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)

iOS-додаток «Sentiment Analyzer»
з використанням Core ML та SwiftUI
(тема)

Виконав:
здобувач другого року навчання,
групи ДСм-24-1

Бондарєва В.О.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки

(код і повна назва спеціальності)
Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма Науки про дані (Data Science)

(повна назва спеціалізації)
Керівник проф. Бодянський Є.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Лариса ЧАЛА
(підпис) (прізвище, ініціали)

2025 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-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Науки про дані (Data Science)
(повна назва)

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

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Бондарєвій Валерії Олександрівні
(прізвище, ім'я, по батькові)

1. Тема роботи iOS-додаток «Sentiment Analyzer» з використанням Core ML та SwiftUI

затверджена наказом університету від 24 листопада 20 25 р. № 1057Ст
2. Термін подання студентом роботи до екзаменаційної комісії 16 грудня 20 25 р.
3. Вихідні дані до роботи офіційні документації до SwiftUI, Core ML, Natural Language Framework, Xcode, iOS Human Interface Guidelines та Create ML, а також методичні матеріали кафедри штучного інтелекту щодо структури та вимог до кваліфікаційних робіт.

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та постановка задачі
2) Теоретичні дослідження
3) Проектування системи
4) Програмна реалізація

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	24.11.2025	виконано
2	Визначення проблеми	25.11.2025	виконано
3	Аналіз конкурентних рішень	27.11.2025	виконано
4	Проектування компонентів системи	28.11.2025	виконано
5	Реалізація додатку	01.12.2025	виконано
6	Оформлення пояснювальної записки	06.12.2025	виконано
7	Захист	16.12.2025	виконано

Дата видачі завдання 24 листопада 20 25 р.

Здобувач _____
(підпис)

Керівник роботи _____ проф. Бодянський Є.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 100 с., 30 рис., 1 дод., 20 джерел.

АНАЛІЗ ТОНАЛЬНОСТІ, МАШИННЕ НАВЧАННЯ, МОБІЛЬНИЙ ДОДАТОК, МОДЕЛЬ ТЕКСТОВОЇ КЛАСИФІКАЦІЇ, ШТУЧНИЙ ІНТЕЛЕКТ, CORE ML, CREATE ML, NATURAL LANGUAGE PROCESSING, SWIFT, SWIFTUI.

Об’єктом дослідження є процес аналізу тональності текстових даних у мобільних інтелектуальних системах.

Предметом дослідження є методи та алгоритми машинного навчання для класифікації емоційного забарвлення тексту в iOS-застосунку.

Метою роботи є розробка мобільного застосунку «Sentiment Analyzer» з використанням технологій Core ML та SwiftUI для визначення тональності введених користувачем текстових фрагментів у реальному часі.

У роботі застосовано методи машинного навчання, алгоритми обробки природної мови та інструменти програмної розробки для платформи iOS.

Результатом роботи стало створення функціонального мобільного застосунку з локальною ML-моделлю, здатного класифікувати текст за тональністю, а також підтвердження ефективності використання Core ML для виконання NLP-задач у мобільному середовищі.

Результати роботи можуть бути використані для розробки освітніх, аналітичних та комунікаційних застосунків, а також як основа для подальших досліджень у сфері мобільних NLP-систем

Створений застосунок може бути інтегрований у сервіси, що потребують аналізу настроїв користувачів або оцінки емоційності текстового контенту.

ABSTRACT

Master’s thesis contains: 100 pp., 30 fig., 1 ann., 20 references.

ARTIFICIAL INTELLIGENCE, CORE ML, CREATE ML, MACHINE LEARNING, MOBILE APPLICATION, NATURAL LANGUAGE PROCESSING, SENTIMENT ANALYSIS, SWIFT, SWIFTUI, TEXT CLASSIFICATION MODEL.

The object of the study is the process of sentiment analysis of textual data in mobile intelligent systems.

The subject of the study is the machine learning methods and algorithms used for classifying the emotional tone of text within an iOS application.

The purpose of the work is to develop a mobile application titled “Sentiment Analyzer” using Core ML and SwiftUI technologies to determine the sentiment of user-entered text fragments in real time.

The study employs machine learning methods, natural language processing algorithms, and software development tools for the iOS platform.

The result of the work is a fully functional mobile application equipped with a local ML model capable of classifying text by sentiment, as well as a demonstrated confirmation of the effectiveness of Core ML for performing NLP tasks in a mobile environment.

The results of the work can be used for the development of educational, analytical, and communication applications, as well as a foundation for further research in the field of mobile NLP systems.

The created application can be integrated into services that require user sentiment assessment or evaluation of emotional characteristics of textual content.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AI – Artificial Intelligence – штучний інтелект;

API – Application Programming Interface – прикладний програмний інтерфейс;

CPU – Central Processing Unit – центральний процесор;

Core ML – Core Machine Learning Framework – фреймворк для виконання ML-моделей на пристроях Apple;

Create ML – Create Machine Learning – інструмент Apple для навчання моделей;

DL – Deep Learning – глибинне навчання;

GPU – Graphics Processing Unit – графічний процесор;

iOS – iPhone Operating System – операційна система мобільних пристроїв Apple;

ML – Machine Learning – машинне навчання;

MLM – Machine Learning Model – модель машинного навчання;

MVVM – Model–View–ViewModel – архітектурний шаблон;

NLL – Natural Language Learning – навчання природної мови;

NLP – Natural Language Processing – обробка природної мови;

NLP Model – Natural Language Processing Model – модель обробки природної мови;

NLSentiment – Natural Language Sentiment Analysis API – API для аналізу тональності в iOS;

PDF – Portable Document Format – формат електронних документів;

UI – User Interface – інтерфейс користувача;

UX – User Experience – користувацький досвід;

WWDC – Worldwide Developers Conference – щорічна конференція Apple для розробників.

ВСТУП

Стрімкий розвиток цифрових технологій та постійне зростання обсягів інформації у глобальних мережах призвели до суттєвого збільшення ролі текстових даних у повсякденній комунікації. Користувачі соціальних мереж, месенджерів, форумів, електронних сервісів та мобільних застосунків щоденно генерують величезні масиви текстових повідомлень, що містять широкий спектр емоційних, оцінкових і суб'єктивних характеристик. У таких умовах значна увага приділяється інструментам автоматичного аналізу тексту, які дозволяють ефективно інтерпретувати настрої користувача, виявляти емоційний контекст висловлювань та формувати відповідні реакції систем.

Сучасний стан досліджень у цій сфері демонструє активний перехід від класичних статистичних методів до моделей глибинного навчання, які забезпечують точнішу та стабільнішу інтерпретацію текстових даних. Методи NLP значно покращили якість аналізу тексту та дозволили реалізувати алгоритми, здатні працювати в режимі реального часу. Попри значні успіхи у цій галузі, залишається проблема адаптації складних моделей для мобільних платформ, де важливими є обмеження ресурсів, автономність, приватність та швидкість обчислень. У зв'язку з цим особливо перспективним напрямом є дослідження й розробка рішень, здатних працювати без підключення до мережі та забезпечувати локальне опрацювання даних користувача.

Значна актуальність теми зумовлена тим, що багато користувацьких задач потребують швидкої та приватної обробки тексту без передачі даних на сервери. У сучасних соціальних та комунікаційних платформах увага приділяється захисту персональної інформації, а алгоритми аналізу тональності можуть використовуватися для покращення взаємодії з користувачами, підвищення якості сервісів та адаптації контенту.

Ціль роботи полягає у розробці мобільного застосунку для аналізу тональності тексту на платформі iOS з використанням моделей машинного навчання, виконуваних локально на пристрої. Реалізація такої системи вимагає проведення комплексного дослідження методів NLP, вибору оптимального підходу до підготовки даних, створення та навчання ML-моделі, а також інтеграції її у програмний продукт. Важливою складовою є забезпечення стабільної роботи застосунку, швидкодії та інтуїтивно зрозумілого інтерфейсу взаємодії з користувачем. Розроблений застосунок має демонструвати можливості побудови інтелектуальних сервісів у мобільному середовищі та підтверджувати ефективність локального аналізу даних у контексті сучасних вимог приватності.

Можливі сфери застосування розробленого рішення є надзвичайно широкими. Застосунок може використовуватися для аналізу емоційного забарвлення текстів у сфері освіти, маркетингу, соціальних мереж, операторської діяльності та цифрової комунікації. Він може бути корисним для фахівців, які працюють з великою кількістю текстових повідомлень, студентів та дослідників, які вивчають NLP, а також для користувачів, що прагнуть швидко оцінити настрій написаного тексту. Система також може слугувати основою для створення складніших інтелектуальних модулів, інтегрованих у персональні помічники, чат-боти та інші програмні продукти, де важливою є інтерпретація емоційної складової мовлення.

Таким чином, дослідження та реалізація мобільного застосунку для аналізу тональності тексту за допомогою технологій Core ML та SwiftUI є актуальним завданням, яке поєднує сучасні підходи до обробки природної мови, машинного навчання та розробки інтерактивних мобільних сервісів. Виконання такої роботи сприяє розвитку компетенцій у сфері штучного інтелекту, демонструє можливості інтеграції ML у мобільне середовище та створює практичну цінність для широкого кола користувачів та застосувань.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Теоретичні основи аналізу тональності тексту

Аналіз тональності тексту є важливим напрямом у сфері обробки природної мови, оскільки дозволяє автоматизовано визначати емоційне забарвлення повідомлень, коментарів або інших текстових даних. Такий підхід став особливо актуальним у зв’язку зі зростанням обсягів неструктурованої текстової інформації, що генерується користувачами у цифровому середовищі. Завдяки методам машинного навчання сучасні системи здатні з високою точністю класифікувати текст як позитивний, нейтральний чи негативний, забезпечуючи можливість оперативного аналізу великих масивів даних.

Розвиток технологій у галузі штучного інтелекту сприяв переходу від простих лексичних правил до складніших моделей, здатних враховувати контекст та структурні особливості тексту. Глибинні нейронні мережі, трансформери та інші сучасні моделі значно підвищили якість автоматичної класифікації, зробивши можливим застосування sentiment analysis у різноманітних інформаційних системах. Цей напрям активно використовується в маркетинговій аналітиці, автоматизованій модерації контенту та інтелектуальних сервісах взаємодії з користувачами.

У контексті мобільних застосунків аналіз тональності став особливо перспективним завдяки розвитку інструментів локального машинного навчання. Платформа iOS надає можливості для ефективного використання моделей безпосередньо на пристрої, що сприяє підвищенню продуктивності та забезпечує конфіденційність даних користувача. Використання фреймворків на кшталт Core ML дозволяє переносити алгоритми sentiment analysis у мобільну екосистему, роблячи такі системи доступними, швидкими та енергоефективними.

1.1.1 Поняття та роль аналізу тональності тексту в сучасних інформаційних системах

Аналіз тональності тексту розглядається як процес автоматичного визначення емоційного забарвлення висловлювання, що може бути позитивним, нейтральним або негативним. Такий підхід дає змогу інтерпретувати зміст коротких та довільних текстових повідомлень, що особливо актуально у цифровому середовищі. Приклади типових висловлювань із різною тональністю ілюструють різноманітність форм вираження емоцій у текстах, які може бути проаналізовано алгоритмами машинного навчання (рисунок 1.1).

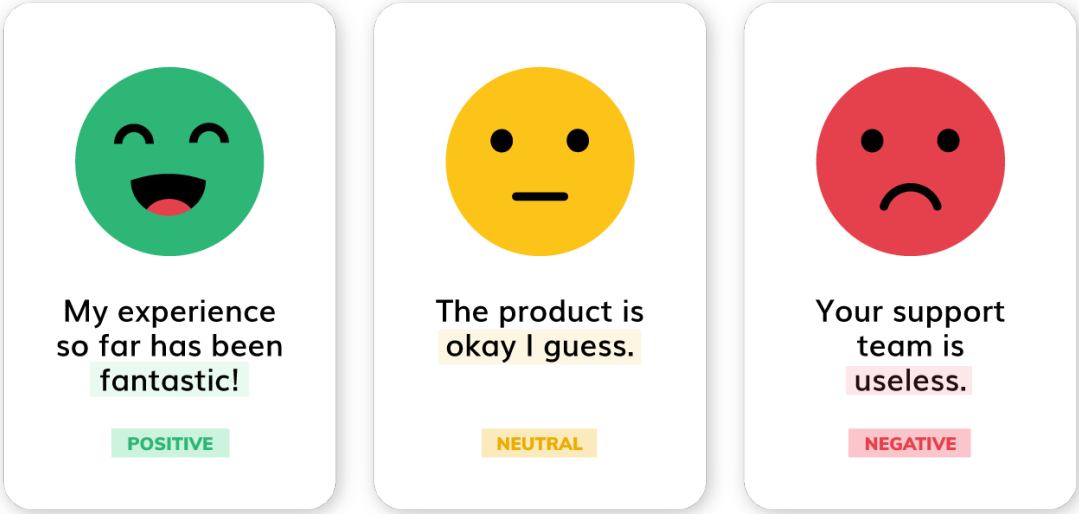

Рисунок 1.1 – Приклади роботи аналізу тональності тексту

Важливою перевагою аналізу тональності є можливість швидкого оброблення великих обсягів користувацьких даних, що робить його незамінним інструментом в автоматизованих системах моніторингу. Завдяки розвитку технологій штучного інтелекту та розповсюдженню мобільних пристроїв такі методи стають доступними більш широкому колу користувачів.

У сучасних інформаційних системах sentiment analysis використовується для покращення взаємодії між користувачем і цифровими сервісами. Соціальні мережі, чат-боти, системи підтримки клієнтів та маркетингові платформи активно впроваджують такі механізми з метою автоматизованого реагування на відгуки, повідомлення або коментарі. Для забезпечення коректності визначення емоційного забарвлення необхідно застосовувати структурований процес (рисунок 1.2), що включає підготовку даних, виділення ознак та побудову відповідної моделі.

Рисунок 1.2 – Етапи процесу аналізу тональності тексту

У результаті аналіз тональності тексту забезпечує підвищення якості сервісів, формуючи гнучкі механізми реагування на настрій та емоційний контекст текстової інформації.

1.1.2 Технологічні основи обробки природної мови

Технологічні основи обробки природної мови охоплюють широкий спектр методів, що дозволяють структурувати, аналізувати та інтерпретувати текстову інформацію. Класичні підходи NLP базуються на використанні правил, статистичних моделей та інженерії ознак, що передбачає попереднє очищення тексту, токенізацію, лематизацію та

подальше формування векторних представлень для моделювання. Структурна схема таких класичних методів узагальнена у верхній частині зображення (рисунок 1.3), де відображено етапи попередньої обробки та побудови моделей.

Поступовий розвиток штучного інтелекту сприяв появі глибинних нейронних мереж, здатних автоматично формувати складні представлення тексту. На відміну від класичних методів, такі моделі не потребують ручного створення ознак, оскільки самостійно навчаються виявляти закономірності у великих корпусах даних.

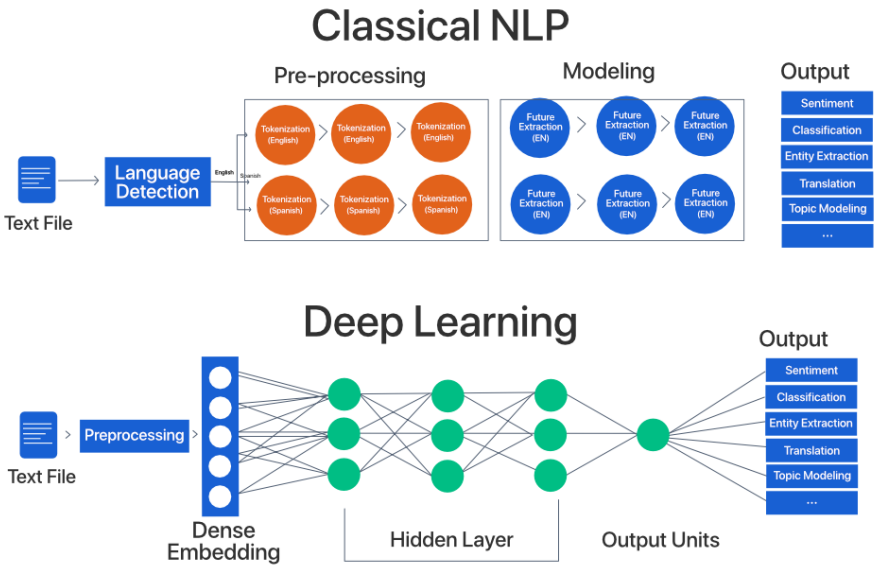

Рисунок 1.3 – Порівняння класичних методів NLP та глибинного навчання у задачах опрацювання тексту

У нижній частині зображення (рисунок 1.3) подано типову структуру глибинної моделі, що включає етап векторизації, приховані шари та блоки вихідної класифікації. Завдяки цим підходам стало можливим більш точно враховувати контекст та семантичні зв’язки між словами.

Особливу роль глибинне навчання відіграє у роботі з короткими повідомленнями, характерними для соціальних мереж та мобільних

застосунків. Класичні методи часто виявляються недостатньо ефективними через брак контексту та неоднозначність формулювань, тоді як моделі на основі нейронних мереж здатні краще інтерпретувати тональність навіть у лаконічних текстах.

1.1.3 Використання машинного навчання у iOS-додатках

Використання машинного навчання у мобільних iOS-додатках стало важливим напрямом розвитку сучасних цифрових сервісів, оскільки дозволяє забезпечувати інтелектуальні функції безпосередньо на пристрої користувача. Моделі аналізу тональності допомагають автоматизовано оцінювати настрій текстових повідомлень, відгуків або коментарів, що є корисним для підвищення якості взаємодії з продуктом (рисунок 1.4).

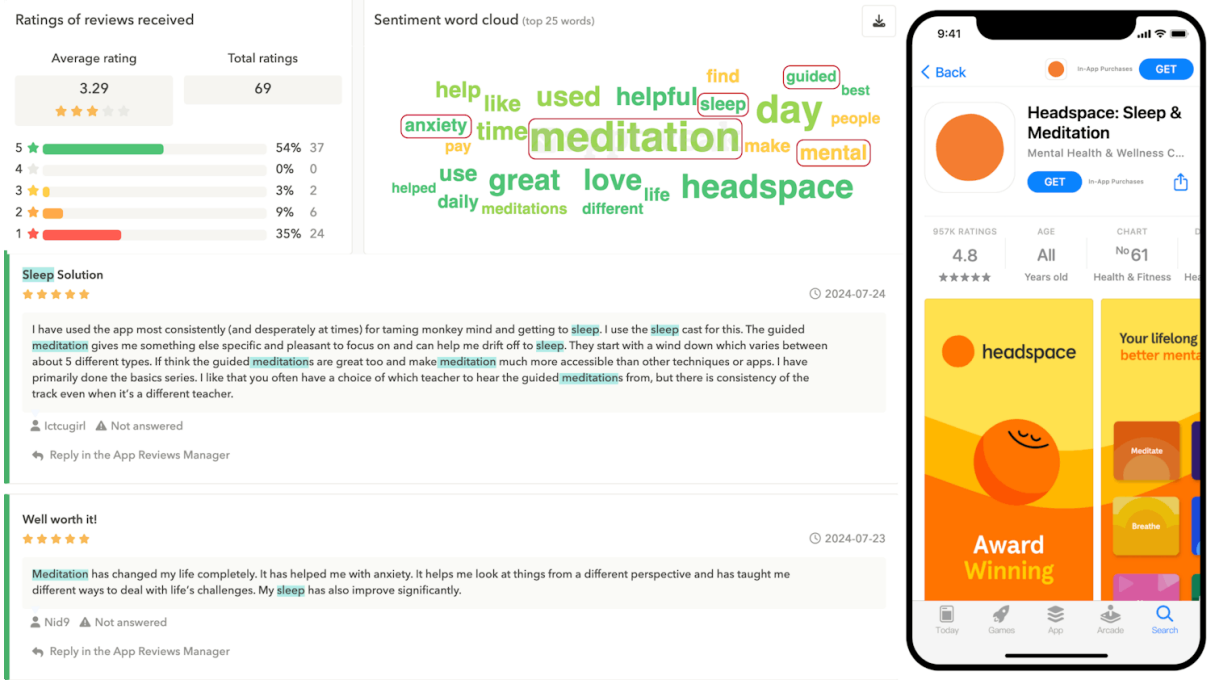

Рисунок 1.4 – Приклад застосування аналізу тональності користувацьких відгуків у мобільному додатку для покращення сервісу

Важливою перевагою локального машинного навчання є забезпечення швидкодії та збереження конфіденційності персональних даних. Обробка тексту на пристрої усуває потребу у відправленні інформації на сторонні сервери, що суттєво зменшує затримку та підвищує безпеку. Це особливо важливо у випадках, коли користувачі залишають особисті коментарі або чутливі відгуки, які не повинні покидати межі мобільного пристрою. Візуалізація результатів, аналітичні панелі та інтерактивні елементи інтерфейсу (рисунок 1.4), демонструють потенціал поєднання машинного навчання та мобільних технологій.

iOS-екосистема надає розробникам інструменти для ефективної інтеграції моделей машинного навчання у застосунки, що дозволяє створювати адаптивні та інтелектуальні функції без складної інфраструктури. Використання таких підходів сприяє підвищенню точності аналізу, оптимізації взаємодії з користувачем та формуванню персоналізованого досвіду. Завдяки поєднанню машинного навчання та сучасних інтерфейсних рішень розробники можуть забезпечити високу якість оброблення текстових даних та зрозуміле подання результатів навіть у мобільних умовах.

1.2 Сучасні технології та інструменти для реалізації текстової класифікації на iOS

Розвиток мобільних технологій створив умови для широкого впровадження алгоритмів машинного навчання безпосередньо у мобільні застосунки. Для задач аналізу тональності тексту це має особливе значення, оскільки користувачі очікують швидкої обробки введених даних та отримання результату в реальному часі. Завдяки спеціалізованим фреймворкам та оптимізованим моделям стало можливим досягати високої продуктивності навіть на пристроях із обмеженими ресурсами.

iOS-екосистема підтримує вбудовані інструменти, що полегшують розробку застосунків з функціями штучного інтелекту. Такі технології забезпечують ефективну інтеграцію моделей, можливість їхнього автоматичного перетворення у формат, придатний для мобільної роботи, а також надають механізми для виконання inference локально. Це відкриває можливість реалізувати повний цикл аналізу тексту у межах застосунку без залучення зовнішніх серверів.

Сучасні засоби розробки також зосереджені на спрощенні взаємодії між інтерфейсом користувача та моделлю машинного навчання. Завдяки декларативному підходу до побудови UI та продуманим механізмам прив’язки даних розробники можуть створювати зручні та інтуїтивні інтерфейси, які реагують на результати класифікації. У комплексі ці технології формують основу для створення ефективних та гнучких систем аналізу тональності тексту на платформі iOS.

1.2.1 Core ML як фреймворк для інтеграції ML-моделей

Core ML є ключовою технологією платформи iOS, яка забезпечує інтеграцію моделей машинного навчання безпосередньо у мобільні застосунки. Завдяки цій інфраструктурі розробники можуть використовувати моделі різних типів, зокрема нейронні мережі, класифікатори тексту та моделі для аналізу зображень, не залучаючи серверні обчислення. Можливості Core ML (рисунок 1.5) відображені нижче, де підкреслено його застосування у задачах класифікації, розпізнавання та рекомендаційних системах.

Фреймворк забезпечує оптимізоване виконання моделей на пристрої, що дозволяє досягати високої швидкодії навіть на компактних мобільних процесорах. Таке рішення робить можливим виконання складних алгоритмів, включно з аналізом тональності тексту, розпізнаванням мовлення або обробкою мультимедійних даних у реальному часі.

Принципова ідея Core ML полягає у тому, що обробка відбувається локально, що зменшує затримку, покращує продуктивність і забезпечує конфіденційність користувацьких даних, що є важливою перевагою у контексті сучасних мобільних сервісів.

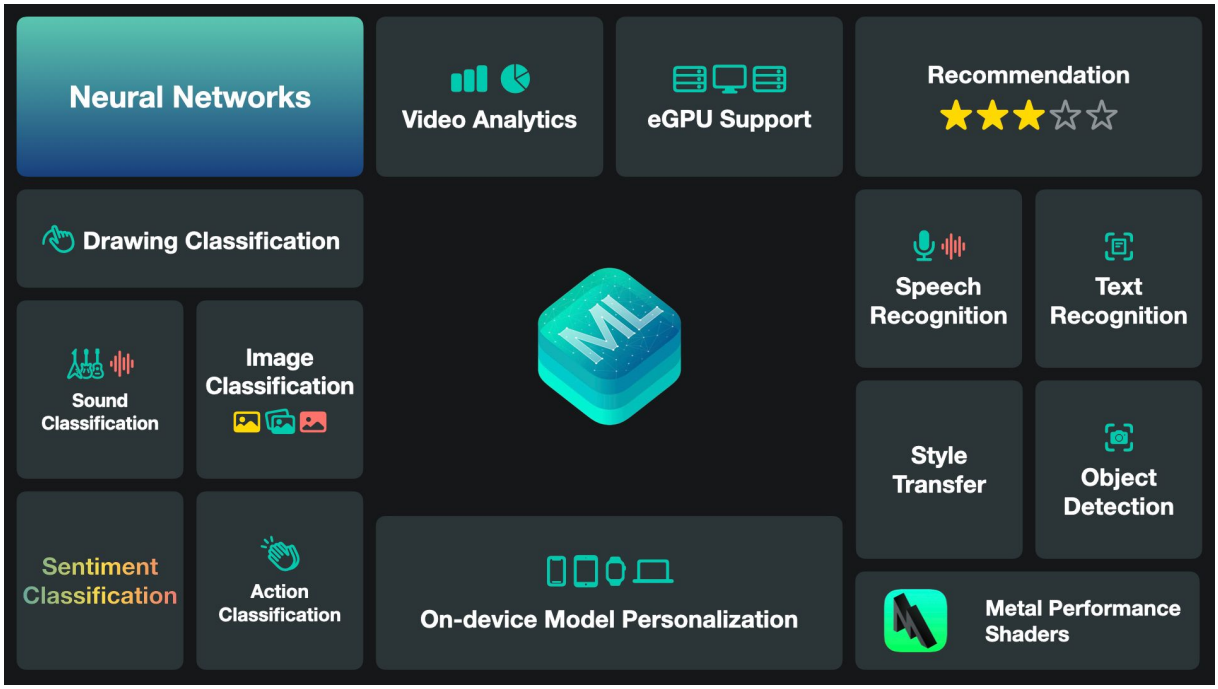

Рисунок 1.5 – Основні можливості Core ML

Ще однією важливою характеристикою Core ML є підтримка автоматичної оптимізації моделей для мобільного середовища. Під час конвертації модель адаптується до можливостей пристрою, що спрощує її подальшу інтеграцію у застосунок. Це дозволяє розробникам зосередитися на логіці роботи системи, не витрачаючи час на ручне налаштування продуктивності або архітектури обчислень. Завдяки цьому Core ML стає універсальним інструментом, що забезпечує просту та ефективну інтеграцію машинного навчання у застосунки для iOS, iPadOS та інших платформ Apple.

1.2.2 Create ML як інструмент для створення моделі тональності

Create ML є інструментом, призначеним для спрощення процесу побудови та тренування моделей машинного навчання у середовищі Apple. Він надає можливість створювати моделі різних типів, включно з класифікаторами тексту, використовуючи інтуїтивний інтерфейс та роботу з наборами даних у режимі drag and drop. На зображені нижче (рисунок 1.6) відображено ключові можливості Create ML, серед яких підтримка кількох датасетів, перегляд проміжних результатів та простий експорт моделей у формат, сумісний із Xcode. Такий підхід дозволяє швидко переходити від підготовки даних до отримання повноцінної моделі для мобільного застосунку.

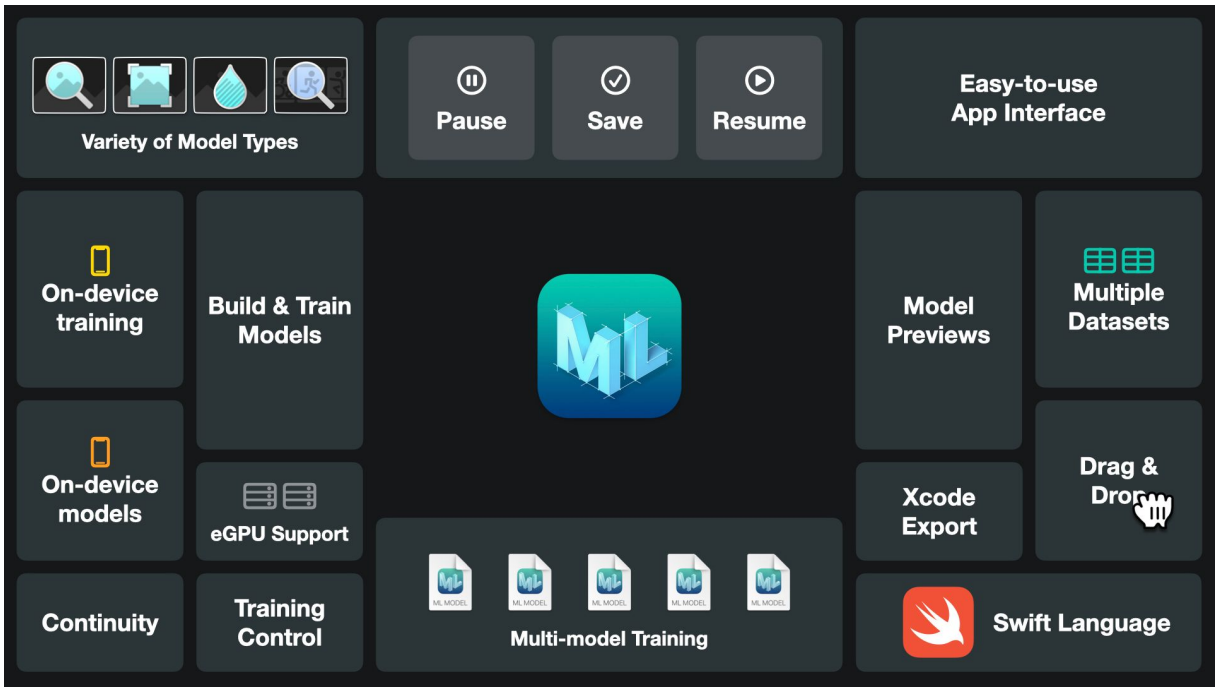

Рисунок 1.6 – Основні можливості Create ML

Оскільки Create ML оптимізовано для екосистеми Apple, результатом тренування є модель, яка автоматично адаптована для ефективної роботи на iOS, iPadOS та macOS. Це дає змогу розробникам концентруватися на логіці

застосунку замість технічних аспектів побудови моделей. Крім того, підтримка on-device тренування та зручні механізми управління процесом навчання, такі як пауза або продовження, роблять інструмент універсальним у контексті задач аналізу тональності тексту, що значно спрощує розгортання моделей у мобільному середовищі (рисунок 1.6).

1.2.3 Технологічні можливості SwiftUI для інтеграції ML-моделі

SwiftUI надає сучасний декларативний підхід до побудови інтерфейсу, що значно спрощує взаємодію між користувацьким інтерфейсом та логікою машинного навчання. Завдяки реактивній природі SwiftUI зміни, які надходять від ML-моделі, миттєво відображаються на екрані, що робить процес аналізу тексту в режимі реального часу плавним та зручним. Такий підхід дозволяє створювати інтерфейси, у яких користувач вводить текст, а результат класифікації автоматично оновлюється без додаткових дій з його боку.

Важливим аспектом SwiftUI є тісна інтеграція зі структурою даних та можливість використання патерну MVVM, що полегшує відокремлення логіки роботи ML-моделі від інтерфейсу. ViewModel може виконувати виклики до Core ML і передавати результати назад до View, що забезпечує зрозумілий та підтримуваний код. Це дозволяє легко масштабувати застосунок, додавати нові функції та удосконалювати точність аналізу тональності без значних змін у користувацькому інтерфейсі.

Крім того, SwiftUI підтримує широкий набір анімацій та засобів візуалізації, які дають змогу створювати інтерфейси з інформативним і приємним поданням результатів роботи ML-моделі. Розробник може легко додати кольорові індикатори, графічні маркери або інші елементи, що відображають позитивну, нейтральну чи негативну тональність тексту.

1.3 Аналіз потреб та вимог користувачів

Аналіз потреб користувачів є важливою складовою під час проєктування мобільних застосунків, що використовують машинне навчання. Розуміння очікувань та поведінкових сценаріїв дозволяє сформувати функціональність, яка буде максимально корисною у реальному використанні. У випадку застосунків для аналізу тональності тексту особливу увагу приділяють простоті взаємодії, швидкості отримання результатів та зрозумілому поданню інформації. Користувачі очікують, що система коректно інтерпретуватиме їхні повідомлення та надаватиме результати без затримок.

У контексті мобільних систем важливими є як функціональні, так і нефункціональні вимоги. Функціональні визначають, які саме можливості має забезпечувати застосунок, наприклад введення тексту, аналіз тональності, отримання миттєвого результату чи формування допоміжних рекомендацій.

Нефункціональні вимоги стосуються швидкодії, стабільності, приватності та зручності роботи. Для систем з аналізом тексту особливо важливою є обробка даних на пристрої, адже це гарантує конфіденційність та підвищує рівень довіри користувача.

Потреби користувачів також залежать від контексту використання. Одні застосунки орієнтовані на швидку оцінку емоційного забарвлення повідомлень, інші можуть використовуватися у навчальних або професійних цілях, наприклад у сфері маркетингу чи аналізу відгуків.

Тому важливо врахувати різні сценарії взаємодії, щоб забезпечити гнучкість і зручність роботи з системою. Комплексне вивчення потреб користувачів створює підґрунтя для подальшого формування ключових функцій майбутнього застосунку та визначення вимог до його архітектури.

1.3.1 Визначення цільової аудиторії

Цільова аудиторія мобільного застосунку для аналізу тональності тексту формується виходячи з потреб користувачів, які регулярно працюють із текстовою інформацією або взаємодіють у цифровому середовищі. До цієї групи належать користувачі, що прагнуть швидко оцінювати емоційний зміст повідомлень, відгуків чи коротких текстових фрагментів. Такий інструмент є корисним для тих, хто активно спілкується у соціальних мережах або працює з великими обсягами коментарів та хоче розуміти емоційний контекст тексту без додаткового аналізу.

Важливою частиною аудиторії є спеціалісти, які професійно застосовують аналіз текстів у своїй діяльності. Це можуть бути маркетологи, SMM-фахівці, контент-мейкери та аналітики, які працюють з користувацькими відгуками та потребують ефективних інструментів для швидкої обробки інформації. Для них мобільний застосунок з аналізом тональності може виступати допоміжним засобом під час формування стратегій комунікації, моніторингу реакцій аудиторії або оцінювання якості продуктів.

Окрему частину цільової аудиторії становлять студенти, дослідники та розробники, які цікавляться природною мовою, машинним навчанням та технологіями Apple. Вони можуть використовувати застосунок як навчальний інструмент для дослідження моделей класифікації тексту або тестування поведінки ML-моделей у реальних умовах. Таким чином, цільова аудиторія охоплює як звичайних користувачів, так і професійних спеціалістів.

1.3.2 Аналіз сценаріїв використання мобільного додатка

Сценарії використання мобільного застосунку для аналізу тональності визначаються тим, у яких ситуаціях користувачі звертаються до таких

інструментів. Одним із найпоширеніших сценаріїв є швидка оцінка емоційного забарвлення коротких текстових повідомлень, наприклад у соціальних мережах, месенджерах або електронних листах. Користувач може вставити текст у застосунок і миттєво отримати результат, що допомагає зрозуміти, як може бути сприйняте повідомлення іншими людьми або яку емоцію містить отриманий коментар.

Ще одним важливим сценарієм є попередній аналіз тексту перед його публікацією. У цьому випадку застосунок виконує роль інструмента самоконтролю або редактора настрою, допомагаючи користувачу перевірити, чи не містить повідомлення надто негативного, різкого або неоднозначного змісту. Такий підхід особливо корисний для блогерів, контент-мейкерів та маркетологів, які прагнуть підтримувати позитивний тон у комунікації з аудиторією.

Крім того, мобільний застосунок може використовуватися у навчальних або дослідницьких цілях. Наприклад, студенти чи спеціалісти у сфері NLP можуть експериментувати з різними фразами, аналізувати реакцію моделі та вивчати її поведінку у різних контекстах. Такі сценарії дозволяють не лише застосовувати функціональність для практичних задач, але й досліджувати роботу машинного навчання у реальному середовищі, що робить застосунок цінним інструментом як для повсякденних користувачів, так і для фахівців.

1.3.3 Функціональні потреби користувачів

Функціональні потреби користувачів визначають, які можливості має забезпечувати мобільний застосунок, щоб ефективно виконувати завдання з аналізу тональності тексту. Однією з ключових вимог є можливість швидкого введення тексту та миттєвого отримання результату класифікації. Користувачі очікують, що система буде працювати просто: достатньо ввести або вставити фрагмент тексту, після чого застосунок одразу

визначить його емоційне забарвлення. Це забезпечує зручність при щоденному використанні та дозволяє швидко аналізувати повідомлення у різних контекстах.

Ще однією важливою потребою є зрозуміла й інтуїтивна візуалізація результату. Більшість користувачів очікує, що результат буде подано у формі, яка одразу передає настрій тексту, наприклад через колірні індикатори, графічні маркери чи короткі пояснення. Це дає змогу оперативно інтерпретувати результат без додаткових пояснень. Такий підхід робить застосунок доступним для широкої аудиторії, включно з тими, хто не має поглиблених знань у галузі аналізу текстів.

Користувачі можуть також потребувати додаткових функцій, які розширюють можливості основного алгоритму. Серед таких функцій можна виділити перегляд історії аналізів, збереження попередніх результатів, можливість повторного використання текстів, а також адаптацію моделі під індивідуальні потреби. Хоча такі можливості не завжди є обов’язковими, вони підвищують зручність використання та роблять застосунок більш універсальним для повсякденних і професійних задач.

1.3.4 Нефункціональні вимоги та очікування

Нефункціональні вимоги охоплюють характеристики, що визначають якість роботи мобільного застосунку, його стабільність та комфорт користування. Однією з головних вимог є висока швидкодія, оскільки аналіз тональності має відбуватися без затримок. Користувач очікує, що застосунок миттєво реагуватиме на введення тексту, а модель зможе виконувати обчислення локально та з мінімальним впливом на ресурси пристрою. Це є важливим фактором, зважаючи на швидкий темп взаємодії у сучасному цифровому середовищі.

Також важливою вимогою є забезпечення приватності даних. Оскільки текстові повідомлення можуть містити особисту або чутливу

інформацію, користувачі очікують, що аналіз буде виконуватися виключно на пристрої без передачі даних на сторонні сервери. Такий принцип підвищує рівень довіри до застосунку і робить його безпечним для роботи з будь-яким типом інформації. Локальне виконання обчислень також сприяє стабільній роботі в умовах відсутності інтернету.

Серед інших очікувань користувачів можна виділити простоту інтерфейсу, легку навігацію та мінімальне енергоспоживання. Застосунок має бути інтуїтивно зрозумілим, без зайвих елементів, що ускладнюють використання. Він повинен адаптуватися до різних моделей пристроїв та підтримувати актуальні версії iOS. Поєднання цих вимог забезпечує комфортну взаємодію з системою та позитивний користувацький досвід, що є ключовим фактором у популярності мобільних застосунків для аналізу тексту.

1.4 Визначення ключових функцій та можливостей системи

Формування ключових функцій майбутнього мобільного застосунку є важливим етапом проєктування, оскільки саме вони визначають його практичну цінність та зручність використання. У випадку системи для аналізу тональності тексту основна увага спрямована на забезпечення швидкої взаємодії між користувачем та ML-моделлю, щоб результати класифікації були доступні в реальному часі. Такий підхід сприяє плавному робочому процесу та створює відчуття природності під час аналізу різних текстових фрагментів.

Важливу роль відіграє також архітектура системи, яка визначає взаємодію між інтерфейсом, логікою застосунку та компонентом машинного навчання. Від її грамотної побудови залежить стабільність роботи застосунку, можливість масштабування та подальшого розширення функціональності. Чітке розділення відповідальностей між компонентами

дає змогу легко вносити зміни, оновлювати ML-модель або вдосконалювати способи подання результатів користувачу.

Крім того, сучасні мобільні застосунки мають враховувати очікування користувачів щодо зручності інтерфейсу та надійності системи. Це стосується як візуального компонента, так і технічних характеристик, таких як швидкість обробки, приватність даних та оптимальне енергоспоживання. Поєднання цих аспектів формує комплексну функціональність, що дозволяє застосунку ефективно виконувати задачу аналізу тональності і забезпечувати позитивний досвід використання навіть у умовах обмежених ресурсів мобільного пристрою.

1.4.1 Функціональні можливості iOS-додатка

Функціональні можливості iOS-додатка формують основу його практичного застосування та визначають головні сценарії взаємодії користувача з системою. Ключовою функцією є введення тексту і миттєве отримання результату аналізу, що забезпечує швидку оцінку емоційного забарвлення повідомлень. Застосунок має підтримувати як короткі фрази, так і довші фрагменти, а результат класифікації повинен оновлюватися автоматично, без додаткових дій з боку користувача. Такий механізм дозволяє інтегрувати застосунок у повсякденну комунікацію.

Додаток також має надавати зрозумілу візуалізацію результатів, щоб користувач міг швидко інтерпретувати тональність тексту. Для цього можуть застосовуватися кольорові індикатори, умовні позначення або символічні елементи, які передають позитивний, нейтральний чи негативний настрій. Візуальна простота є важливою складовою, оскільки багато користувачів прагнуть отримувати інформацію без необхідності ознайомлення зі складними термінами чи технічними деталями.

За потреби система може включати додаткові функції, які розширюють її можливості. Це може бути історія попередніх аналізів,

можливість повторного використання текстів або персоналізація роботи моделі. Хоча такі можливості не є обов’язковими, вони покращують гнучкість застосунку і дозволяють адаптувати його до різних сфер діяльності, зокрема для професійних користувачів, які працюють з великими обсягами текстової інформації.

1.4.2 Архітектура роботи ML-компонента

Архітектура ML-компонента визначає, яким чином відбувається інтеграція моделі машинного навчання в iOS-додаток, і забезпечує чітку взаємодію між інтерфейсом, бізнес-логікою та обчислювальними механізмами. Основу становить модель формату mlmodel, яка завантажується застосунком і використовується для обробки введених текстових даних. Архітектура має передбачати попередню обробку тексту, виконання inference та передачу результатів у зручному форматі.

У структурі iOS-додатка важливо виділити окремий шар, який відповідає за взаємодію з ML-моделлю. Зазвичай це ViewModel або спеціалізований сервіс, який отримує текст, передає його до моделі та обробляє результат. Це дозволяє ізолювати логіку машинного навчання від інтерфейсу, забезпечуючи гнучкість і масштабованість. У разі зміни або оновлення моделі достатньо модифікувати лише цей компонент, не порушуючи структури інших частин застосунку.

Однією з ключових вимог до архітектури є оптимізація обчислень. ML-компонент має працювати швидко та без втрати точності, щоб забезпечувати аналіз у реальному часі. Для цього використовуються можливості Core ML, які автоматично адаптують модель під апаратні ресурси пристрою. Такий підхід не лише підвищує ефективність виконання, але й дозволяє зберігати стабільність роботи застосунку на широкому спектрі пристроїв різних поколінь.

1.4.3 UX/UI-функціональність

UX/UI-функціональність відіграє ключову роль у зручності використання мобільного застосунку, оскільки саме інтерфейс визначає якість взаємодії користувача з результатами машинного навчання. Інтерфейс має бути мінімалістичним та інтуїтивним, щоб користувач без додаткових пояснень міг зрозуміти, як саме вводити текст та інтерпретувати результат аналізу. Чітке розташування елементів, логічні переходи між екранами та проста навігація є основою ефективності UX.

Застосунок повинен забезпечувати інформативну та зрозумілу візуалізацію результатів роботи ML-моделі. Колірні індикатори, анімації або графічні позначення дають користувачу можливість швидко зрозуміти, яку тональність визначив алгоритм. Візуальна складова також має враховувати доступність: оптимальні кольори, достатній контраст та можливість адаптації інтерфейсу для користувачів із порушеннями зору.

Крім зовнішнього вигляду, UX включає загальну зручність роботи з застосунком. Інтерфейс має реагувати миттєво та плавно, без надмірних анімацій, які можуть уповільнювати взаємодію. Підтримка сучасних жестів, адаптивний дизайн та узгодженість елементів із рекомендаціями Apple Human Interface Guidelines забезпечують комфортну роботу користувача та створюють відчуття цілісності системи. У поєднанні з ML-компонентом UX/UI робить застосунок не лише функціональним, а й приємним у використанні.

1.4.4 Вимоги до продуктивності та приватності

Вимоги до продуктивності є одними з ключових при розробці застосунку з аналізу тональності тексту, адже користувачі очікують миттєвого отримання результатів. ML-модель має працювати швидко, забезпечуючи обчислення у режимі реального часу навіть під час введення

тексту. Це передбачає оптимізацію виконання, використання апаратних можливостей пристрою та мінімальні затримки при виклику моделі. Стабільність роботи застосунку на різних пристроях також є важливим критерієм.

Приватність даних є не менш важливою вимогою, оскільки введений текст може містити конфіденційну або персональну інформацію. Модель має працювати виключно локально, без передачі даних на сторонні сервери. Такий принцип підвищує рівень довіри користувача та відповідає сучасним стандартам безпеки. Важливо також, щоб застосунок не зберігав текстові дані без явної згоди користувача та не здійснював фонові надсилання інформації.

Застосунок має бути оптимізований з точки зору ресурсів пристрою. Мінімальне використання оперативної пам’яті, помірне споживання енергії та стабільна робота під різним навантаженням забезпечують комфортну взаємодію з системою. У поєднанні з вимогами конфіденційності це формує високий рівень якості та надійності, який є необхідним для мобільних продуктів, що використовують машинне навчання.

1.5 Постановка задачі

Постановка задачі передбачає визначення цілей розробки та основних вимог до створення мобільного застосунку для аналізу тональності тексту. Метою є створення iOS-додатка, здатного виконувати класифікацію текстових фрагментів на основі локальної моделі машинного навчання, забезпечуючи швидкість обробки та конфіденційність даних користувача. Для цього необхідно дослідити предметну область, підібрати відповідні технології, розробити архітектуру системи, інтегрувати ML-модель у мобільне середовище та забезпечити інтуїтивний інтерфейс, який дозволить користувачам ефективно взаємодіяти з результатами класифікації.

2 ТЕОРЕТИЧНІ ДОСЛІДЖЕННЯ

2.1 Сентиментний аналіз як задача обробки природної мови

Сентиментний аналіз є одним із ключових напрямів обробки природної мови, що передбачає визначення емоційного або оцінного забарвлення текстових даних. Такий підхід дозволяє автоматизувати інтерпретацію суб’єктивної інформації, що міститься у повідомленнях, коментарях, відгуках чи соціальних взаємодіях. Оскільки текстові дані є одним з основних каналів комунікації в цифровому середовищі, застосування методів сентиментного аналізу стає важливою складовою багатьох інтелектуальних систем.

У контексті обробки природної мови сентиментний аналіз поєднує методи лінгвістичного аналізу та машинного навчання, що дозволяє системам виявляти закономірності у текстах різного типу. Залежно від підходу, аналіз може виконуватися на основі словникових ресурсів або за допомогою алгоритмів навчання на розмічених даних. Розвиток сучасних технологій призвів до появи більш складних моделей, здатних працювати з контекстом, враховувати неоднозначність висловлювань та адаптуватися до різних сфер застосування. Це робить сентиментний аналіз універсальним інструментом для підтримки прийняття рішень у бізнесі, медіа, соціальних сервісах та інших цифрових платформах.

2.1.1 Визначення та сутність сентиментного аналізу

Сентиментний аналіз визначають як процес автоматичного виявлення емоційного забарвлення тексту з метою класифікації його як позитивного, нейтрального або негативного (рисунок 2.1). Цей підхід ґрунтується на аналізі лексичних, синтаксичних та семантичних характеристик повідомлення, що дозволяє системам інтерпретувати емоційний зміст

висловлювань. На зображені нижче демонструється поділ текстів за типами оцінки, де кожен фрагмент має відповідний емоційний індикатор, що відображає загальний настрій вислову. Така класифікація дає змогу алгоритмам уніфіковано обробляти суб'єктивні повідомлення, які відіграють важливу роль у цифровій комунікації.

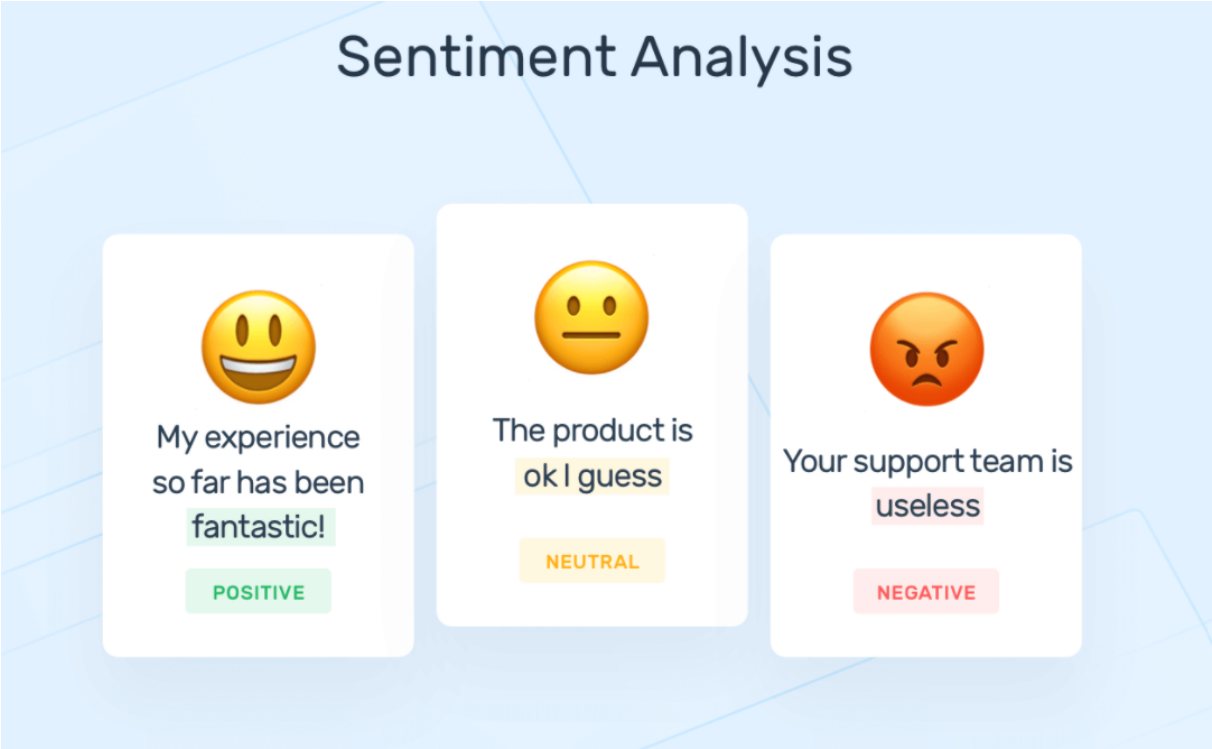

Рисунок 2.1 – Приклад роботи сентиментного аналізу

Сутність сентиментного аналізу полягає у здатності алгоритмів виявляти не лише загальну тональність тексту, але й приховані емоційні аспекти, що можуть відображати задоволення, розчарування, нейтральне ставлення або інші емоційні реакції користувачів. Завдяки цьому сентиментний аналіз стає основою для побудови інтелектуальних систем, здатних реагувати на емоційний стан користувача, адаптувати контент і покращувати взаємодію у цифрових сервісах.

2.1.2 Класифікація різновидів аналізу тональності тексту

Класифікація різновидів сентиментного аналізу охоплює кілька основних напрямів, що дозволяють глибше інтерпретувати емоційний зміст текстів (рисунок 2.2). Поширеним підходом є fine-grained аналіз, який передбачає деталізовану шкалу емоцій від дуже позитивної до дуже негативної та використовується для точнішої оцінки настрою. Інший важливий різновид – aspect-based аналіз, що фокусується на визначенні тональності стосовно конкретних характеристик або аспектів об’єкта.

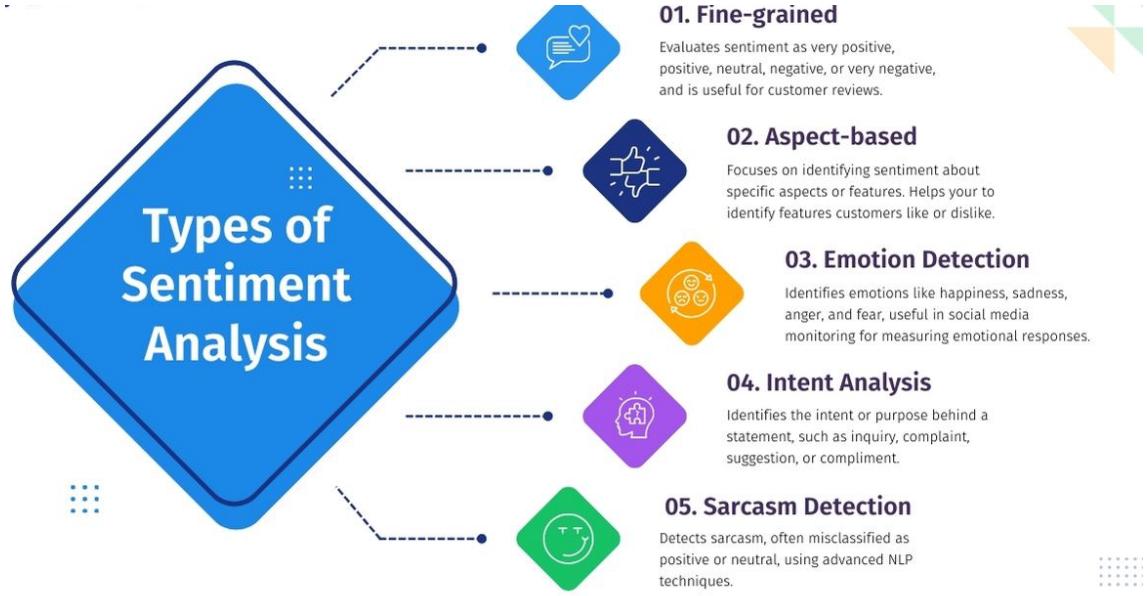

Рисунок 2.2 – Основні різновиди сентиментного аналізу

До сучасних різновидів також належать emotion detection, intent analysis та sarcasm detection. Emotion detection спрямований на визначення конкретних емоцій, таких як радість, сум або гнів, тоді як intent analysis допомагає виявляти наміри користувача, наприклад подяку, скаргу чи пропозицію. Sarcasm detection вирішує складне завдання розпізнавання сарказму, який часто маскує справжню тональність висловлювання.

2.1.3 Сфери застосування сентиментного аналізу

Сфера застосування сентиментного аналізу (рисунок 2.3) охоплює широке коло завдань, пов’язаних із виявленням настроїв та ставлення користувачів у різних цифрових середовищах. Однією з ключових областей є моніторинг соціальних медіа, де алгоритми аналізують коментарі, дописи та відгуки для виявлення тенденцій у поведінці та настроях аудиторії.

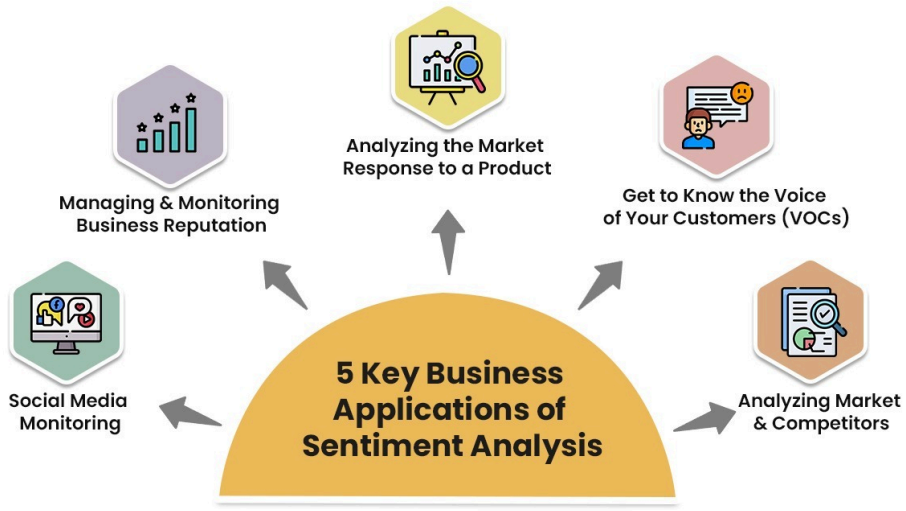

Рисунок 2.3 – Основні напрями застосування сентиментного аналізу

До важливих напрямів належить також управління репутацією бізнесу, що передбачає аналіз реакцій на події, продукти або комунікаційні кампанії. Також цей метод відіграє важливу роль у дослідженні ринку та конкурентного середовища, де він допомагає оцінювати реакцію на продукти, порівнювати ефективність брендів та визначати потреби цільової аудиторії. Даний інструмент використовується у маркетингу, сфері обслуговування, електронній комерції та аналітичних платформах, де важливо отримувати швидку та об’єктивну оцінку емоційного змісту текстових повідомлень.

2.1.4 Класифікація підходів до аналізу тональності

Класифікація підходів до аналізу тональності (рисунок 2.4) охоплює дві основні групи методів: лексикон-орієнтовані та на основі машинного навчання. Лексиконні методи ґрунтуються на використанні словників, у яких слова вже мають визначену емоційну полярність. Такі алгоритми порівнюють текст зі словниковими ресурсами та обчислюють сумарний емоційний бал. Існує два напрями: dictionary-based та corpus-based, що різняться способом формування словників. Перший передбачає ручне створення або використання готових лексиконів, тоді як другий використовує статистичні методи з корпусів текстів.

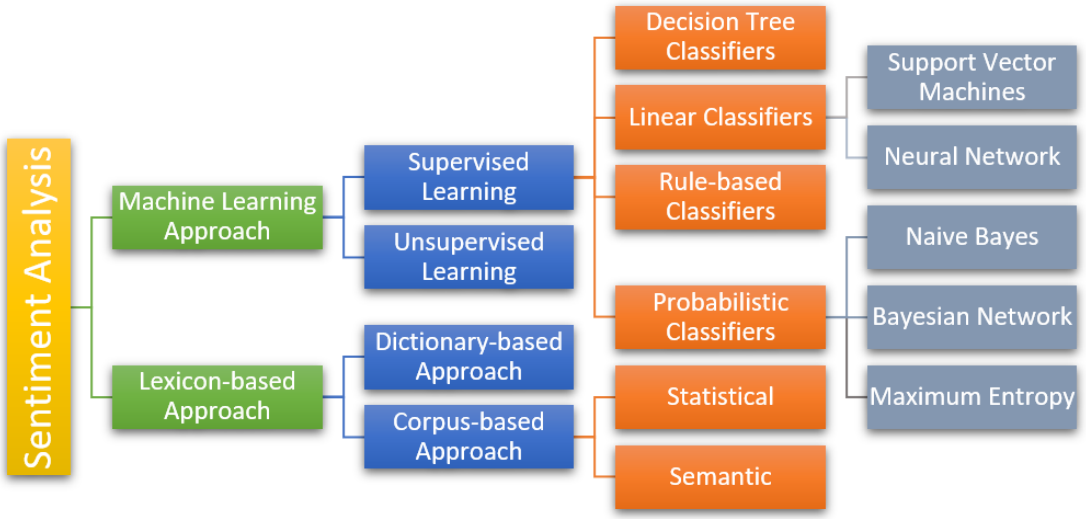

Рисунок 2.4 – Класифікація основних підходів до сентиментного аналізу

Підходи машинного навчання базуються на тому, що моделі вчаться розпізнавати емоції за прикладами розмічених текстів. Ці методи поділяються на supervised learning та unsupervised learning. Supervised система отримує тексти з мітками полярності й навчається визначати тональність нових даних. До таких підходів належать decision tree classifiers, linear classifiers, rule-based classifiers та ймовірнісні моделі. Unsupervised

learning використовують тоді, коли немає готових розмічених даних, і мета полягає у виявленні прихованих закономірностей, наприклад за допомогою кластеризації або тематичного моделювання.

Така класифікація демонструє багаторівневість підходів та дозволяє обирати метод залежно від доступних даних, складності задачі та вимог до точності.

2.1.5 Основні етапи аналізу тональності

Основні етапи аналізу тональності тексту (рисунок 2.5) охоплюють процеси підготовки даних, вилучення ознак та побудови моделі машинного навчання.

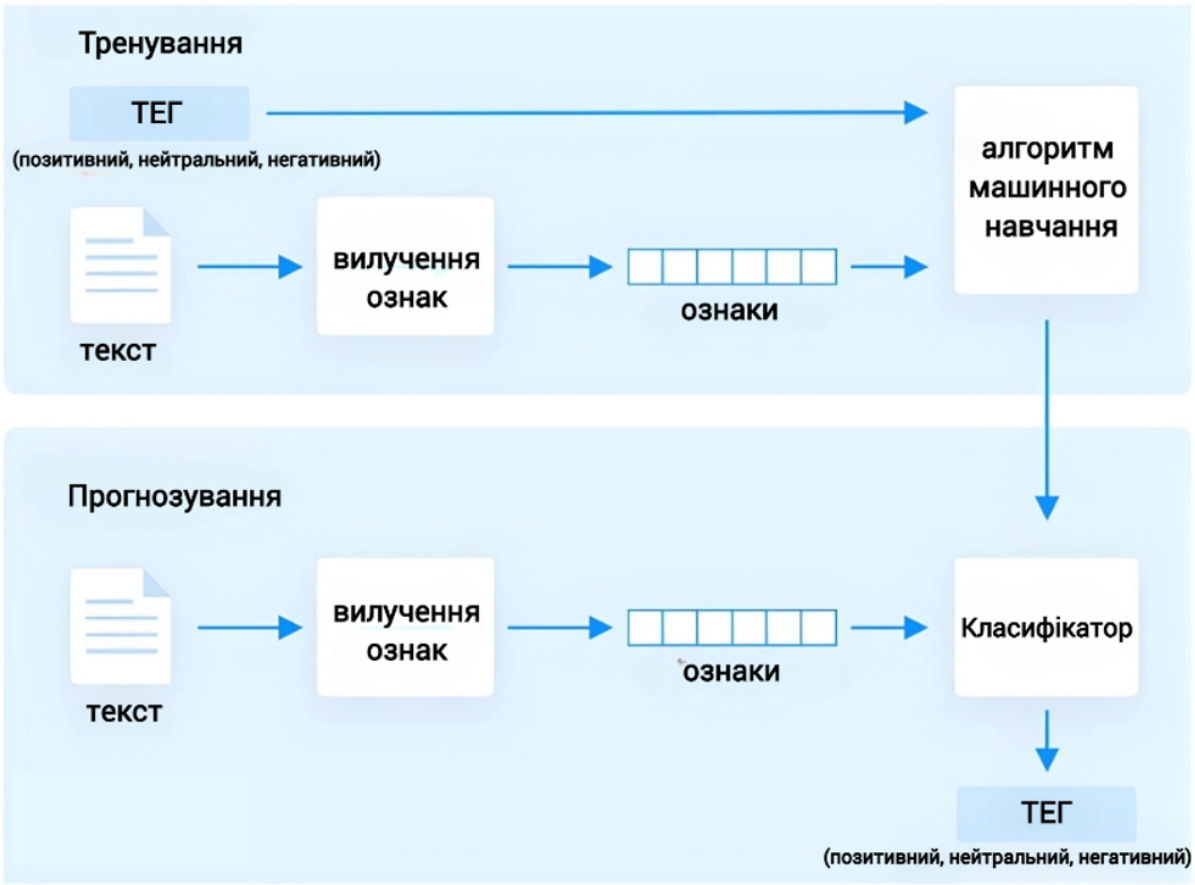

Рисунок 2.5 – Алгоритм аналізу тональності

На етапі тренування текстові дані проходять через процедуру вилучення ознак, у результаті чого формуються числові векторні представлення. Після цього алгоритм машинного навчання навчається на основі цих ознак та відповідних міток тональності, що дозволяє формувати модель, здатну розпізнавати закономірності між текстовими конструкціями та їх емоційним забарвленням.

Під час прогнозування підготовлений текст проходить ідентичний етап вилучення ознак, проте тепер ці ознаки подаються вже не на тренування, а на класифікацію. Класифікатор визначає найбільш імовірну тональність нового тексту, використовуючи знання, отримані під час навчання моделі. Така структурована послідовність етапів забезпечує узгодженість між процесами тренування та прогнозування і дозволяє досягати високої точності класифікації при застосуванні різних методів машинного навчання та сучасних моделей обробки природної мови.

2.2 Лексикон-орієнтовані підходи

Лексикон-орієнтовані підходи базуються на (рисунок 2.6) попередньо сформованих словниках, у яких кожне слово має задану емоційну полярність. Такі методи не потребують етапу тренування моделей машинного навчання, а натомість працюють шляхом порівняння тексту зі словниковими ресурсами, визначаючи кількість позитивних і негативних термів.

Процес починається з завантаження текстових даних, їх очищення та перетворення у документний формат, після чого кожне слово позначається відповідно до наявності у словнику позитивних чи негативних лексем. Це дозволяє визначити загальну тональність тексту на основі простих, але ефективних правил відповідності.

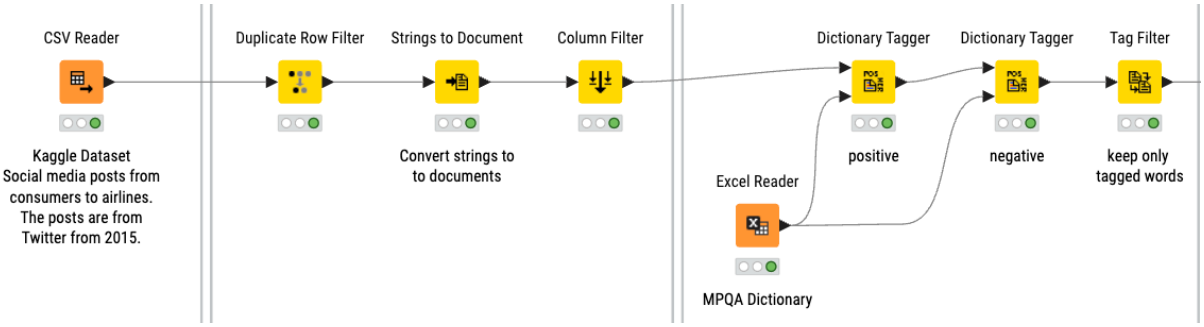

Рисунок 2.6 – Узагальнена схема лексикон-орієнтованого аналізу тональності

Застосування лексиконних підходів є поширеним у задачах, де важлива прозорість алгоритму та можливість інтерпретувати рішення. Такі методи мають низку обмежень, зокрема чутливість до багатозначності слів і складність обробки контексту, проте вони залишаються фундаментальною складовою теоретичної бази досліджень у сфері сентиментного аналізу та використовуються як базові моделі порівняння.

2.2.1 Словникові підходи

Словникові підходи ґрунтуються на використанні спеціально сформованих лексичних ресурсів, у яких кожне слово має наперед визначену полярність або емоційний коефіцієнт. Такі словники можуть містити як універсальні емоційні маркери, так і специфічні терміни для окремих доменів. Метод аналізує текст шляхом пошуку збігів між його словами та словниковими елементами, після чого визначає тональність на основі сумарних оцінок. Завдяки своїй простоті та інтерпретованості словникові підходи широко використовуються у початкових дослідженнях і в системах, де важлива прозорість результатів.

До переваг цього підходу належить можливість отримати результат без етапу тренування моделі, що значно зменшує обчислювальні витрати. Лексикони легко редагувати або розширювати вручну, адаптуючи їх під

конкретні потреби. Водночас мінусом є обмежена гнучкість, адже словники погано враховують контекст, ідентифікацію сарказму або складних синтаксичних структур. Крім того, однакові слова можуть мати різну полярність залежно від ситуації, що знижує точність аналізу в реальних сценаріях.

2.2.2 Corpus-based підходи

Corpus-based підходи виникли як спроба подолати недоліки статичних словників шляхом автоматичного формування лексичних ресурсів із великих корпусів текстів. В основі цього підходу лежить статистичний аналіз частотності слів, співвзустрічей та контекстних залежностей. Система визначає полярність термінів не вручну, а на основі закономірностей, виявлених у текстових масивах. Це забезпечує кращу адаптацію словника до конкретної предметної області й дозволяє автоматично враховувати контекст.

Перевагою corpus-based методів є здатність відображати реальне вживання слів та їх емоційні відтінки в конкретних типах текстів, зокрема в соціальних мережах або тематичних форумах. Такі словники краще передають специфіку мови користувачів та менш чутливі до переносних значень. Однак значним недоліком є потреба в наявності великих корпусів даних і відповідних обчислювальних ресурсів. Крім того, автоматично згенеровані лексикони можуть містити шум або помилки, що потребує додаткового контролю якості.

Corpus-based та словникові підходи суттєво відрізняються за способом формування лексикону. Перший забезпечує гнучкість та адаптивність, тоді як другий гарантує стабільність і прозорість. У практиці сентиментного аналізу їх часто комбінують, щоб досягти балансу між точністю та інтерпретованістю.

2.2.3 Переваги та обмеження лексиконних методів

Лексиконні методи мають низку суттєвих переваг, які роблять їх привабливими для базового або початкового аналізу тональності. Вони не потребують наявності навчальних вибірок, оскільки працюють виключно на основі списків слів та правил (рисунок 2.7), де лексиконний підхід функціонує через застосування словника до нового тексту. Така відсутність залежності від великих масивів даних робить метод доступним у ситуаціях, коли розмічені корпуси відсутні або їх складно отримати. Крім того, результати лексиконного підходу легко інтерпретувати, адже класифікація базується на прозорих відповідностях між словами та їхньою полярністю.

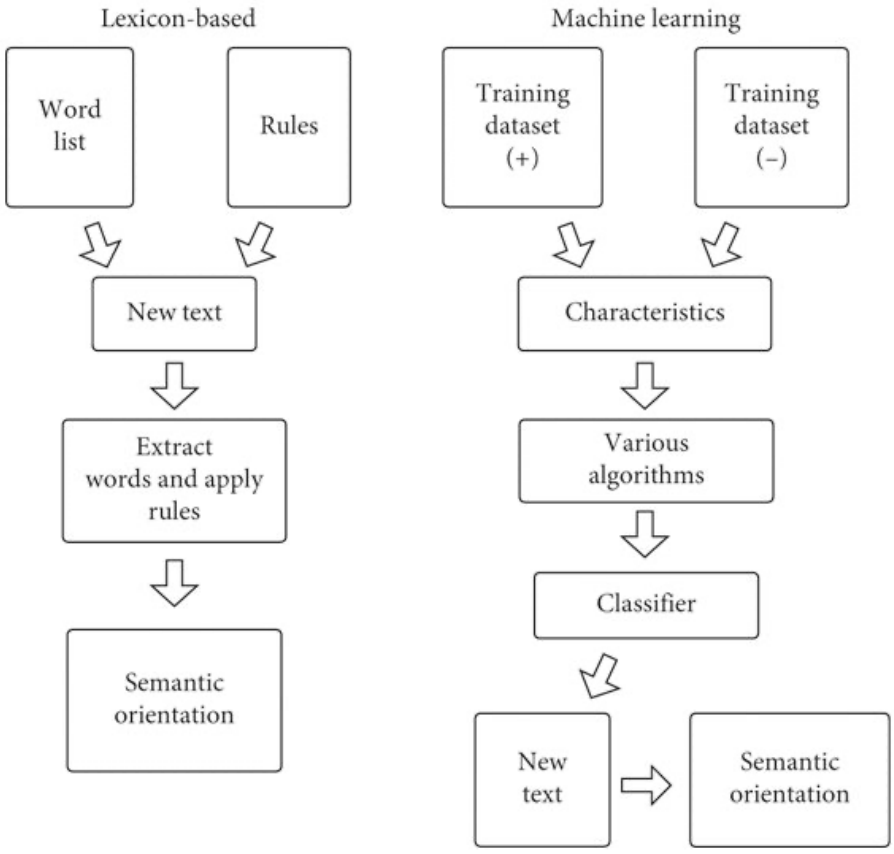

Рисунок 2.7 – Порівняння лексиконного підходу та підходу машинного навчання в аналізі тональності

Водночас лексиконні методи мають ряд обмежень, що знижує їх ефективність у порівнянні з моделями машинного навчання. Як продемонстровано вище, машинні алгоритми можуть враховувати широкий спектр характеристик тексту, тоді як лексиконні підходи обмежені статичністю словників і не здатні повноцінно інтерпретувати контекст або складні мовні конструкції. Такі методи часто помиляються у випадках багатозначних слів, сарказму або специфічної жанрової лексики. Крім того, ефективність залежить від якості словника, а його ручне створення чи оновлення є ресурсомістким процесом. Саме тому лексиконні підходи здебільшого застосовуються як допоміжні або базові методи у більш комплексних системах аналізу тональності.

2.3 Методи машинного навчання для сентиментного аналізу

Методи машинного навчання посідають центральне місце у сучасних підходах до сентиментного аналізу, оскільки дозволяють моделі навчатися на реальних текстових прикладах і виявляти складні закономірності, недоступні для словникових методів.

Як зображено нижче, процес (рисунок 2.8) включає кілька ключових етапів: збирання даних, їх попередню обробку, перетворення тексту у набори ознак та навчання моделі на основі цих ознак. Така структура забезпечує автоматичне формування узагальнених правил, на основі яких алгоритм здатен передбачати полярність нових текстів. Важливо, що моделі машинного навчання враховують широкий спектр характеристик, зокрема частоту слів, структури висловлювань та статистичні залежності між мовними елементами.

Ефективність методів машинного навчання оцінюється на тестових вибірках (рисунок 2.8), що дозволяє об’єктивно визначити здатність моделі узагальнювати інформацію й правильно класифікувати нові дані.

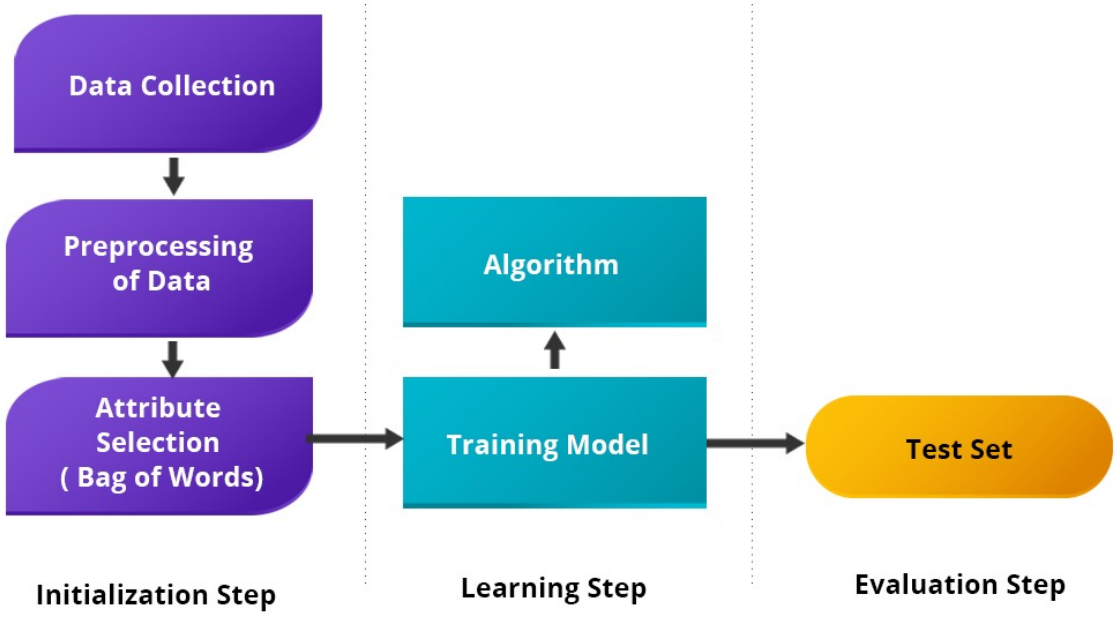

Рисунок 2.8 – Процес машинного навчання для сентиментного аналізу

Однією з ключових переваг цих методів є їх адаптивність, адже точність класифікації може зростати разом із розширенням і оновленням навчальних корпусів. Крім того, використання різних алгоритмів, таких як наївний баєсівський класифікатор, максимальна ентропія або метод опорних векторів, дає змогу налаштовувати систему відповідно до складності задачі. Саме тому машинне навчання є основою високоточних систем аналізу тональності у сучасних додатках і сервісах.

2.3.1 Підходи з учителем

Підходи з учителем ґрунтуються на використанні розмічених даних, у яких кожному тексту наперед присвоєна мітка тональності. Модель машинного навчання вивчає залежності між ознаками тексту та його емоційною категорією, формуючи узагальнені правила класифікації. Такий процес дозволяє алгоритмам автоматично розпізнавати складні мовні конструкції, контекстуальні зв’язки та латентні особливості даних, що робить підходи з учителем одними з найточніших у задачах сентиментного

аналізу. Класичними представниками цього напряму є метод опорних векторів, наївний баєсівський класифікатор, логістична регресія та сучасні нейронні мережі.

Перевагою підходів з учителем є їх висока прогностична здатність за умови наявності достатньої кількості якісно розмічених даних. Моделі здатні адаптуватися до різних типів текстів, зокрема відгуків, коментарів або коротких повідомлень, що робить їх універсальними для широкого спектра застосувань. Водночас їхнім недоліком є потреба у великих навчальних корпусах, які не завжди доступні. Розмітка даних є трудомістким процесом, що потребує часу та залучення експертів, а моделі можуть втрачати точність при переході до нових доменів, якщо розмітка не відповідає специфіці текстів.

2.3.2 Підходи без учителя

Підходи без учителя не потребують розмічених даних і застосовуються у випадках, коли мітки тональності невідомі або їх створення є неможливим чи надто дорогим. У таких підходах моделі аналізують структуру текстових даних, шукаючи приховані закономірності, схожість між текстами або спільні семантичні властивості. Найпоширенішими методами є кластеризація, тематичне моделювання та статистичні підходи до оцінки полярності.

Перевагою підходів без учителя є відсутність потреби у розмічених даних, що робить їх гнучкими та придатними для аналізу великих масивів інформації, зокрема соціальних мереж або потокових даних. Вони добре працюють на початкових етапах досліджень, коли необхідно виявити структуру масиву текстів або сформувати попередні гіпотези. Однак точність таких моделей зазвичай нижча, ніж у підходів з учителем, оскільки відсутність міток ускладнює інтерпретацію кластерів або тем.

2.3.3 Ймовірнісні класифікатори

Ймовірнісні класифікатори базуються на математичних моделях, які визначають полярність тексту шляхом обчислення ймовірностей належності документа до певної категорії. Найвідомішим представником цього класу є наївний баєсівський класифікатор, який використовує теорему Байєса та припущення незалежності ознак. Попри свою простоту, ці методи забезпечують достатньо високу точність на коротких текстах і добре працюють у задачах із великим розміром словника. Ймовірнісні моделі оцінюють внесок кожного слова у формування загальної тональності, що робить їх ефективними для аналізу частотності та статистичних залежностей.

Головною перевагою ймовірнісних класифікаторів є їх стійкість до шуму та здатність працювати на відносно невеликих обсягах даних. Крім того, вони мають високий рівень пояснюваності, оскільки модель надає чітке уявлення про те, як певні слова впливають на класифікацію. Основним недоліком є лежача в основі гіпотеза про незалежність ознак, яка рідко відповідає реальній структурі природної мови, що може призводити до помилок у складних контекстах. Проте ймовірнісні моделі часто використовують як базові класифікатори або порівняльні методи у дослідженнях сентиментного аналізу.

2.3.4 Правило-орієнтовані класифікатори

Правило-орієнтовані класифікатори працюють на основі набору логічних або лінгвістичних правил, що визначають тональність тексту шляхом аналізу його структурних і семантичних властивостей. Ці правила можуть описувати наявність полярних слів, особливості синтаксису, модифікатори інтенсивності або спеціальні конструкції, що впливають на емоційний зміст. Такий підхід дозволяє експліцитно моделювати знання

експертів та враховувати складні мовні явища, що іноді є недосяжним для статистичних методів.

Перевагою правило-орієнтованих класифікаторів є їх прозорість і повна контрольованість процесу класифікації. Можна точно простежити, яке саме правило вплинуло на визначення тональності, що є цінним у критичних застосуваннях. Проте побудова великого набору правил є трудомісткою, а підтримка їх актуальності вимагає значних зусиль. Такі системи погано масштабуються на великі обсяги текстів і слабко адаптуються до нових мовних трендів. У зв’язку з цим правило-орієнтовані підходи найчастіше використовуються як допоміжний інструмент або як складова гібридних систем.

2.3.5 Лінійні класифікатори

Лінійні класифікатори визначають полярність тексту шляхом побудови гіперплощини, яка розділяє класи у просторі ознак. До поширених представників належать логістична регресія та метод опорних векторів. Вони добре працюють у високовимірних просторах, характерних для текстових даних, де кожне слово може бути окремою ознакою. Лінійні моделі здатні захоплювати основні залежності між словами та полярністю тексту, а також ефективно працюють із великими наборами даних.

Перевагами лінійних класифікаторів є швидкість навчання, висока узагальнювальна здатність та стійкість до перенавчання при застосуванні регуляризації. Вони забезпечують стабільні результати та добре підходять для задач, де важливе балансування точності й обчислювальної ефективності. Водночас лінійні моделі мають обмеження у випадках, коли тональність залежить від складної нелінійної структури тексту, або коли вирішальну роль відіграє контекст. У таких ситуаціях необхідно застосовувати більш гнучкі моделі, наприклад нейронні мережі.

2.4 Нейронні мережі в задачах аналізу тональності

Нейронні мережі відіграють ключову роль у сучасних підходах до аналізу тональності, забезпечуючи здатність моделі автоматично виявляти складні мовні залежності та контекстуальні зв'язки. Різниця між класичними методами машинного навчання та глибинними моделями – є спосіб обробки текстових даних (рисунок 2.9). На відміну від традиційних алгоритмів, які покладаються на ручне формування ознак за допомогою методів типу bag of words або TF-IDF, нейронні мережі створюють щільні векторні представлення слів, що дозволяє більш точно відображати семантику тексту та відносини між словами.

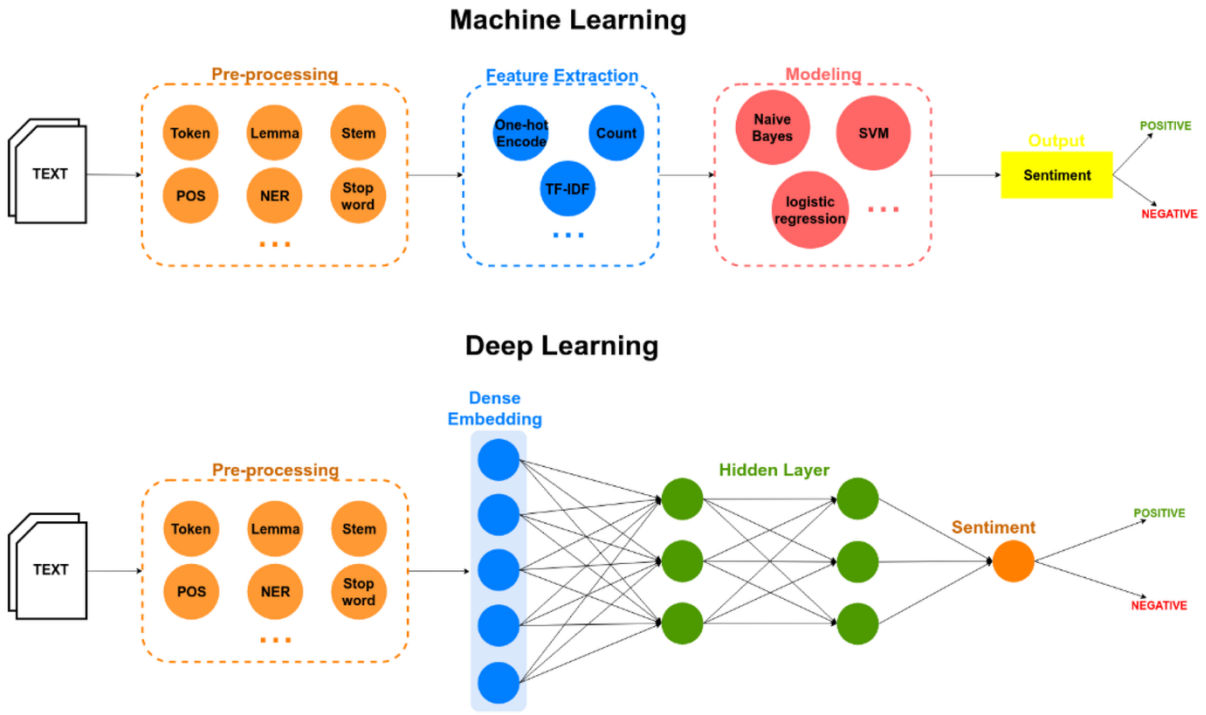

Рисунок 2.9 – Порівняння традиційних моделей машинного навчання та глибинних нейронних мереж у задачі аналізу тональності

Глибинні моделі здатні працювати з нелінійними структурами й виявляти патерни, які не очевидні при поверхневому аналізі. На рисунку

вище це ілюструється появою прихованих шарів, що дозволяють моделі автоматично навчатися складним залежностям і ефективно класифікувати тексти за тональністю. Нейронні мережі демонструють високу точність у задачах, де важливе врахування контексту, послідовності слів та емоційних відтінків. Саме тому вони стали базовим інструментом у сучасних системах, включно з мобільними застосунками, які використовують вбудовані моделі для локального та швидкого визначення полярності текстів.

2.4.1 Моделі на основі word embeddings

Моделі на основі word embeddings стали важливим етапом розвитку методів обробки природної мови, оскільки вони забезпечують щільне та семантично змістовне представлення слів у вигляді векторів. На відміну від традиційних підходів, де кожне слово подається як окрема ознака без урахування його значення, embeddings дозволяють моделі виявляти подібність між словами завдяки близькості їхніх векторів у багатовимірному просторі. Це дозволяє краще відображати контекст і значення лексичних одиниць, що є критичним для задачі аналізу тональності, де відтінки емоційного забарвлення можуть бути тонкими та контекстуально залежними.

Серед найпоширеніших підходів до побудови векторних представлень є Word2Vec, GloVe та FastText. Вони формують embeddings на основі закономірностей у великих корпусах текстів, дозволяючи моделі враховувати синтаксичні та семантичні залежності. Використання embeddings значно підвищує точність класифікації та є ефективним способом зменшення розмірності ознак, зберігаючи при цьому повноту інформації. Саме тому word embeddings стали фундаментальною складовою сучасних систем аналізу тональності та широко використовуються в нейронних мережах.

2.4.2 Рекурентні нейронні мережі

Рекурентні нейронні мережі розроблено для роботи з послідовними даними, що робить їх природним вибором для аналізу текстів, де важливим є порядок слів та контекст. На відміну від моделей, які розглядають текст як набір незалежних елементів, РНМ здатні запам'ятовувати та враховувати попередні стани під час обробки нових елементів послідовності. Завдяки цьому вони добре підходять для аналізу емоційного забарвлення висловлювань, де значення може залежати від структури та контексту речення.

Одним із найбільш ефективних різновидів РНМ є LSTM та GRU моделі, що розв’язують проблему короткої пам’яті та здатні працювати з довшими послідовностями. Ці архітектури дозволяють AI-системам точніше розпізнавати тональність, особливо у випадках, коли емоційний зміст залежить від віддалених слів або складних граматичних конструкцій. Рекурентні нейронні мережі тривалий час були основою високоточних систем аналізу тональності, однак сьогодні їх частково витісняють трансформерні моделі.

2.4.3 Трансформери як сучасний стандарт NLP

Трансформери стали революцією в обробці природної мови, оскільки вони дозволяють моделі одночасно враховувати всі елементи послідовності завдяки механізму самоуваги. На відміну від РНМ, які обробляють текст послідовно, трансформери аналізують контекст паралельно, що забезпечує високу ефективність та точність. Механізм самоуваги дає змогу визначати важливість кожного слова щодо інших, що є ключовим для задачі аналізу тональності, де емоційний відтінок може бути розподілений по всьому реченню.

Моделі на основі трансформерів, такі як BERT, RoBERTa та DistilBERT, стали стандартом у багатьох NLP-завданнях завдяки здатності захоплювати складні контекстуальні залежності та глибоко інтерпретувати текст. Їхня масштабованість та підтримка попереднього навчання на великих корпусах дозволяють досягати високої точності на задачах класифікації тональності навіть при обмежених обсягах даних. Трансформери забезпечують стабільність, узагальнювальну здатність та адаптивність, що робить їх провідною технологією сучасних інтелектуальних систем аналізу емоційного змісту текстів.

2.5 Технології інтеграції моделей машинного навчання у мобільні системи

Інтеграція моделей машинного навчання у мобільні системи базується на поєднанні фреймворків, оптимізованих бібліотек та апаратних прискорювачів, які забезпечують ефективне виконання обчислень без потреби у серверних ресурсах. У сучасних мобільних платформах реалізовано багаторівневу архітектуру, де ML-модель проходить шлях від високорівневого застосунку до низькорівневих примітивів обчислень. Як показано на відповідних схемах архітектури, фреймворки на кшталт Core ML забезпечують зв’язок між моделлю та апаратним забезпеченням, автоматично визначаючи найефективніший спосіб виконання inference. Завдяки цьому мобільні системи можуть працювати повністю автономно, зберігаючи високу продуктивність.

Важливою характеристикою мобільної інтеграції є можливість поєднання ML-моделей з доменними фреймворками, такими як Vision або SpeechKit, які розв’язують типові задачі комп’ютерного зору та обробки мовлення. Така модульність дозволяє розробникам швидко створювати інтелектуальні можливості у додатках, спираючись на оптимізовані інструменти платформи. Водночас низькорівневі бібліотеки, наприклад

Accelerate або Metal Performance Shaders, забезпечують швидке виконання тензорних операцій, що є критичним для роботи моделей у реальному часі. Таким чином, інтеграція ML у мобільні системи є результатом комплексної взаємодії програмних рівнів та апаратного прискорення.

2.5.1 Принципи локального виконання ML-моделей

Локальне виконання моделей машинного навчання ґрунтується на принципі автономності, що дозволяє здійснювати inference без звернення до зовнішнього сервера. Це забезпечує низьку затримку, високу швидкодію та підвищений рівень безпеки, оскільки дані користувача не покидають пристрою. Завдяки використанню спеціалізованих фреймворків мобільна система може автоматично обирати оптимальний ресурс для виконання обчислень, включно з CPU, GPU або Neural Engine. Такий підхід забезпечує стабільність роботи та дозволяє виконувати класифікацію тексту у режимі реального часу, що є особливо важливим для інтерактивних застосунків.

Перевагою локального виконання є також можливість працювати без доступу до мережі, що робить застосунок незалежним від інтернет-підключення та забезпечує безперервний користувацький досвід. Крім того, локальний inference зменшує навантаження на сервери та дозволяє масштабувати застосунок без додаткових інфраструктурних витрат. Для розробника це означає, що модель повинна бути компактною, оптимізованою та здатною працювати в умовах обмежених ресурсів мобільних пристроїв.

2.5.2 Оптимізація моделей для мобільних платформ

Оптимізація моделей для мобільних платформ передбачає зменшення їх розміру, прискорення виконання та зниження енергоспоживання, зберігаючи при цьому прийнятний рівень точності. До найпоширеніших

технік оптимізації належать квантизація, прунінг та конвертація у формат, адаптований до мобільних фреймворків. Такі процедури дозволяють суттєво зменшити вагу моделі, що особливо важливо для мобільних застосунків, де ресурси пам’яті та обчислювальної потужності є обмеженими. Оптимізовані моделі також швидше завантажуються та забезпечують меншу затримку під час обробки даних.

Мобільні платформи, такі як iOS, надають додаткові інструменти для оптимізації, автоматично адаптуючи модель під можливості апаратного забезпечення. Core ML, наприклад, застосовує внутрішні оптимізації та використовує спеціалізовані обчислювальні бібліотеки для пришвидшення inference. Це дозволяє досягати високої продуктивності навіть з відносно складними моделями. Оптимізація є ключовою умовою успішної інтеграції ML у мобільні застосунки, оскільки напряму впливає на час відгуку, стабільність роботи та загальний користувацький досвід.

2.5.3 Фреймворк Core ML

Фреймворк Core ML є центральним елементом екосистеми машинного навчання у мобільних пристроях Apple та забезпечує можливість виконання моделей безпосередньо на пристрої. Основна ідея фреймворку полягає у створенні єдиного механізму для імпорту, оптимізації та виконання моделей, незалежно від алгоритму чи бібліотеки, у якій вони були створені.

Нижче можна побачити (рисунок 2.10) узагальнену схему інтеграції ML-моделі у застосунок: модель у форматі .mlmodel проходить через систему Core ML, після чого стає частиною мобільного застосунку і може використовуватися для локального inference. Такий підхід приховує від розробника низькорівневі деталі роботи з апаратними ресурсами та надає високорівневий API для швидкого застосування моделі у практиці.

Рисунок 2.10 – Інтеграція ML-моделі у мобільний застосунок за допомогою Core ML

Виконання inference у Core ML передбачає обробку вхідних мультимедійних або текстових даних, які надходять у модель у вигляді структурованих ознак. Загальний процес (рисунок 2.11) виглядає наступним чином: дані надходять у Core ML модель, проходять етап внутрішньої обробки, після чого система повертає результат класифікації або регресії.

Така схема є універсальною та застосовується як для моделей комп'ютерного зору, так і для моделей обробки природної мови, включно з аналізом тональності тексту. Core ML автоматично застосовує оптимальні алгоритми обчислення та використовує відповідні апаратні компоненти, забезпечуючи низьку затримку й високу продуктивність.

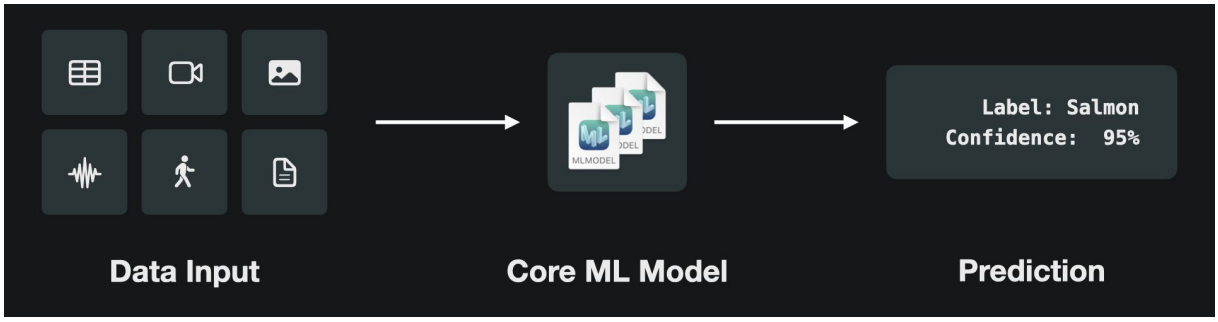

Рисунок 2.11 – Процес виконання inference у Core ML на основі вхідних даних

Місце фреймворку Core ML розташоване між доменними фреймворками (рисунок 2.12), Vision, Natural Language, Speech, Sound Analysis, та низькорівневими прискорювальними бібліотеками, Accelerate, BNNS, Metal Performance Shaders. Така структура забезпечує модульність, гнучкість і масштабованість системи. Core ML отримує можливість використовувати широкі можливості апаратного забезпечення, зокрема CPU, GPU та Neural Engine, автоматично обираючи найефективніший ресурс для виконання конкретної моделі. Завдяки цьому фреймворк здатний забезпечувати швидку обробку текстових даних, що є критично важливим для застосунків, де класифікація виконується у режимі реального часу.

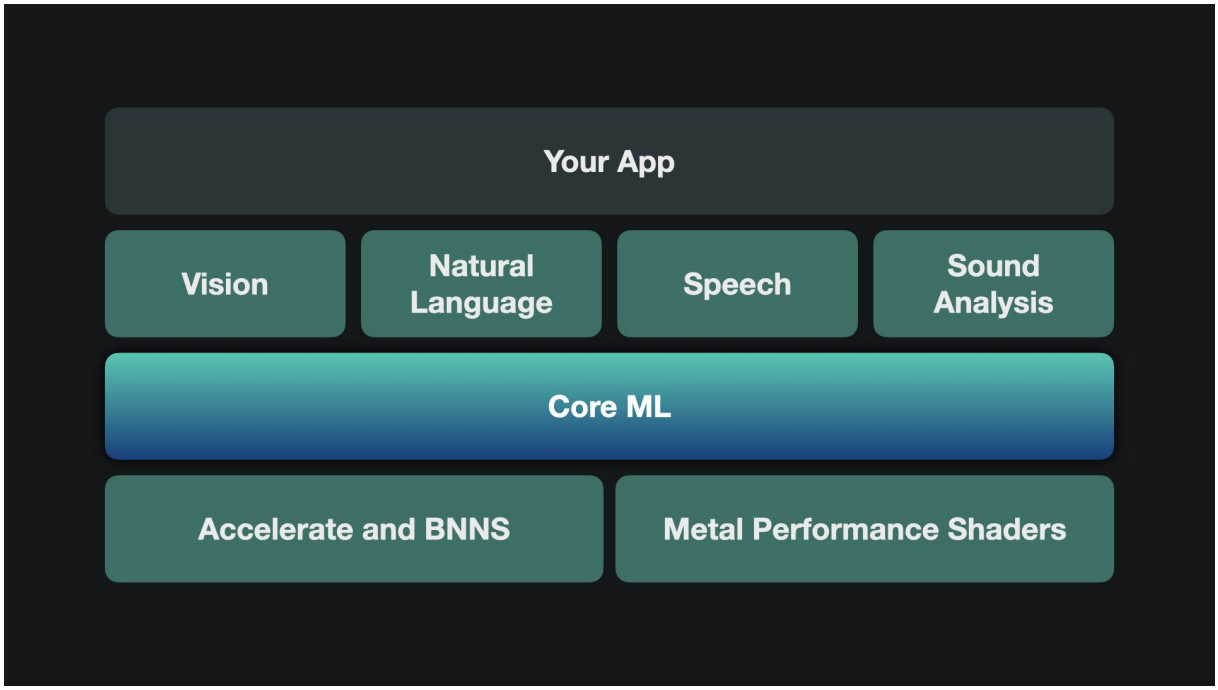

Рисунок 2.12 – Багаторівнева архітектура виконання ML-моделей на пристроях Apple

Core ML демонструє повну екосистему, яка дозволяє використовувати сучасні методи машинного навчання у мобільних додатках, забезпечуючи їхню автономність, продуктивність і енергоефективність.

3 ПРОЕКТУВАННЯ СИСТЕМИ

3.1 Аналіз вимог до мобільного застосунку

Підрозділ присвячений систематичному аналізу вимог, що визначають ключові характеристики та обмеження мобільного застосунку. У межах підрозділу розглядаються технічні вимоги, що визначають платформу, ресурси та середовище виконання; функціональні вимоги, які деталізують поведінку модулів і основні сценарії взаємодії; вимоги до інтерфейсу користувача, що забезпечують зручність та інтуїтивність роботи; а також нефункціональні вимоги, які окреслюють критерії продуктивності, надійності та безпеки системи.

3.1.1 Технічні вимоги

Оскільки система реалізована під iOS, ключовим є забезпечення сумісності з актуальними версіями операційної системи, оптимізація продуктивності під архітектури ARM64 та використання фреймворків Apple, включаючи Core ML для локального виконання моделей машинного навчання. Також враховується необхідність мінімального споживання ресурсів під час обробки тексту та побудови візуалізацій.

Другим аспектом технічних вимог є інтеграція системних компонентів, які забезпечують стабільність та безпечність роботи. До них належать можливість автономного функціонування без постійного підключення до мережі, локальне зберігання історії настрою, підтримка швидкого завантаження інтерфейсу та коректної обробки великих текстових фрагментів. Застосунок має бути оптимізований для роботи на різних моделях iPhone, враховуючи відмінності у продуктивності, розмірах екрану та обчислювальних можливостях нейронних блоків.

3.1.2 Функціональні вимоги

Функціональні вимоги визначають ядро поведінки системи та описують, які саме операції користувач може виконувати всередині застосунку. Основною вимогою є можливість здійснення миттєвого аналізу тональності введеного тексту з отриманням результатів у вигляді категорій настрою, числових індикаторів та візуальних діаграм. Також до функціональних вимог належить підтримка модулю Mood Journal, що забезпечує створення щоденних записів, їх автоматичну обробку та відображення статистики настрою в календарній сітці.

Окрему групу функціональних можливостей становить професійний модуль Tools, який передбачає роботу з текстовими файлами, імпорт з буфера обміну, ручне введення великих текстів та експорт результатів у PDF або PNG. Застосунок має коректно обробляти різні формати даних, забезпечувати стабільний бізнес-процес аналізу та гарантувати відтворюваність результатів незалежно від способу введення тексту.

3.1.3 Вимоги до інтерфейсу користувача

Вимоги до інтерфейсу охоплюють аспекти зручності, доступності та візуальної логіки застосунку. Інтерфейс повинен бути інтуїтивно зрозумілим, мінімалістичним і адаптованим під різні розміри екранів. Основні функції мають бути згруповані тематично: миттєвий аналіз, журнал настрою та професійні інструменти. Важливо забезпечити чітку візуальну ідентифікацію станів настрою через кольорові схеми, діаграми та акцентні елементи, щоб користувач без додаткових пояснень міг інтерпретувати результати аналізу.

Система також повинна підтримувати плавну навігацію між екранами, швидкі переходи та зрозумілі зворотні зв’язки при взаємодії. Значну увагу приділяють ергономіці введення тексту, читабельності елементів,

зрозумілому відображенню статистики у календарі та доступності інструментів експорту. Інтерфейс має будувати довіру та створювати відчуття особистої взаємодії користувача із власним психоемоційним профілем.

3.1.4 Нефункціональні вимоги

Нефункціональні вимоги визначають якісні показники системи, що впливають на загальний користувацький досвід. Одним із ключових критеріїв є продуктивність: застосунок повинен миттєво обробляти текст та формувати результати аналізу без затримок. Важливими є ефективне використання пам’яті та оптимізація моделі машинного навчання для мобільних пристроїв, що мінімізує навантаження на процесор і збільшує час роботи від батареї.

Безпека та конфіденційність також виступають пріоритетними нефункціональними вимогами. Оскільки застосунок працює з особистими текстами та записами емоційного характеру, усі дані повинні зберігатися локально, без передачі на зовнішні сервери. Додатковими вимогами є надійність роботи, коректне відновлення стану після перезапуску, відсутність некоректних збоїв та відповідність рекомендаціям Apple щодо приватності та UX. Такий підхід забезпечує стабільність роботи системи та захищає персональні дані користувача.

3.2 Функціональна модель системи

Функціональна модель системи (рисунок 3.1) відображає логічну структуру застосунку та взаємозв’язки між його основними модулями. Нижче показано, що мобільний застосунок Sentiment Analyzer складається з трьох окремих функціональних блоків: модуль миттєвого аналізу настрою, модуль ведення емоційного щоденника та модуль професійних інструментів

роботи з текстом. Кожен із цих модулів реалізує власні унікальні можливості, спрямовані на аналіз текстових даних, їх інтерпретацію та подальше представлення користувачеві у зручному вигляді. Така структуризація забезпечує логічний поділ завдань і підвищує зрозумілість використання системи.

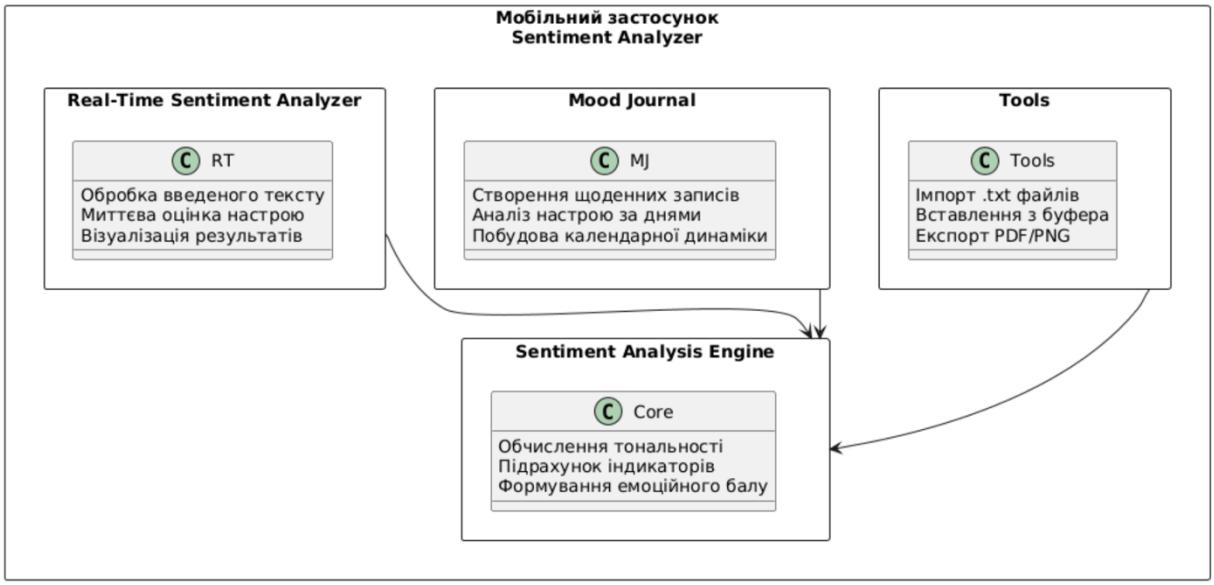

Рисунок 3.1 – Функціональна модель мобільного застосунку

Ключовим елементом моделі є компонент Sentiment Analysis Engine, який виконує роль ядра всієї системи. Саме через нього здійснюються обчислення тональності тексту, підрахунок емоційних індикаторів та формування інтегрального емоційного балу.

Як видно на зображені (рисунок 3.1), усі три модулі звертаються до зазначеного ядра, що забезпечує єдину аналітичну логіку для всіх сценаріїв використання. Така архітектурна організація не тільки спрощує взаємодію між компонентами та дозволяє постійно додавати нові модулі для масштабування, але й забезпечує узгодженість результатів аналізу незалежно від того, де саме в додатку здійснюється обробка тексту.

3.2.1 Модуль Real-Time Sentiment Analyzer

Модуль Real-Time Sentiment Analyzer відповідає за миттєве опрацювання текстових даних, введених користувачем, та негайне формування результатів аналізу. Його основне завдання полягає у визначенні емоційної тональності тексту, обчисленні позитивних і негативних індикаторів та формуванні інтегрального емоційного балу. Модуль забезпечує динамічне оновлення інтерфейсу та відображення результатів без затримок, що робить процес взаємодії з системою максимально природним та зручним для користувача.

Другим ключовим елементом функціональності є візуалізація результатів у графічному форматі. Система будує кругову діаграму та динамічні акцентовані сегменти, які дозволяють користувачеві швидко оцінити співвідношення емоційних компонентів тексту. Такий підхід сприяє інтуїтивному сприйняттю інформації та робить інструмент корисним для повсякденних рефлексивних задач, роботи з повідомленнями, нотатками чи професійним контентом.

3.2.2 Модуль Mood Journal

Модуль Mood Journal реалізує функціональність персонального емоційного щоденника, де користувач може створювати щоденні записи та аналізувати свій психоемоційний стан у динаміці. Під час введення тексту модуль здійснює його опрацювання через ядро аналізу тональності, зберігає результати та пов’язує їх із конкретною датою. Таким чином формується структурований календарний набір даних, який відображає загальні тенденції настрою користувача.

Окреме значення має інтерактивна календарна візуалізація, що дозволяє переглядати дні зі збереженими записами та оперативно повернутися до аналізу. Модуль дає змогу простежувати зміни настрою,

виявляти повторювані патерни та аналізувати власні емоційні реакції на події чи тексти. Це робить Mood Journal не тільки інструментом збереження інформації, але й засобом самоспостереження та психоемоційної рефлексії.

3.2.3 Модуль Tools

Модуль Tools зорієнтований на професійні задачі роботи з текстовими даними та надає розширений набір засобів для аналізу великих або складних матеріалів. Користувач має можливість імпортувати текстові файли формату .txt, вставляти інформацію з буфера обміну або вводити її вручну. Усі ці способи забезпечують гнучкість у роботі з коментарями, відгуками, маркетинговими матеріалами, соціальними постами чи сценаріями, що потребують тонального аналізу.

Після обробки тексту модуль дозволяє експортувати результати у формати PDF або PNG, що є важливим для презентацій, аналітичних звітів або командного обговорення. Завдяки цьому Tools перетворюється на справжній аналітичний інструментарій, який можна використовувати у маркетингу, SMM, копірайтингу та інших професійних сферах, де важливо розуміти емоційну реакцію аудиторії та оптимізувати комунікаційні матеріали.

3.3 Модель користувацької взаємодії

Цей підрозділ має на меті описати логіку роботи користувача із системою та визначити ключові сценарії, що забезпечують коректну та інтуїтивну експлуатацію мобільного застосунку. Формування моделі взаємодії є важливим етапом проєктування, оскільки саме від неї залежить зрозумілість інтерфейсу, послідовність виконання дій та ефективність доступу до функціональних можливостей системи. Чітке визначення ролей користувачів і можливих варіантів їхнього поведінкового маршруту

дозволяє забезпечити узгодженість архітектури застосунку та підвищити якість користувацького досвіду.

У межах цього підрозділу буде розглянуто основні ролі користувачів і характерні для них сценарії взаємодії із системою, визначено ключові користувацькі потоки, що описують послідовність переходів між екранами та функціональними модулями, а також побудовано схему використання мобільного застосунку. Такий підхід дає можливість сформувати цілісне бачення того, як користувач працює із системою, які дані вводить, які результати отримує та як взаємодіє з різними модулями.

3.3.1 Ролі користувачів

У контексті мобільного застосунку Sentiment Analyzer структура ролей користувачів є відносно простою, оскільки система орієнтована на індивідуальне використання без розмежування доступу між різними типами облікових записів. Основною роллю виступає звичайний користувач, який взаємодіє з функціональними модулями застосунку, вводить текстові дані, переглядає результати аналізу та працює з журналом настрою. Для цієї ролі передбачено повний доступ до можливостей системи, включно з аналізом текстів у режимі реального часу, формуванням щоденних записів і використанням інструментів імпорту та експорту даних.

Окрім основної ролі, у межах моделі взаємодії може виділятися умовна роль розширеного користувача, яка охоплює професійне використання застосунку спеціалістами, такими як маркетологи, SMM-фахівці чи аналітики. Ця роль не потребує додаткових прав доступу, проте передбачає інший характер взаємодії з модулем Tools, що дає змогу працювати з великими обсягами тексту, здійснювати імпорт даних, формувати звіти та використовувати результати аналізу в професійних цілях.

3.3.2 Основні сценарії користувацьких потоків

Основні сценарії користувацьких потоків у межах мобільного застосунку відображають характер взаємодії різних типів користувачів із системою та демонструють логіку виконання ключових дій.

У першому сценарії (рисунок 3.2) показано поведінку звичайного користувача, який працює з базовими можливостями застосунку. Його взаємодія починається з введення текстової інформації та подальшого отримання результатів миттєвого аналізу. Користувач також може створювати щоденні записи, переглядати календар настрою та аналізувати власні емоційні тенденції.

Така модель відображає послідовність дій, що формують основний користувацький досвід і забезпечують індивідуальне використання застосунку для аналізу настрою та відстеження змін емоційного стану.

Рисунок 3.2 – Сценарії для звичайного користувача

Другий сценарій (рисунок 3.3) стосується професійного користувача, який взаємодіє переважно з модулем Tools. Цей тип сценарію охоплює роботу з великими текстовими файлами, імпортом даних, аналізом відгуків, рекламних матеріалів чи контенту для соціальних мереж.

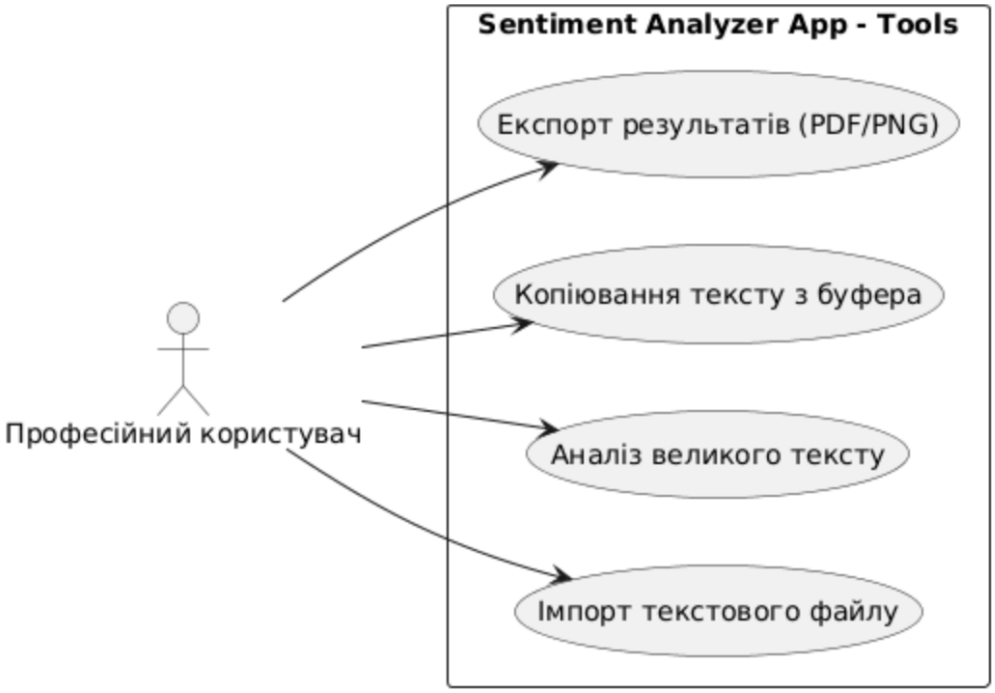

Рисунок 3.3 – Сценарії для професійного користувача

Професійний користувач може виконувати аналіз у розширеному форматі, копіювати текст із зовнішніх джерел, а також експортувати результати у PDF чи PNG для подальшої презентації або командного обговорення.

3.3.3 Схема використання мобільного застосунку

Діаграма послідовностей (рисунок 3.4), відображає цілісний процес взаємодії користувача з основними модулями мобільного застосунку Sentiment Analyzer. Вона демонструє, як різні функціональні компоненти системи координують свою роботу для забезпечення коректного аналізу

тексту, формування результатів та їх подальшої візуалізації. Такий підхід дозволяє чітко простежити логіку передачі даних між модулями та зрозуміти, яким чином користувацькі дії трансформуються у конкретні обчислювальні процеси всередині системи.

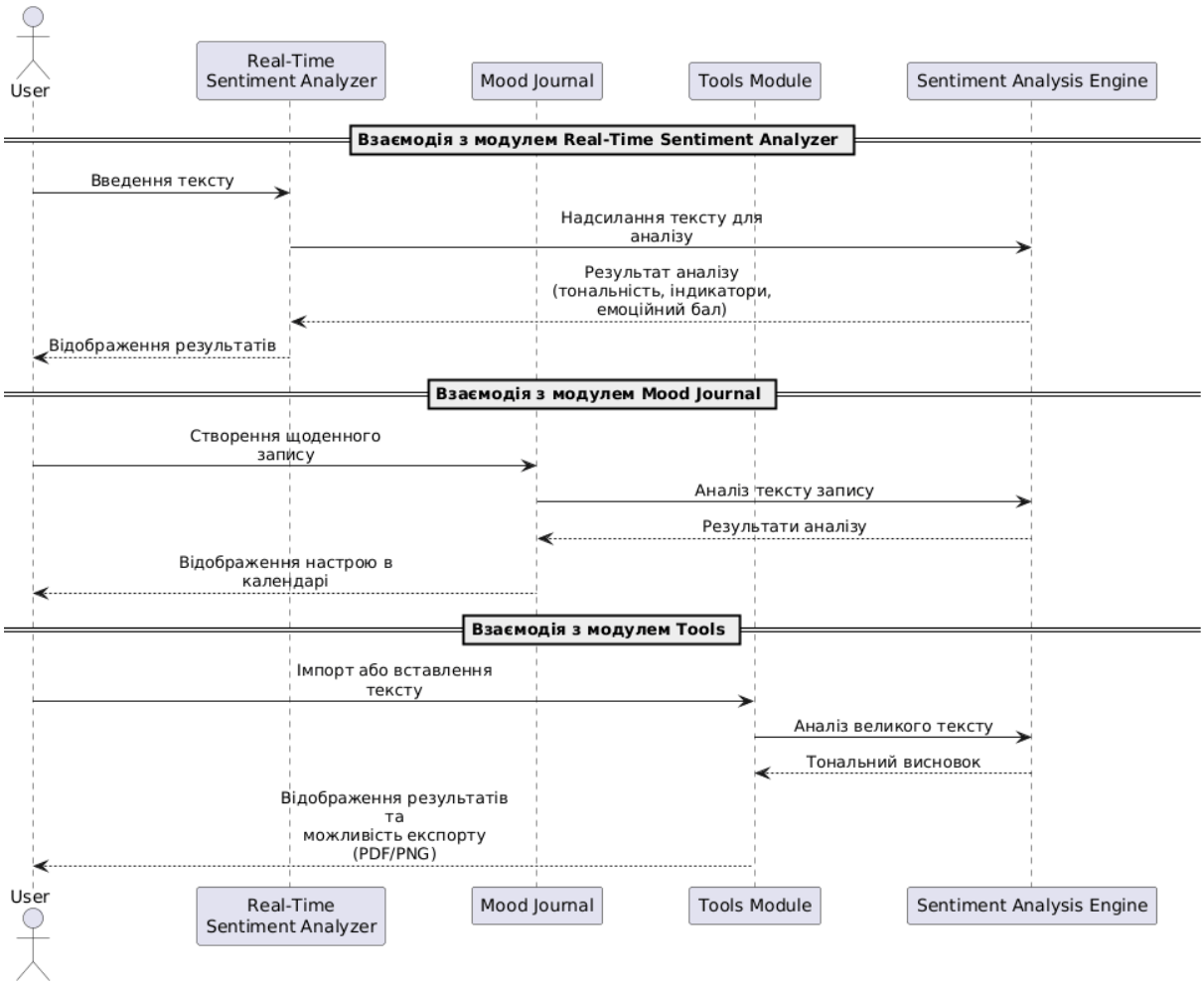

Рисунок 3.4 – Схема використання мобільного застосунку

Перша частина діаграми описує роботу з модулем Real-Time Sentiment Analyzer, де користувач вводить текст, а система через Sentiment Analysis Engine виконує оцінку тональності та повертає результати. Наступний блок демонструє роботу Mood Journal, у межах якого створені користувачем щоденні записи аналізуються тим самим механізмом, а результати

передаються назад для візуалізації у календарній структурі. Завдяки цьому користувач отримує можливість простежити власні емоційні зміни у часі.

Остання частина діаграми описує сценарій взаємодії з модулем Tools, орієнтованим на професійні аналітичні задачі. Користувач може імпортувати або вставити довші текстові матеріали, які також проходять через ядро аналізу тональності. Отримані результати виводяться у форматі, придатному для подальшої обробки та експорту. Така структура взаємодій підтверджує централізовану роль Sentiment Analysis Engine та демонструє, як усі функціональні модулі спираються на спільний механізм обробки тексту, забезпечуючи узгодженість роботи всієї системи.

3.4 Архітектура мобільного застосунку

Архітектура мобільного застосунку Sentiment Analyzer визначає структуру його внутрішньої організації та способи взаємодії між компонентами системи. Вона має забезпечувати стабільність, масштабованість та зручність у подальшій підтримці, оскільки застосунок поєднує UI-компоненти, модулі бізнес-логіки та підсистеми машинного навчання. Чітке розмежування обов’язків між архітектурними шарами дозволяє спростити процес розробки, забезпечити повторне використання окремих модулів та оптимізувати роботу з ML-моделями.

Мобільний застосунок побудований із урахуванням принципів модульності, що дає змогу незалежно розробляти й удосконалювати окремі функціональні частини, такі як Real-Time Analyzer, Mood Journal та Tools. Архітектура також передбачає централізований доступ до Sentiment Analysis Engine, який виступає єдиним джерелом машинного аналізу тексту для всіх модулів. Такий підхід забезпечує узгодженість обчислень та уніфікацію результатів аналізу.

3.4.1 Вибір технологічного стеку та його обґрунтування

В основі реалізації застосунку використано стек технологій iOS-розробки, до якого входять Swift, SwiftUI та фреймворк Core ML. Swift забезпечує високу продуктивність, безпечну роботу з пам’яттю та підтримку сучасних парадигм програмування. SwiftUI обрано як декларативний інструмент для створення інтерфейсу завдяки його гнучкості, реактивності та легкості адаптації до різних типів пристроїв. Це дозволяє створити інтуїтивно зрозумілий UI, який динамічно реагує на зміни стану даних та результати аналізу.

Використання Core ML є ключовим елементом технологічного стеку, оскільки саме цей фреймворк забезпечує можливість виконання моделей машинного навчання безпосередньо на пристрої. Такий підхід дозволяє мінімізувати затримки при аналізі тексту та гарантує приватність користувацьких даних. Обрана комбінація технологій оптимально підходить для застосунків, що потребують швидкого обчислення та якісної інтеграції ML-функцій.

3.4.2 Архітектурний підхід

Архітектура застосунку ґрунтується на шаблоні MVVM, який забезпечує чітке розмежування між інтерфейсом, логікою та даними. Модель (Model) відповідає за структуру збереження текстових записів і результати аналізу тональності, ViewModel реалізує бізнес-логіку та координує виклики Core ML-моделі, а View формує реактивний інтерфейс, який автоматично оновлюється відповідно до змін стану. Така структура підвищує зрозумілість коду та полегшує тестування окремих компонентів.

Завдяки MVVM забезпечується незалежність UI від механізмів аналізу тексту, що дозволяє легко змінювати або розширювати функціонал системи. Наприклад, додавання нового типу візуалізації або окремого

модуля не потребує значної модифікації бізнес-логіки. Цей підхід також полегшує інтеграцію ML-моделі та формування ViewModel, що опрацьовує результати inference та передає їх у зручному форматі інтерфейсу користувача.

3.4.3 Інтеграція ML-моделі за допомогою Core ML

Інтеграція ML-моделі в мобільний застосунок здійснюється за допомогою фреймворку Core ML, який забезпечує механізм локального виконання inference. Модель у форматі .mlmodel імпортується до Xcode, після чого автоматично генерується Swift-клас, що дозволяє виконувати аналіз тексту у вигляді звичайного функціонального виклику. Це істотно спрощує використання машинного навчання в мобільних застосунках і дозволяє отримувати результати з мінімальними затримками.

Під час обробки тексту ViewModel надсилає дані в модель Core ML, яка виконує класифікацію та повертає результат у вигляді ймовірностей та обраного класу тональності. Перевагою такого підходу є повністю автономна робота без використання серверних обчислень, що забезпечує приватність, високу продуктивність і доступність ML-функцій навіть у режимі офлайн. Це робить Core ML оптимальним рішенням для застосунків на кшталт Sentiment Analyzer, які потребують швидкої та надійної роботи з текстовими моделями.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ

4.1 Опис застосованих технологій

Реалізація мобільного застосунку для аналізу настрою ґрунтується на сучасних технологіях екосистеми iOS, які забезпечують високу продуктивність, стабільність, масштабованість і можливість глибокої інтеграції з вбудованими засобами машинного навчання. Базовим середовищем розробки виступає Xcode – офіційний інструмент Apple для створення застосунків під iOS, який об’єднує підтримку Swift, інструменти профілювання, дебагінгу, роботу з симуляторами та засоби оптимізації продуктивності. Завдяки Xcode забезпечується повний цикл створення застосунку – від початкової розробки структури проєкту до складання кінцевого білду для App Store або локального тестування.

Ключовою мовою програмування у проєкті є Swift, яка надає безпечну, сучасну й виразну модель програмування. Використання Swift значно спрощує побудову реактивних інтерфейсів, підвищує надійність завдяки суворій типізації та дозволяє ефективно працювати з даними у реальному часі. Особливо важливою є взаємодія Swift з фреймворками Apple, що надають API для обробки тексту, машинного навчання, оптимізації графіки та зберігання даних.

Графічний інтерфейс застосунку побудований на основі SwiftUI – декларативного фреймворку, який дозволяє описувати інтерфейс як набір взаємопов’язаних станів. SwiftUI забезпечує автоматичну адаптацію під різні розміри екранів, підтримку темного та світлого режимів, а також спрощує повторне використання компонентів. Структура Views у SwiftUI дає можливість ефективно організувати інтерфейс, що особливо важливо для модульної архітектури застосунку, де кожен функціональний компонент має власний набір візуальних представлень.

Для виконання аналізу тональності вбудовано Core ML – фреймворк, який дозволяє інтегрувати машинні моделі безпосередньо на пристрої. Core ML забезпечує швидке та енергоефективне виконання inference, адже обчислення здійснюються локально, без потреби в мережевих запитах. Це критично важливо для модулів Real-Time Sentiment Analyzer та Mood Journal, де результати повинні з’являтися негайно. Крім того, Core ML може використовувати апаратне прискорення за допомогою Neural Engine та GPU, що підвищує загальну продуктивність аналізу.

Для роботи з локальними даними застосунок використовує власні моделі, представлені як структуровані Swift-моделі. Вони забезпечують формальне описання результатів аналізу та дозволяють ефективно передавати дані між компонентами інтерфейсу та Core ML-моделлю. Крім базових моделей, у модулі Mood Journal застосовуються власні структури для зберігання, сортування та відображення щоденних записів, що дозволяє створити повноцінну систему персонального трекінгу настрою.

Також застосунок використовує графічні фреймворки SwiftUI Charts для побудови діаграм. Це дозволяє формувати інформативні елементи візуалізації, такі як кругові діаграми та акцентовані сегменти, що є важливою частиною представлення результатів аналізу.

Реалізація імпорту файлів, роботи з буфером обміну та експорту PDF/PNG використовує стандартні API iOS: UIDocumentPicker, UIPasteboard, а також SwiftUI ShareLink. Таке поєднання технологій робить модуль Tools універсальним інструментом для професійного аналізу контенту та зручної взаємодії з зовнішніми даними.

4.2 Реалізація моделей даних

Модель Sentiment, подана у лістингу 4.1, визначає категорії тональності, що використовуються у всій системі для класифікації емоційного забарвлення тексту. Ініціалізація на основі числового значення

дозволяє узгодити формальний результат алгоритму аналізу з якісною оцінкою, яку отримує користувач. Додаткові властивості icon та sentimentColor забезпечують відповідність кожного типу настрою певному візуальному маркеру, що необхідно для підтримки інтуїтивного інтерфейсу застосунку.

Лістинг 4.1 – Програмний код, що реалізує модель даних Sentiment

```
enum Sentiment: String {
    case positive = "Positive"
    case negative = "Negative"
    case moderate = "Moderate"
    case mixed = "Mixed"

    init(_ score: Double) {
        if score > 0.2 {
            self = .positive
        } else if score < -0.2 {
            self = .negative
        } else {
            self = .moderate
        }
    }

    var icon: String {
        switch self {
        case .positive: return "chevron.up.2"
        case .negative: return "chevron.down.2"
        case .moderate: return "minus"
        case .mixed: return "equal.circle"
        }
    }

    var sentimentColor: Color {
        switch self {
```

Продовження лістингу 4.1

```
            case .positive: return .teal
            case .negative: return .red
            case .moderate: return .gray
            case .mixed: return .primary
            }
        }
}
```

Модель Response, наведена у лістингу 4.2, описує структуру окремої текстової реакції користувача. Вона містить ідентифікатор, вихідний текст та числовий показник тональності, що визначається механізмом аналізу. Обчислювана властивість sentiment автоматично трансформує числовий бал у категорію настрою, забезпечуючи уніфікований спосіб представлення результатів на різних екранах застосунку.

Лістинг 4.2 – Програмний код, що реалізує модель даних Response

```
struct Response: Identifiable {
    let id: String
    let text: String
    let score: Double
    var sentiment: Sentiment {
        Sentiment(score)
    }
}
```

У лістингу 4.3 наведено реалізацію моделі Scorer, що здійснює обчислення тональності тексту з використанням NLTagger. Цей компонент відповідає за попередню обробку тексту, застосування вбудованої мовної моделі та отримання числового значення настрою. Метод score виконує аналіз покроково, перебираючи окремі речення та зчитуючи їхні емоційні

індикатори, що забезпечує більш точний результат навіть у випадку складних текстів.

Лістинг 4.3 – Програмний код, що реалізує модель Scorer для обчислення тональності тексту за допомогою NLTagger.

```
struct Scorer {
    let tagger = NLTagger(tagSchemes: [.sentimentScore])

    func score(_ text: String) -> Double {
        var sentimentScore = 0.0
        tagger.string = text

        tagger.enumerateTags(in:
text.startIndex..<text.endIndex,  unit:  .sentence,  scheme:
.sentimentScore) { tag, _ in
            if
              let sentimentString = tag?.rawValue
              let score = Double(sentimentString)
            {
                sentimentScore = score
            }
            return false
        }
        return sentimentScore
    }
}
```

Модель MoodType у лістингу 4.4 визначає внутрішнє подання емоційного стану, що використовується здебільшого в модулі Mood Journal. Вона спрощує класифікацію настрою, зводячи її до трьох базових категорій. Додатково кожен стан асоціюється з кольором, що дозволяє створювати календарний огляд настроїв у візуально послідовному форматі.

Лістинг 4.4 – Програмний код, що реалізує модель MoodType

```
private enum MoodType {
    case positive
    case moderate
    case negative

    var color: Color {
        switch self {
        case .positive: return .teal
        case .moderate: return .gray
        case .negative: return .red
        }
    }
}
```

У лістингу 4.5 подано модель MoodDay, що описує дані для одного календарного дня в журналі настрою. Структура містить унікальний ідентифікатор, дату та визначений стан настрою, що дозволяє формувати повноцінну матрицю місяця і відображати емоційну динаміку користувача у зручному графічному форматі. Такий підхід забезпечує логічну основу для реалізації інтерактивного календаря.

Лістинг 4.5 – Програмний код, що реалізує модель MoodDay

```
private struct MoodDay: Identifiable {
    let id = UUID()
    let date: Date
    let mood: MoodType
}
```

Модель SentimentResult, подана у лістингу 4.6, об’єднує результат аналізу тональності у зручну для відображення структуру. Вона містить тип настрою, нормалізоване числове значення та коротке пояснення, яке допомагає користувачу інтерпретувати результат. Властивість labelText

формує стислий текстовий опис, що використовується у візуальних компонентах модуля Mood Journal.

Лістинг 4.6 – Програмний код, що визначає модель SentimentResult для відображення результатів аналізу настрою

```
private struct SentimentResult {
    let moodType: MoodType
    let score: Double
    let explanation: String

    var labelText: String {
        switch moodType {
        case .positive: return "Mostly Positive"
        case .moderate: return "Mixed / Neutral"
        case .negative: return "Mostly Negative"
        }
    }
}
```

Модель JournalEntry, наведена у лістингу 4.7, представляє окремий запис емоційного щоденника. Вона зберігає дату, визначений стан настрою та текстову анотацію, що дозволяє користувачу переглядати й аналізувати історію власних емоційних реакцій. Метод dateString формує уніфікований формат дати, придатний для відображення в інтерфейсі, забезпечуючи цілісність представлення журналу.

Лістинг 4.7 – Програмний код, що реалізує модель JournalEntry для збереження щоденних записів у Mood Journal

```
private struct JournalEntry: Identifiable {
    let id = UUID()
    let date: Date
    let mood: MoodType
    let preview: String
```

Продовження лістингу 4.7

```
    var dateString: String {
        let formatter = DateFormatter()
        formatter.dateStyle = .medium
        return formatter.string(from: date)
    }
}
```

Створені моделі даних формують цілісну та узгоджену основу для функціонування всього мобільного застосунку, забезпечуючи структуроване представлення як результатів аналізу тональності, так і елементів емоційного щоденника.

4.3 Інтеграція моделі машинного навчання для аналізу тональності

Інтеграція моделі машинного навчання для аналізу тональності в мобільний застосунок здійснюється через використання попередньо натренованої моделі SentimentClassifier, яка була побудована на основі корпусу текстових даних із розміченими прикладами позитивних, нейтральних та негативних висловлювань.

На етапі навчання модель було оптимізовано для задачі текстової класифікації, що дало змогу одержати узагальнювальну здатність до коректного розпізнавання настрою нових фрагментів тексту, які не входили до тренувальної вибірки.

У процесі інтеграції результати моделі інкапсулюються у функцію високого рівня analyzeSentiment для того, щоб вона завантажила модель, в яку передається вхідний текст та повертається інтерпретований результат у вигляді текстового опису настрою, як це реалізовано у програмному коді, який поданий у лістингу 4.8.

Лістинг 4.8 – Програмний код, що аналізує настрій певного фрагмента тексту за допомогою власної моделі

```
func analyzeSentiment(text: String) -> String {
    guard let model = try? SentimentClassifier(configuration: MLModelConfiguration()) else {
        return "Model loading failed"
    }
    guard let prediction = try? model.prediction(text: text) else {
        return "Prediction failed"
    }
    return "Sentiment: \(prediction.label.capitalized)"
}
```

Такий підхід дозволяє ізолювати деталі роботи Core ML від інших компонентів застосунку, забезпечити повторне використання логіки аналізу в різних модулях та підтримувати можливість подальшої заміни або перенавчання моделі без суттєвих змін у загальній архітектурі системи.

4.4 Реалізація функціональності модулів системи

У цьому підрозділі буде розглянуто практичну реалізацію основних модулів мобільного застосунку, які забезпечують виконання ключових функцій системи аналізу тональності.

Опис подано з урахуванням внутрішньої логіки роботи кожного модуля, його взаємодії з моделлю машинного навчання та специфіки обробки користувацьких даних. Кожен модуль реалізує окремий сценарій використання – від миттєвого аналізу тексту до ведення емоційного щоденника та професійної роботи з контентом – що забезпечує цілісність архітектури та узгодженість інструментів між собою.

4.4.1 Модуль Real-Time Sentiment Analyzer

Модуль Real-Time Sentiment Analyzer забезпечує миттєвий аналіз тексту, який вводить користувач, та відображення результатів у динамічному інтерфейсі.

Основою його роботи є обчислення тональності у реальному часі та формування нових об’єктів відповіді, що додаються до списку попередніх оцінок. Такий підхід дозволяє користувачу одразу бачити емоційний результат для кожного нового фрагмента тексту. Логіка обробки введених даних представлена у лістингу 4.9.

Лістинг 4.9 – Програмний код, що реалізує функцію для обробки введеного тексту та формування відповіді

```
    private func submit() {
        guard !responseText.isEmpty else { return }
        let score = scorer.score(responseText)
        let response = Response(id: UUID().uuidString, text:
responseText, score: score)
        responses.insert(response, at: 0)
        responseText = ""
        isFocused = false
    }
```

У наведеному коді здійснюється перевірка наявності тексту, обчислення його тональності за допомогою Scorer, створення нової моделі Response та додавання її до початку списку відповідей. Після цього текстове поле очищується, а фокус знімається, що забезпечує плавність взаємодії та безперервний аналіз наступних фрагментів. Модуль таким чином виконує роль центрального інструмента швидкої емоційної оцінки текстових даних.

4.4.2 Модуль Mood Journal

Модуль Mood Journal виконує функцію персонального емоційного щоденника, дозволяючи користувачу фіксувати власні думки та отримувати їх емоційну інтерпретацію. На відміну від миттєвого аналізатора, цей модуль орієнтований не лише на одноразовий аналіз, а на формування довгострокової картини емоційного стану користувача. Завдяки цьому він відіграє важливу роль у виявленні тенденцій, коливань настрою та аналізі психологічного благополуччя на основі текстових записів. Алгоритм визначення настрою для щоденного запису подано у лістингу 4.10.

Лістинг 4.10 – Програмний код, що реалізує алгоритм аналізу тональності щоденного запису в модулі Mood Journal

```
    private func analyzeSentiment() {
        let    text    =    journalText.trimmingCharacters(in:
.whitespacesAndNewlines)
        guard !text.isEmpty else {
            sentimentResult = nil
            return
        }

        let scorer = Scorer()
        let score = scorer.score(text)

        let moodType: MoodType
        if score > 0.2 {
            moodType = .positive
        } else if score < -0.2 {
            moodType = .negative
        } else {
            moodType = .moderate
        }
```

Продовження лістингу 4.10

```
        let explanation: String
        switch moodType {
        case .positive:
    explanation = "Your thoughts indicate a generally positive
emotional tone."
        case .moderate:
    explanation = "Your text shows mixed or neutral emotional
signals."
        case .negative:
    explanation = "Your writing contains signs of tension or
negative emotions."
        }
        let normalizedScore = (score + 1) / 2.0
        sentimentResult = SentimentResult(
            moodType: moodType,
            score: normalizedScore,
            explanation: explanation
        )
    }
```

У наведеному коді спочатку здійснюється нормалізація тексту та перевірка на його порожність. Далі за допомогою Scorer обчислюється числовий показник тональності, який класифікується у відповідний MoodType залежно від порогу.

Крім того, формується текстове пояснення, що допомагає користувачу краще зрозуміти результат аналізу. Отримані дані об’єднуються у модель SentimentResult, що включає тип настрою, нормалізований бал та пояснення, після чого результат відображається в інтерфейсі. Такий підхід забезпечує інформативність і підвищує корисність емоційного щоденника як аналітичного інструмента.

4.4.3 Модуль Tools

Модуль Tools є окремим функціональним блоком застосунку, орієнтованим на роботу з текстовим контентом зовнішнього походження. Його ключова роль полягає у забезпеченні професійного інструментарію для маркетологів, копірайтерів, аналітиків та інших користувачів, які бажають оперативно оцінювати емоційну тональність файлів.

У лістингу 4.11 наведено функціональність, що забезпечує отримання текстових даних із буфера обміну. Метод pasteFromClipboard перевіряє наявність текстового фрагмента у UIPasteboard і, у разі успіху, записує його у внутрішню змінну importedText. Такий механізм підвищує зручність модулю, даючи змогу миттєво аналізувати скопійовані коментарі або повідомлення.

Лістинг 4.11 – Програмний код, що реалізує вставлення тексту з буфера обміну

```
func pasteFromClipboard() {
    if let text = UIPasteboard.general.string {
        importedText = text
    }
}
```

У лістингу 4.12 представлено механізм генерації PDF-документа, який дозволяє формувати структурований текстовий звіт на основі імпортованих даних. Функція generatePDF застосовує UIGraphicsPDFRenderer для створення документа з визначеними параметрами сторінки, а подальше відображення тексту виконується за допомогою форматування з атрибутами шрифту. Цей функціонал є корисним для створення презентаційних матеріалів, внутрішніх звітів та формальної документації.

Лістинг 4.12 – Програмний код, що реалізує генерацію PDF-файлу результатів аналізу

```
    func generatePDF() {
        let renderer = UIGraphicsPDFRenderer(bounds: CGRect(x:
0, y: 0, width: 595, height: 842))
        let data = renderer.pdfData { context in
            context.beginPage()

            let paragraph = NSMutableParagraphStyle()
            paragraph.lineBreakMode = .byWordWrapping
            paragraph.alignment = .left

            let attributes: [NSAttributedString.Key : Any] = [
                .font: UIFont.systemFont(ofSize: 14),
                .paragraphStyle: paragraph
            ]

            importedText.draw(
                with: CGRect(x: 32, y: 32, width: 531, height:
778),
                options: .usesLineFragmentOrigin,
                attributes: attributes,
                context: nil
            )
        }
    }
```

У лістингу 4.13 демонструється створення графічного зображення, що містить текст аналізу. Метод generateImage будує зображення у форматі UIImage шляхом рендерингу тексту в контексті зазначеного розміру. Ця можливість особливо корисна для швидкого поширення результатів у соціальних мережах або месенджерах, де графічний формат часто є більш зручним порівняно з PDF.

Лістинг 4.13 – Програмний код, що реалізує формування графічного зображення результатів аналізу

```
func generateImage() {
    let size = CGSize(width: 1080, height: 1920)

    let renderer = UIGraphicsImageRenderer(size: size)
    let image = renderer.image { context in
        let paragraph = NSMutableParagraphStyle()
        paragraph.lineBreakMode = .byWordWrapping
        paragraph.alignment = .left

        let attributes: [NSAttributedString.Key: Any] = [
            .font: UIFont.systemFont(ofSize: 36),
            .paragraphStyle: paragraph
        ]

        importedText.draw(
            with: CGRect(x: 40, y: 60, width: size.width -
80, height: size.height - 120),
            options: .usesLineFragmentOrigin,
            attributes: attributes,
            context: nil
        )
    }
}
```

Останнім елементом модулю є механізм імпорту файлів, наведений у лістингу 4.14. Функція handleFileImport отримує URL імпортованого документа та намагається прочитати його вміст як текстовий файл. У разі успішного зчитування дані автоматично передаються для подальшого аналізу. Таким чином, модуль Tools забезпечує гнучкий і багатофункціональний підхід до обробки текстів, що робить його важливою складовою застосунку.

Лістинг 4.14 – Програмний код, що виконує імпорт текстового файлу та обробку отриманих даних

```
func handleFileImport(_ result: Result<URL, Error>) {
        if let text = try? String(contentsOf: url) {
            importedText = text
    }
}
```

4.5 Реалізація користувацького інтерфейсу

У цьому підрозділі описується загальна стилістика інтерфейсу мобільного застосунку, включно з логікою побудови іконки, вибором кольорової палітри та підтримкою світлого і темного режимів. Окрему увагу зосереджено на створенні мінімалістичного, емоційно нейтрального та сучасного дизайну, що узгоджується з характером застосунку для аналізу настрою.

4.5.1 Візуальна концепція застосунку

Візуальна концепція мобільного застосунку створена відповідно до принципів сучасного інтерфейсного дизайну, що поєднує мінімалізм, плавні градієнти та емоційно виразні елементи.

Центральним графічним акцентом виступає стилізована іконка у вигляді діалогової хмаринки з індикаторами настрою (рисунок 4.1), яка метафорично підкреслює основну ідею застосунку – аналіз емоцій, прихованих у тексті.

Використання об’ємних контурів і м’яких переходів між відтінками синього та фіолетового формує відчуття глибини та технологічності, одночасно зберігаючи дружній і невимушений візуальний стиль, що є невід’ємною частиною для мого застосунку.

Рисунок 4.1 – Іконка застосунку

Інтерфейс застосунку оптимізований для комфортного щоденного використання, а його кольорова палітра адаптується до світлого та темного режимів системи (рисунок 4.2). У світлій темі інтерфейс побудований на ніжних акварельних градієнтах, що підсилюють відчуття легкості, тоді як темна тема використовує глибокі синьо-фіолетові відтінки, які створюють виразний контраст і забезпечують комфортну взаємодію в умовах низької освітленості.

Особливе місце в дизайні займає система візуальних маркерів настрою: позитивний, нейтральний і негативний стани подаються через впізнавані символи та чіткі колірні асоціації, що забезпечує швидке інтуїтивне сприйняття результатів аналізу навіть без детального ознайомлення з текстовою інтерпретацією. Для кожного типу настрою використано унікальну комбінацію піктограм та кольорів: стрілка вверх у бірюзовому відтінку позначає позитивну тональність, стрілка вниз червоного кольору – негативну, а горизонтальна риска сірого кольору – нейтральний стан.

Рисунок 4.2 – Підтримка світлого та темного режиму

Крім того, маркери настрою інтегровані в усі ключові екрани застосунку – від списку аналізованих повідомлень до візуальних компонентів журналу настрою – що робить їх універсальним елементом навігації та індикатором емоційної динаміки. Завдяки цьому користувач отримує не лише числовий чи текстовий опис тональності, а й миттєвий емоційний сигнал, який допомагає швидко визначити характер тексту, оцінити власний стан або порівняти різні записи між собою. У сукупності

це створює інтерфейс, який не просто відображає дані, а допомагає користувачу краще зрозуміти власні емоції та реагувати на них усвідомлено.

Такий підхід підсилює інтуїтивність взаємодії й дозволяє користувачу миттєво розпізнати зміст аналізу навіть без текстових пояснень. Завдяки ретельно продуманим візуальним рішенням застосунок виступає не просто технічним інструментом, а емоційно комфортним середовищем для рефлексії та самоспостереження.

4.5.2 Екран миттєвого аналізу тональності

Екран миттєвого аналізу тональності є центральним елементом мобільного застосунку, оскільки саме він забезпечує користувача основним функціоналом – оперативною оцінкою емоційного змісту введеного тексту.

Загальний вигляд інтерфейсу в світлому режимі (рисунок 4.3) демонструє гармонійне поєднання аналітичних компонентів і візуальної естетики. Центральне місце займає інтерактивна донаподібна діаграма, яка не лише відображає співвідношення позитивних, нейтральних і негативних відповідей, але й формує первинний емоційний образ стану користувача.

Сегменти діаграми відрізняються не лише кольором, а й виразністю відтінків, що дозволяє швидко ідентифікувати домінувальну категорію настрою. Такий підхід забезпечує інформативність із першого погляду, оскільки навіть без детального читання значень користувач миттєво сприймає загальну тенденцію емоційного профілю.

Діаграма органічно інтегрується в загальний дизайн екрана: м’які тіні, округлені кути та напівпрозоре тло формують легку, повітряну композицію, яка мінімізує відволікання та підсилює зосередженість на ключових даних. Крім того, плавні анімації оновлення сегментів сприяють природному сприйняттю змін, створюючи ефект «живої» аналітики.

Рисунок 4.3 – Екран миттєвого аналізу тональності

Така візуалізація дозволяє моментально оцінити загальний емоційний фон, а плавні кольорові переходи створюють відчуття цілісності та легкості взаємодії. Додаткові індикатори – кількість оброблених відповідей та текстовий опис загальної тональності – формують інформативний контекст і створюють умови для правильного сприйняття результатів аналізу.

Далі можна побачити у середній частині екрана представлено картку Overall Sentiment (рисунок 4.4). Вона інтегрує піктограму, текстовий результат класифікації та показники розподілу емоційних станів.

Завдяки візуальній акцентуації (кольорові стрілки, контрастні відтінки, напівпрозоре тло) користувач швидко розпізнає домінуючий емоційний напрям.

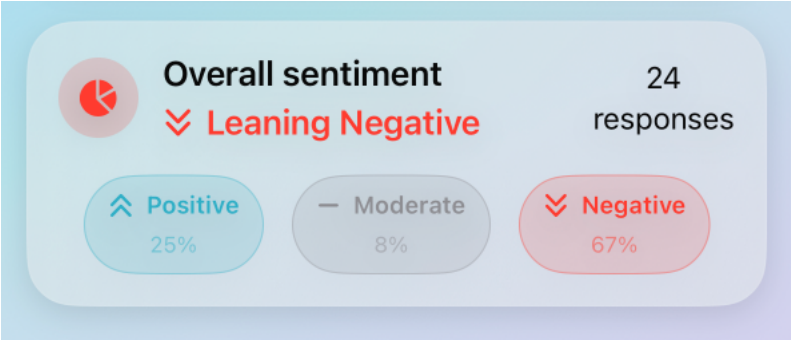

Рисунок 4.4 – Секція з більш детальними даними

Індивідуальні картки окремих відповідей (рисунок 4.5) забезпечують деталізований огляд кожного введеного фрагмента тексту. Кожна картка містить піктограму відповідного настрою, короткий текст та кастомний градієнтний індикатор, що полегшує візуальне сканування навіть за великої кількості записів.

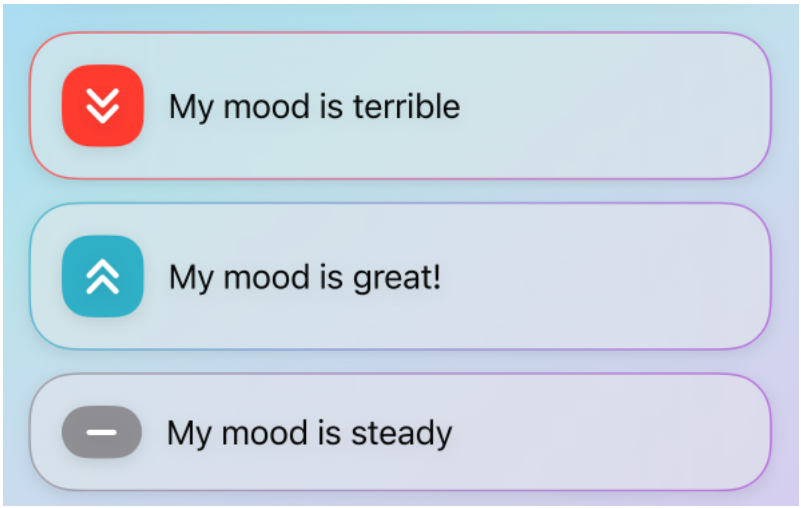

Рисунок 4.5 – Приклад відображення результатів

Структура взаємодії на екрані побудована таким чином, щоб мінімізувати когнітивне навантаження. Поле введення розміщено внизу екрана, що відповідає стандартам iOS Human Interface Guidelines та забезпечує зручність введення тексту однією рукою.

Кнопка Send, виконана в насичено-оранжевих тонах, створює чіткий контраст і виступає основним тригером дії, фокусуючи увагу користувача. Важливою частиною реалізації є динамічна анімація появи нових відповідей, що візуально підсилює відчуття «живої» аналітики та забезпечує приємний користувацький досвід.

Загалом екран миттєвого аналізу тональності відповідає ключовим принципам сучасних мобільних інтерфейсів: емоційності, адаптивності та виразності. Він об’єднує складні алгоритми аналізу емоційного змісту з інтуїтивно зрозумілою візуальною мовою, що забезпечує комфортне й ефективне використання системи в реальному часі.

У лістингу 4.15 подано програмний код, що визначає інтерфейс екрана миттєвого аналізу настрою. Його структура вибудувана на основі контейнера NavigationStack, який забезпечує коректну навігаційну поведінку та підтримку великого заголовка екрана. Основу візуального оформлення становить градієнтний фон backgroundGradient, інтегрований у ZStack, який накладається на весь екран і формує цілісне стилістичне середовище застосунку.

Внутрішня логіка компонування реалізована через вертикальний стек VStack, де послідовно розміщені елементи аналізу настрою. У прокручуваній області ScrollView розташовано ключові аналітичні компоненти: діаграма розподілу настроїв ResponsePieChartView, узагальнювальний модуль OverallSentimentSectionView, а також список індивідуальних відповідей користувача у вигляді карток ResponseRowView. Для останніх застосовується анімація появи з одночасним зміщенням та зміною прозорості, що підвищує динамічність інтерфейсу. Завершальним елементом виступає панель введення inputBar, яка відповідає за додавання

нових текстових фрагментів для аналізу та забезпечує безперервний цикл взаємодії користувача із системою. Завдяки такій структурі екран залишається візуально збалансованим, інформативним та зручним у використанні.

Лістинг 4.15 – Програмний код, що реалізує інтерфейс екрана миттєвого аналізу настрою

```
var body: some View {
        NavigationStack {
            ZStack {
              backgroundGradient.ignoresSafeArea()
                VStack(spacing: 20) {
                    ScrollView(showsIndicators: false) {
                        VStack(spacing: 24) {
ResponsePieChartView(responses: responses)
OverallSentimentSectionView(responses: responses)
                            VStack(spacing: 12) {
                                ForEach(responses)       {
response in ResponseRowView(response: response)
.transition(.opacity.combined(with: .move(edge: .bottom)))
                                }
                            }
                            .padding(.horizontal)
                        }
                        .padding(.top, 12)
                    }
                    inputBar
                }
                .padding(.bottom, 18)
                .navigationTitle("Sentiment Analyzer")
                .navigationBarTitleDisplayMode(.large)
            }
        }
```

4.5.3 Екран журналу настрою

Екран журналу настрою (рисунок 4.6) забезпечує користувача інструментом для щоденного відстеження власного емоційного стану, поєднуючи візуальну простоту з аналітичними можливостями моделі тональності.

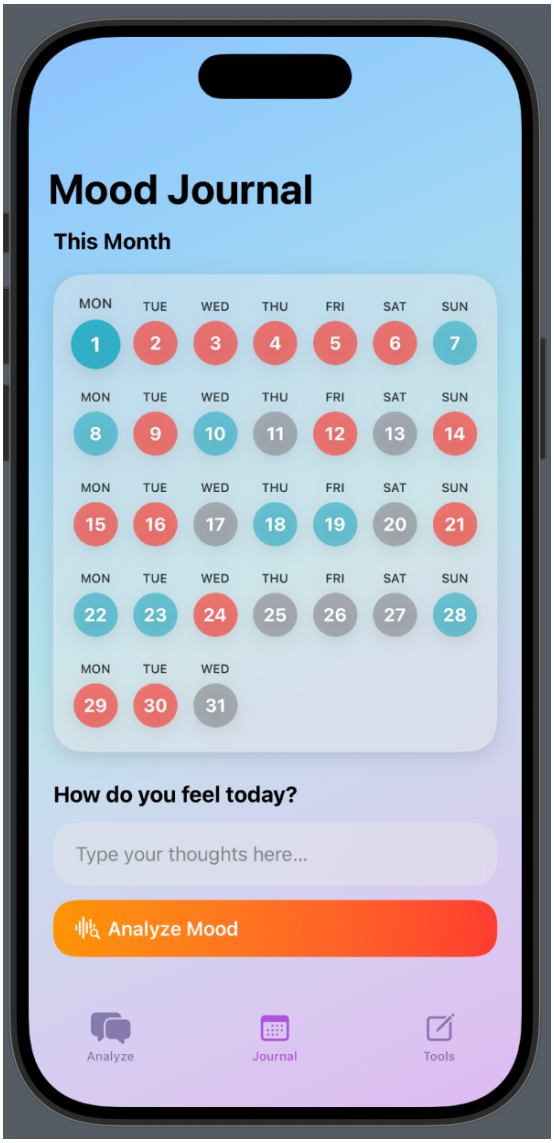

Рисунок 4.6 – Екран журналу настрою

Ключовим елементом інтерфейсу є календар настроїв, який відображає кожен день поточного місяця у вигляді маркерів з кольоровим

кодуванням. Відтінки бірюзового, сірого та червоного сигналізують відповідно про позитивний, нейтральний чи негативний емоційний стан, дозволяючи швидко оцінити динаміку настроїв. Такий підхід надає користувачу інтуїтивний інструмент самоаналізу, де візуальні патерни можуть бути легко помічені без додаткового пояснення.

Нижче розміщується текстове поле для внесення щоденної думки та кнопка запуску аналізу (рисунок 4.7). Після виконання аналізу користувач отримує зрозумілий результат у вигляді індикатора настрою, числового значення загальної оцінки та короткого пояснення. Карта результатів оформлена у стилі застосунку: контрастна, але ненав’язлива панель з чітким типографічним поданням висновку моделі та прогрес-баром, що відображає інтенсивність емоційного тону.

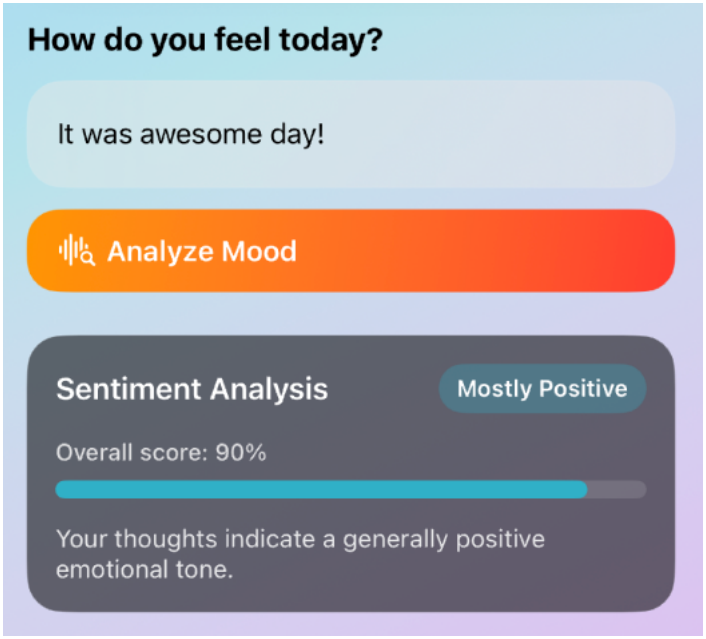

Рисунок 4.7 – Секція для аналізу настрою

Таким чином екран журналу настрою виконує наступні дві ключові ролі. По-перше, він допомагає користувачу зручно реєструвати власні емоції упродовж місяця. По-друге, завдяки інтегрованій моделі аналізу тональності, екран надає об’єктивний зворотний зв’язок щодо того, як

змінюється емоційна динаміка з часом. Це робить модуль не лише засобом самоспостереження, а й інструментом для формування усвідомленості щодо власного психологічного стану.

Програмний код, поданий у лістингу 4.16, реалізує структуру інтерфейсу екрана журналу настрою, поєднуючи календарну візуалізацію, поле введення щоденної думки та блок відображення результатів аналізу. Основою компонування виступає NavigationStack, який забезпечує стандартну навігаційну поведінку iOS та дозволяє інтегрувати екран у загальну структуру застосунку. Поверх нього розташований ZStack із градієнтним фоном, що стилістично узгоджується з іншими екранами системи та створює візуальну цілісність між модулями.

Основний вміст організовано у ScrollView, який містить три ключові секції: календар місяця, форму введення настрою та область результатів аналізу. Така структуризація відповідає логіці роботи користувача: спочатку він бачить загальну динаміку настроїв, потім може внести власний текст, а після виконання аналізу переглянути оцінку моделі. Виклик onAppear встановлює поточний день за замовчуванням, забезпечуючи коректну ініціалізацію інтерфейсу при першому завантаженні екрана. Таким чином, реалізація, наведена в лістингу 4.16, демонструє послідовно організований, інтуїтивний і функціональний інтерфейс модуля Mood Journal.

Лістинг 4.16 – Програмний код, що реалізує екран журналу настрою

```
var body: some View {
        NavigationStack {
            ZStack {
                backgroundGradient.ignoresSafeArea()
                VStack{
                    ScrollView(showsIndicators: false) {
                        VStack{
                            monthlyCalendarSection
                            todayInputSection
                            sentimentSection
```

Продовження лістингу 4.16

```
                        }
                        .padding(.horizontal, 20)
                        .padding(.vertical, 4)
                    }
                }
                .padding(.bottom, 18)
                .navigationTitle("Mood Journal")
                .navigationBarTitleDisplayMode(.large)
            }
            .onAppear {
                if selectedDayId == nil {
                    selectedDayId = monthDays.first?.id
                }
            }
        }
    }
```

4.5.4 Екран інструментів роботи з текстом

Екран інструментів роботи з текстом забезпечує користувачеві розширені можливості взаємодії з текстовими даними, що виходять за межі стандартного аналізу настрою. На цьому екрані зібрані функції імпорту, обробки та експорту текстів, що дозволяє використовувати застосунок як універсальний інструмент для генерації PDF-звітів, створення графічних матеріалів та швидкого копіювання або вставлення тексту з буфера обміну.

Можна побачити загальну структуру інтерфейсу (рисунок 4.8), де кожна дія подана у вигляді окремої картки-кнопки з чіткою назвою, піктограмою та коротким підписом, що підвищує зрозумілість. Такий підхід дозволяє користувачу швидко зорієнтуватися у доступних інструментах та інтуїтивно обрати потрібну дію.

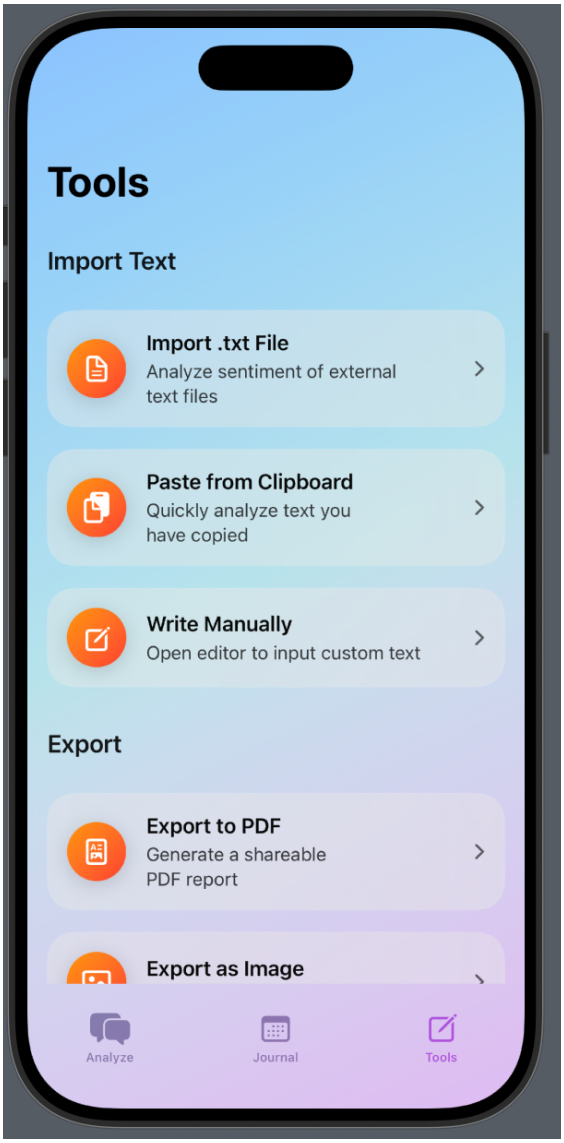

Рисунок 4.8 – Екран інструментів роботи з текстом

У лістингу 4.17 подано спрощений програмний код, який реалізує інтерфейс екрана Tools. У цьому прикладі використано NavigationStack як базовий навігаційний контейнер, а вміст екрана сформовано за допомогою ScrollView та вертикального стеку елементів. Кожна кнопка викликає певну функціональність: імпорт текстового файлу, вставлення тексту з буфера, відкриття редактора для ручного введення, генерацію PDF або графічного зображення. Така реалізація забезпечує модульність, а також дозволяє легко розширювати або змінювати функціональність у межах цього модуля.

Лістинг 4.17 – Програмний код, що реалізує інтерфейс екрана Tools

```
NavigationStack {
    VStack {
        ScrollView {
            VStack(spacing: 20) {
                Button("Import .txt File")
                Button("Paste from Clipboard")
                Button("Write Manually"
                Button("Export to PDF") { generatePDF() }
                Button("Export as Image") { generateImage()
                Button("Share") { showShareSheet = true }
            }
            .padding()
        }
    }
    .navigationTitle("Tools")
}
```

Завершуючи опис реалізації користувацького інтерфейсу, слід підкреслити, що всі екрани застосунку побудовані відповідно до єдиної візуальної концепції, яка поєднує сучасну естетику, інформативність та високу зручність користування. Узгоджені кольорові акценти, адаптивні компоненти та продумана логіка взаємодії формують цілісний досвід, у якому кожен модуль – від миттєвого аналізу настрою до журналу та інструментів – органічно доповнює загальний функціонал системи.

Така інтеграція забезпечує безперервність роботи користувача, водночас підкреслюючи ключову мету застосунку – створити простий, надійний та візуально привабливий інструмент для аналізу й відстеження емоційного стану.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було розроблено мобільний застосунок Sentiment Analyzer, призначений для аналізу емоційної тональності тексту та відстеження змін настрою користувача у динаміці. Робота повністю відповідає поставленим завданням: створено три основні функціональні модулі – Real-Time Sentiment Analyzer, Mood Journal та Tools, кожен з яких забезпечує окремий аспект аналізу текстових даних та емоційної взаємодії. Реалізовано як локальний аналіз тональності на основі бібліотеки NaturalLanguage, так і можливість інтеграції власної машинної моделі, що забезпечує розширюваність системи. Окрім того, створено сучасний, адаптивний інтерфейс, орієнтований на забезпечення інтуїтивності та зручності для користувача. Застосунок характеризується високою стабільністю роботи, швидкою обробкою тексту та зрозумілими механізмами візуалізації даних, що підтверджує досягнення якісних показників, визначених у межах проєкту.

З точки зору кількісних характеристик робота демонструє ефективність алгоритмів обчислення тональності, здатність обробляти довгі текстові фрагменти та забезпечувати миттєву реакцію системи на введення інформації. Якісні показники відображаються у можливості чітко класифікувати емоційні індикатори, формувати узагальнений емоційний профіль, а також вести журнал настроїв з прив’язкою до календарної структури. Система забезпечує комплексний огляд емоційної динаміки, що дозволяє користувачеві отримувати не лише окремі результати аналізу, а й відстежувати тенденції та прогрес у власному психоемоційному стані.

Порівняння розробленого застосунку з аналогами, присутніми на ринку, показує, що Sentiment Analyzer поєднує традиційні можливості аналізу тональності з додатковими функціями, які не завжди представлені в інших рішеннях. На відміну від багатьох мобільних застосунків, що виконують лише базовий аналіз настрою, створена система пропонує три

модулі, які охоплюють різні сценарії використання: миттєві емоційні реакції, довготривале відстеження настрою та професійні інструменти для роботи з текстовим контентом. Інтеграція журналу настрою з календарною моделлю підвищує унікальність застосунку, а можливість експорту результатів у вигляді PDF/PNG розширює сферу його використання – від саморефлексії до професійної діяльності маркетологів, SMM-фахівців і контент-редакторів.

Перспективи подальшої роботи в даному напрямку є широкими та різноманітними. Одним із ключових напрямів розвитку може стати інтеграція дизайну поведінкової аналітики, яка дозволяла б виявляти довгострокові патерни настроїв та формувати рекомендації щодо покращення емоційного стану. Також доцільно розширити функціональність машинного навчання – наприклад, додати можливість багатокласової класифікації емоцій (радість, смуток, гнів, здивування тощо), впровадити модель для оцінки емоційного інтенсиву або додати надійніші методи обробки великих текстових масивів. Перспективним є створення синхронізації з хмарними сервісами, що дозволить вести історію настроїв на різних пристроях та виконувати резервне копіювання даних. Додаткові можливості відкриває інтеграція з носимими пристроями, такими як Apple Watch: аналіз пульсу, фізичної активності чи режиму сну може значно збагачувати контекст емоційних записів і підвищувати точність оцінок.

У підсумку розроблений застосунок повністю виконує поставлену мету, забезпечує високий рівень функціональності та пропонує інноваційні можливості для аналізу текстових емоційних індикаторів. Він є конкурентоздатним порівняно з аналогами, а запропоновані напрями розвитку відкривають можливість створення ще більш комплексного та інтелектуального інструменту для моніторингу психоемоційного стану користувачів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Bird S., Klein E., Loper E. Natural Language Processing with Python. Sebastopol : O'Reilly Media, 2009. 504 p.

2. Brownlee J. Deep Learning for Natural Language Processing. Machine Learning Mastery, 2019. 276 p.

3. Chollet F. Deep Learning with Python. Shelter Island : Manning Publications, 2021. 504 p.

4. Goldberg Y. Neural Network Methods for Natural Language Processing. San Rafael : Morgan & Claypool, 2017. 184 p.

5. Apple Inc. Core ML Documentation. URL: https://developer.apple.com/documentation/coreml (date of access: 20.10.2025).

6. Apple Inc. Core ML Models – Machine Learning. URL: https://developer.apple.com/machine-learning/models/ (date of access: 01.11.2025).

7. Apple Inc. Core ML Tools API Reference. URL: https://apple.github.io/coremltools/ (date of access: 27.10.2025).

8. Apple Inc. Create ML Documentation. URL: https://developer.apple.com/documentation/createml (date of access: 22.10.2025).

9. Apple Inc. Get started with SwiftUI. URL: https://developer.apple.com/swiftui/get-started/ (date of access: 30.10.2025).

10. Apple Inc. SwiftUI Documentation. URL: https://developer.apple.com/documentation/SwiftUI (date of access: 25.10.2025).

11. Apple Inc. SwiftUI Tutorials – Develop in Swift. URL: https://developer.apple.com/tutorials/develop-in-swift/welcome-to-swiftui (date of access: 28.10.2025).

12. Jurafsky D., Martin J. H. Speech and Language Processing. 2023. 1300 p.

13. Kreitschmann D. User Manual for the Apple CoreCapture Framework. 2018.

14. Runtime O. CoreML Execution Provider. URL: https://onnxruntime.ai/docs/execution-providers/CoreML-ExecutionProvider.html (date of access: 03.11.2025).

15. Tveit A., Morland T., Røst T. B. DeepLearningKit – GPU Optimized Deep Learning Framework for iOS, OS X and tvOS. 2016.

16. *Pattern Recognition*. Machine Learning. 2021. URL: https://doi.org/10.7551/mitpress/13811.003.0006 (date of access: 01.11.2025).

17. Practical Natural Language Processing: A Comprehensive Guide to Building Real-World NLP Systems / S. Vajjala et al. O'Reilly Media, 2020. 325 p.

18. Sabharwal N., Agrawal A. BERT Algorithms Explained. *Hands-on Question Answering Systems with BERT*. Berkeley, CA, 2021. P. 65–95. URL: https://doi.org/10.1007/978-1-4842-6664-9_4 (date of access: 01.11.2025).

19. Wolf T., Tunstall L., Werra L. v. Natural Language Processing with Transformers. O'Reilly Media, Incorporated, 2022.

20. Zhao Y. The State-of-art Applications of NLP: Evidence from ChatGPT. *Highlights in Science, Engineering and Technology*. 2023. Vol. 49. P. 237–243. URL: https://doi.org/10.54097/hset.v49i.8512 (date of access: 01.11.2025).