastering Python 3. – O'Reilly Media, 2013. – 706 с. ISBN 978-1449340377.

8. Льочек Ф. Pro Git. – Apress, 2014. – 456 с. ISBN 978-1484200773.

9. Бейтс М., Дойл Д. Clean Code: A Handbook of Agile Software Craftsmanship. – Prentice Hall, 2008. – 464 с. ISBN 978-0132350884.

10. Джерард А. Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2. – 3-е видання. – Packt Publishing, 2019. – 770 с. ISBN 978-1789955750.

11. Бредлі О., Паттон Дж. Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems. – O'Reilly Media, 2017. – 600 с. ISBN 978-1449373320.

12. Седжвік С., Вейн К. Algorithms, 4th Edition. – Addison-Wesley Professional, 2011. – 976 с. ISBN 978-0321573513.

13. Бішоп Е. Pattern Recognition and Machine Learning. – Springer, 2006. – 738 с. ISBN 978-0387310732.

14. Муллер Г., Гвідо Р. Introduction to Machine Learning with Python: A Guide for Data Scientists. – O'Reilly Media, 2016. – 400 с. ISBN 978-1449369415.

15. Расмуссен К. Gaussian Processes for Machine Learning. – MIT Press, 2006. – 248 с. ISBN 978-0262182539.

16. Гудфеллоу І., Бенджіо Й., Курвіль А. Deep Learning. – MIT Press, 2016. – 800 с. ISBN 978-0262035613.

17. Шапіро Р., Мацумото С. Understanding Machine Learning: From Theory to Algorithms. – Cambridge University Press, 2014. – 560 с. ISBN 978-1107057135.

18. Капур Д. Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. – 2-е видання. – O'Reilly Media, 2019. – 850 с. ISBN 978-1492032649.

ДОДАТОК А

ТЕКСТ ПРОГРАМНОГО МОДУЛЯ

«РЕАЛІЗАЦІЯ АНАЛІЗУ ТА ПЕРЕПИСУВАННЯ ТОНАЛЬНОСТІ» ТЕКСТУ

УКР.НТУУ”КПІ ім. Ігоря Сікорського”_ІАТЕ_ЦТЕ_ТР-02

Аркушів 3

Київ-2024

Програмний код

Main.py

```
import streamlit as st
import redis
from dotenv import load_dotenv
import os

try:
    from transformers import RobertaTokenizer, RobertaForSequenceClassification, RobertaConfig
except:
    pass
import torch
import json
import openai

model_path = 'modelRBRT.h5'
config = RobertaConfig.from_pretrained("roberta-base", num_labels=3)
model = RobertaForSequenceClassification(config)
model.load_state_dict(torch.load(model_path))
tokenizer = RobertaTokenizer.from_pretrained('roberta-base')

load_dotenv()

client = openai.OpenAI(
    api_key=os.getenv('OPENAI_API_KEY'),
)
```

```
r = redis.Redis(host='redis', port=6379, db=0, decode_responses=True)  # host='localhost', якщо запускаємо не через Docker!!!

def analyze_sentiment(text):
    st.session_state['sentiment'] = None
    inputs = tokenizer(text, return_tensors="pt", max_length=512, truncation=True, padding=True)
    outputs = model(**inputs)
    prediction = torch.nn.functional.softmax(outputs.logits, dim=-1)
    sentiment = ['negative', 'neutral', 'positive'][torch.argmax(prediction)]
    scores = prediction.tolist()[0]
    return sentiment, scores

def store_analysis(username, text, sentiment, scores, rewritten_text=None):
    analysis_data = {
        "text": text,
        "sentiment": sentiment,
        "scores": scores,
        "rewritten_text": rewritten_text
    }
    r.lpush(f"{username}:analysis_history", json.dumps(analysis_data))

def generate_text_rewrite(text, sentiment):
    prompt = f"Rewrite this text to be the most positive, make this text very positive : {text}" if sentiment == "negative" else f"Rewrite this text to be the most neutral or negative: {text}"
    message = {
        'role': 'user',
        'content': prompt
    }
```

```
response = client.chat.completions.create(
    model="gpt-3.5-turbo-0125",
    messages=[message],
    max_tokens=100
)
return response.choices[0].message.content.strip()
```