Міністерство освіти і науки України
Харківський національний університет імені В. Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки

До захисту допущено
Кафедрою комп'ютерних систем та робототехніки
протокол № __ від __ грудня 2025р.

завідувач кафедри _____ Максим ХРУСЛОВ
(підпис)

«___»_____ 2025 р.

Кваліфікаційна робота

здобувача другого (магістерського) рівня вищої освіти

«НЕЙРОМЕРЕЖЕВА МОДЕЛЬ ДЛЯ АНАЛІЗУ ЕМОЦІЙНОГО ЗАБАРВЛЕННЯ ТЕКСТУ»

Спеціальність ***174 – Автоматизація, комп'ютерно-інтегровані технології та робототехніка***
Освітня програма ***Комп'ютеризовані системи управління та автоматика***

Виконавець _____ **Д. І. ГОЛОЦВАН**
(підпис)

Науковий керівник _____ **О. І. ЧУБ**
(підпис)

Харків – 2025

АНОТАЦІЯ

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи складається зі вступу, трьох розділів, висновків, списку джерел та додатків. Загальний обсяг роботи складає 75 сторінки, із яких 60 сторінки основної частини з 10 рисунками, 4 таблицями, 28 найменувань списку використаних джерел та трьома додатками.

Метою кваліфікаційної роботи є забезпечення ефективної та оперативної класифікації емоційного забарвлення текстових повідомлень через створення і впровадження моделі автоматизованої системи на основі згорткової нейронної мережі та алгоритму адаптивного прийняття рішень.

Об'єкт дослідження – процес автоматизованої обробки неструктурованої текстової інформації в інформаційно-керуючих системах.

Предмет дослідження – математичні моделі глибокого навчання (TextCNN), методи попередньої обробки природної мови (NLP) та алгоритми оптимізації порогів класифікації для роботи з незбалансованими даними.

Проблема, яка вирішується в кваліфікаційній роботі, полягає в тому, щоб підвищити точність детектування рідкісних емоційних станів в умовах «шумних» коротких текстів та дисбалансу класів, мінімізувати час інференсу моделі для роботи в реальному часі та забезпечити гнучке налаштування чутливості системи (баланс Precision/Recall).

Область застосування – системи моніторингу соціальних мереж, автоматизована модерація контенту, інтелектуальні служби підтримки клієнтів (HelpDesk), управління репутацією бренду. Розроблений програмний модуль може використовуватися як мікросервіс у складі CRM-систем.

***Ключові слова**: АНАЛІЗ ЕМОЦІЙ, БАГАТОМІТКОВА КЛАСИФІКАЦІЯ, TEXTCNN, NLP, GOEMOTIONS, PYTHON, KERAS, АДАПТИВНІ ПОРОГИ, MACRO-F1.*

ABSTRACT

The explanatory note to the qualification paper consists of an introduction, three chapters, conclusions, a list of references, and appendices. The total volume of the work is 75 pages, of which 60 pages are the main part with 10 figures, 4 tables, 28 items in the list of references, and three appendices.

The aim of the qualification paper is to ensure effective and rapid classification of the emotional sentiment of text messages through the creation and implementation of an automated system model based on a Convolutional Neural Network and an adaptive decision-making algorithm.

The object of research is the process of automated processing of unstructured text information in information and control systems.

The subject of research is deep learning mathematical models (TextCNN), Natural Language Processing (NLP) preprocessing methods, and classification threshold optimization algorithms for working with imbalanced data.

The problem solved in the qualification paper is to improve the detection accuracy of rare emotional states under conditions of "noisy" short texts and class imbalance, minimize model inference time for real-time operation, and ensure flexible tuning of system sensitivity (Precision/Recall balance).

The field of application covers social media monitoring systems, automated content moderation, intelligent customer support services (HelpDesk), and brand reputation management. The developed software module can be used as a microservice within CRM systems.

Keywords: *EMOTION ANALYSIS, MULTI-LABEL CLASSIFICATION, TEXTCNN, NLP, GOEMOTIONS, PYTHON, KERAS, ADAPTIVE THRESHOLDS, MACRO-F1.*

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

API – Application Programming Interface (інтерфейс прикладного програмування)

BCE – Binary Cross-Entropy (функція втрат бінарної перехресної ентропії)

CNN – Convolutional Neural Network (згорткова нейронна мережа)

CRM – Customer Relationship Management (система управління взаємовідносинами з клієнтами)

GPU – Graphics Processing Unit (графічний процесор для прискорення обчислень)

GRU – Gated Recurrent Unit (керований рекурентний блок)

HCI – Human-Computer Interaction (людино-машинна взаємодія)

JSON – JavaScript Object Notation (текстовий формат обміну даними)

LR – Logistic Regression (логістична регресія)

LSTM – Long Short-Term Memory (довга короткочасна пам’ять)

ML – Machine Learning (машинне навчання)

NLP – Natural Language Processing (обробка природної мови)

OvR – One-vs-Rest (стратегія класифікації «один проти всіх»)

PR-AUC – Precision-Recall Area Under Curve (площа під кривою «точність-повнота»)

ReLU – Rectified Linear Unit (випрямлена лінійна функція активації)

ROC-AUC – Receiver Operating Characteristic Area Under Curve (площа під робочою характеристикою приймача)

RNN – Recurrent Neural Network (рекурентна нейронна мережа)

TextCNN – Convolutional Neural Networks for Sentence Classification (архітектура згорткової мережі для класифікації тексту)

TF-IDF – Term Frequency-Inverse Document Frequency (статистична міра оцінки важливості слова в документі)

СППР – Система підтримки прийняття рішень

ВСТУП

У зв’язку з глобальною інформатизацією суспільства та лавиноподібним зростанням обсягів неструктурованих текстових даних (User-Generated Content) у мережі Інтернет, все більш актуальною постає проблема переходу від ручної обробки інформації до автоматизованих інтелектуальних рішень. В умовах високої конкуренції та репутаційних ризиків ключовим фактором успіху для бізнесу та соціальних платформ стає здатність оперативно та точно розуміти емоційний стан користувачів, виражених у коментарях, відгуках та повідомленнях.

Актуальність роботи. Серед прикладних областей, де зазначена проблема є вкрай гострою, слід віднести автоматизовану модерацію контенту, аналітику клієнтського досвіду (Customer Experience) та системи підтримки користувачів. Специфіка цієї галузі полягає у необхідності обробки коротких, «шумних» повідомлень, що містять сленг, сарказм та змішані емоції. Традиційні методи аналізу, які використовуються в більшості систем (пошук за ключовими словами або бінарний сентимент-аналіз «позитив/негатив»), демонструють низьку надійність в умовах складної семантики природної мови. Вони не здатні виявити тонкі емоційні відтінки (наприклад, розрізнити злість та роздратування) або адаптуватися до дисбалансу класів, що призводить до пропуску критичних сигналів. Водночас сучасні важкі трансформерні моделі (BERT) часто є занадто ресурсоємними для інтеграції в системи реального часу. Отже, задача розробки автоматизованої системи, яка поєднує точність згорткових нейронних мереж (TextCNN) із швидкодією та інженерною адаптивністю (через оптимізацію порогів), є вкрай актуальною науково-прикладною проблемою.

Метою дослідження є забезпечення ефективної та оперативної класифікації емоційного забарвлення текстових повідомлень через створення і впровадження моделі автоматизованої системи на основі згорткової нейронної мережі та алгоритму адаптивного прийняття рішень.

Об’єкт дослідження – це процеси автоматизованої обробки неструктурованої текстової інформації в інформаційно-керуючих системах в умовах лінгвістичної невизначеності.

Методи дослідження: методи системного аналізу, математичне моделювання (для векторизації тексту TF-IDF та Embeddings), методи глибокого навчання (архітектури CNN, TextCNN), методи теорії ймовірностей та оптимізації (для адаптивного налаштування порогів класифікації), методи оцінки якості класифікаторів (метрики Micro/Macro F1-score, Precision-Recall).

Предмет дослідження – моделі, алгоритми та програмні засоби побудови нейромережевих систем багатоміткової класифікації тексту та їх адаптації до незбалансованих наборів даних.

Завдання дослідження:

1. Виконати аналіз існуючого науково-методичного апарата обробки природної мови (NLP) та обґрунтувати переваги використання згорткових нейронних мереж (TextCNN) для аналізу коротких текстів порівняно з класичними лінійними методами.
2. Розробити методику попередньої обробки «шумних» даних та архітектуру нейронної мережі, що враховує локальні контекстні залежності (n-грами) у текстових повідомленнях.
3. Спроектувати структуру автоматизованої системи та розробити алгоритм адаптивної оптимізації порогів прийняття рішень (Threshold Tuning) для мінімізації помилок класифікації рідкісних емоційних станів.
4. Виконати програмну реалізацію системи та провести експериментальне порівняння ефективності розробленої нейромережевої моделі TextCNN із базовим методом (Logistic Regression) на реальних даних GoEmotions, оцінивши їх точність та придатність до практичного впровадження.

РОЗДІЛ 1.
ТЕОРЕТИЧНІ ОСНОВИ ТА ВИБІР МЕТОДІВ АНАЛІЗУ ЕМОЦІЙНОГО ЗАБАРВЛЕННЯ ТЕКСТІВ

1.1. Постановка задачі автоматизованої класифікації емоцій у "шумних" текстових даних.

Стрімкий розвиток інформаційно-комунікаційних технологій призвів до експоненційного зростання обсягів неструктурованих текстових даних, що генеруються користувачами в мережі Інтернет. Соціальні мережі, форуми, системи відгуків та чати технічної підтримки створюють безперервний потік інформації, ручна обробка якої стає неможливою. У цьому контексті задача автоматизованого аналізу емоційного забарвлення тексту (Emotion Analysis) трансформується з суто лінгвістичної проблеми в актуальну інженерну задачу побудови ефективних систем автоматичного керування інформаційними потоками [1].

На відміну від класичного сентимент-аналізу (Sentiment Analysis), який зазвичай обмежується бінарною (позитивний/негативний) або тринарною (плюс нейтральний) шкалою оцінки, аналіз емоцій передбачає класифікацію тексту за більш гранулярною шкалою дискретних психоемоційних станів.

Для розуміння місця досліджуваної задачі в загальній ієрархії методів штучного інтелекту доцільно розглянути таксономію задач обробки природної мови.

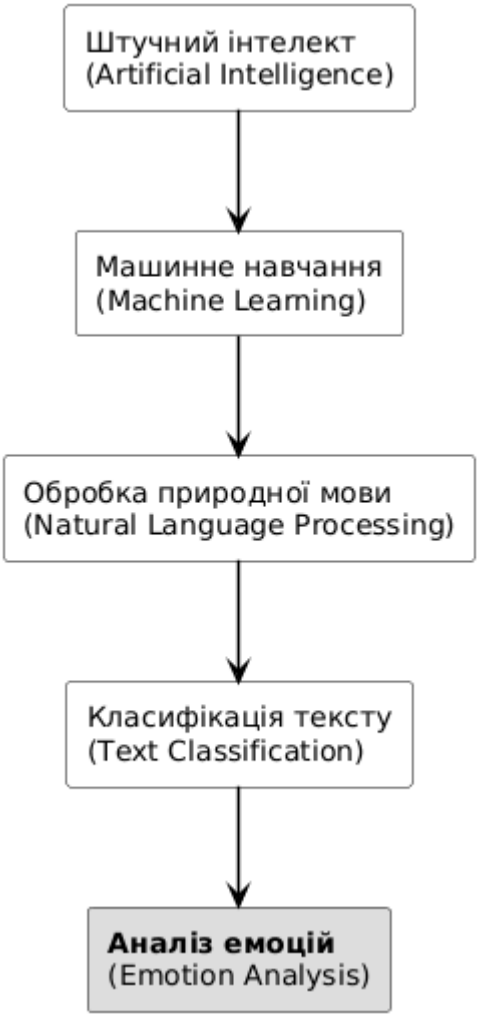

Рисунок 1.1 — Місце задачі емоційного аналізу в ієрархії технологій штучного інтелекту.

Як видно, задача класифікації емоцій є підмножиною задач класифікації тексту, однак має специфічні особливості, зумовлені природою вхідних даних та складністю формалізації емоційного простору.

З точки зору теорії автоматичного керування, система аналізу емоцій виступає як інтелектуальний датчик, що перетворює вхідний неструктурований сигнал (текст повідомлення x) у формалізований вектор стану (y), придатний для подальшої обробки в контурі управління (наприклад, для пріоритезації тікетів у службі підтримки або автоматичної модерації контенту).

Ключовою проблемою при проектуванні таких систем є робота з так званими «шумними» даними. Тексти в соціальних мережах та месенджерах характеризуються:

1. Високим рівнем граматичних та орфографічних помилок.
2. Використанням сленгу, абревіатур та неологізмів.
3. Наявністю графічних символів (емодзі), які несуть значне семантичне навантаження.
4. Коротким розміром повідомлень, що обмежує контекст, необхідний для визначення емоції.

Формально задачу дослідження можна визначити як задачу багатоміткової класифікації (Multi-label Classification). Нехай X — простір вхідних текстових послідовностей, а $L = \{l_1, l_2, \ldots, l_K\}$ — множина можливих емоційних міток, де K — кількість емоцій (для набору даних GoEmotions $K = 28$, включаючи нейтральний стан) [16]. Необхідно побудувати відображення $f: X \to \{0,1\}^K$, яке ставить у відповідність кожному тексту $x \in X$ бінарний вектор $y = (y_1, \ldots, y_K)$, де $y_i = 1$, якщо i-та емоція присутня в тексті, і $y_i = 0$ в іншому випадку.

Складність цієї постановки полягає в тому, що класи не є взаємовиключними. Одне повідомлення може одночасно містити, наприклад, «радість» та «здивування», або «сум» та «розчарування». Це вимагає відмови від традиційних функцій активації типу Softmax, що використовуються в однокласовій класифікації, на користь незалежних активацій (Sigmoid) для кожного класу.

Окремим викликом є інтеграція таких моделей у системи підтримки прийняття рішень (СППР). Як зазначають дослідники [9], ефективність СППР на основі аналізу тексту критично залежить не лише від точності моделі, але й від її здатності коректно обробляти невизначеність та дисбаланс класів, коли кількість прикладів для різних емоцій суттєво різниться.

Таким чином, розробка нейромережевої моделі для цієї задачі вимагає комплексного підходу, що включає спеціальні методи передобробки «шумного» тексту, вибір архітектури, здатної виділяти локальні ознаки на коротких дистанціях, та застосування алгоритмів адаптації порогів прийняття рішень для нівелювання ефекту дисбалансу даних.

1.2. Огляд існуючих методів та алгоритмів NLP: від словникових підходів до трансформерів.

Історія розвитку методів обробки природної мови (NLP) демонструє еволюцію від жорстких правил до ймовірнісних моделей та глибокого навчання. Для задачі автоматизованої класифікації емоцій у текстах можна виділити три основні парадигми: словникові методи, класичне машинне навчання та нейромережеві архітектури [1].

Словникові та правилові методи (Lexicon-based)

Найпростішим підходом до аналізу емоційного забарвлення є використання лексиконів — заздалегідь складених словників, де кожному слову (токену) присвоєно певну емоційну вагу або мітку. Прикладами таких ресурсів є *NRC Emotion Lexicon* або *VADER*. Алгоритм роботи таких систем зводиться до пошуку слів у тексті та агрегації їх ваг [4].

- **Переваги:** Висока інтерпретованість результатів, відсутність необхідності у навчанні (training-free), швидкість роботи.
- **Недоліки:** Нездатність враховувати контекст, сарказм, заперечення (наприклад, фраза «не дуже добре» може бути класифікована як позитивна через слово «добре») та багатозначність слів (полісемію).

У контексті сучасних автоматизованих систем, що працюють з «шумними» даними соціальних мереж, ефективність чистих словникових методів є недостатньою [8], оскільки сленг та неологізми з'являються швидше, ніж оновлюються словники.

Класичне машинне навчання (Classical Machine Learning)

Наступним етапом розвитку стало використання статистичних алгоритмів навчання з учителем. Цей підхід розділяє процес на два етапи: вилучення ознак (feature extraction) та класифікацію.

Для представлення тексту у вигляді числового вектора найчастіше використовується модель «Мішка слів» (Bag-of-Words) або її модифікація **TF-IDF** (Term Frequency-Inverse Document Frequency). Метод TF-IDF дозволяє оцінити важливість слова в контексті документа відносно всього корпусу даних [11]. Вага $w_{\{i,j\}}$ слова i в документі j обчислюється як:

$$w_{\{i,j\}} = tf_{\{i,j\}} \times log(\frac{N}{df_i}),$$

де $tf_{\{i,j\}}$ — частота слова в документі, N — загальна кількість документів, df_i — кількість документів, що містять слово i.

У поєднанні з лінійними класифікаторами, такими як **Логістична регресія (Logistic Regression)** або Метод опорних векторів (SVM), TF-IDF демонструє напрочуд високу ефективність на задачах класифікації текстів [24]. Саме тому в рамках даного дослідження модель **Logistic Regression + TF-IDF** обрана як базовий метод (baseline) для порівняння. Її перевагою є простота реалізації та "прозорість" прийняття рішень, що є критичним для інженерних задач автоматизації.

Нейромережеві методи та векторні представлення (Deep Learning)

Революція в NLP пов’язана з появою щільних векторних представлень слів (Word Embeddings), таких як Word2Vec [23] та GloVe [25]. На відміну від розріджених векторів TF-IDF, ембеддінги кодують семантичну близькість слів у багатовимірному просторі, що дозволяє моделям розуміти синоніми та аналогії.

Серед архітектур глибокого навчання для класифікації тексту виділяють два основні класи:

1. **Рекурентні нейронні мережі (RNN/LSTM/GRU).** Ці мережі обробляють текст послідовно, слово за словом, зберігаючи інформацію про попередній контекст у прихованому стані. Хоча LSTM ефективно вирішують проблему зникаючого градієнта на довгих послідовностях [15], їхнім суттєвим недоліком є неможливість розпаралелювання обчислень, що сповільнює роботу системи в реальному часі.
2. **Згорткові нейронні мережі (CNN).** Спочатку розроблені для комп'ютерного зору, CNN були успішно адаптовані для NLP [20]. У текстових CNN одновимірні фільтри (Kernels) "ковзають" по реченню, виявляючи локальні патерни — n-грами (стійкі сполучення з 2, 3 або більше слів).

Для задачі аналізу коротких повідомлень (коментарів, твітів) архітектура **TextCNN** часто перевершує RNN, оскільки емоційне забарвлення в таких текстах зазвичай визначається локальними ключовими фразами (наприклад, «я так радий», «жахливий сервіс»), а не глобальною структурою довгого документа [27]. Крім того, операції згортки легко розпаралелюються на GPU, що робить TextCNN привабливим вибором для побудови високошвидкісних автоматизованих систем моніторингу [5].

Трансформери (Transformers)

Сучасним стандартом у NLP є архітектура Transformer (BERT, RoBERTa, GPT), яка базується на механізмі самонавчання уваги (Self-Attention) [26]. Трансформери дозволяють моделювати складні залежності між усіма словами в реченні одночасно [17].

Попри найвищу точність (State-of-the-Art), використання трансформерів у промислових системах автоматизації пов'язане з низкою проблем:

- **Висока обчислювальна складність:** Моделі типу BERT містять сотні мільйонів параметрів, що вимагає потужних апаратних ресурсів для інференсу.
- **Затримка (Latency):** Час обробки одного запиту може бути неприйнятним для систем, що працюють у режимі реального часу з великим потоком даних.

Враховуючи вищезазначене, для проектування системи аналізу емоцій у даній кваліфікаційній роботі доцільно зосередитися на порівнянні ефективного класичного методу (Logistic Regression) та швидкої нейромережевої архітектури (TextCNN), яка забезпечує баланс між точністю та швидкодією [14].

1.3. Аналіз особливостей багатоміткової (multi-label) класифікації та проблеми дисбалансу класів.

Специфіка задачі визначення емоційного стану за текстом полягає в тому, що емоції рідко виступають у чистому, ізольованому вигляді. Психологічні дослідження показують, що людина часто відчуває змішані почуття (наприклад, *«joy»* та *«relief»* або *«sadness»* та *«disappointment»* одночасно) [16]. Це фундаментально відрізняє задачу даного дослідження від класичної багатокласової (multi-class) класифікації.

Математична формалізація багатоміткової постановки

У традиційній **багатокласовій класифікації** (Multi-class) припускається, що кожен об'єкт x належить рівно до одного класу з множини L. Для моделювання ймовірнісного розподілу в таких задачах використовується функція **Softmax**, яка нормалізує вихідний вектор так, щоб сума всіх ймовірностей дорівнювала одиниці:

$$Softmax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_i}}$$

де z — вихідний вектор логітів нейромережі. Це створює конкуренцію між класами: збільшення ймовірності одного класу автоматично зменшує ймовірності інших.

Натомість, у **багатомітковій класифікації** (Multi-label), яка розглядається в даній роботі, класи є незалежними (або умовно незалежними). Присутність мітки *«admiration»* не виключає присутності мітки *«gratitude»*. Тому замість Softmax застосовується функція активації **Sigmoid** для кожного нейрона вихідного шару окремо [14]:

$$\sigma(z_i) = \frac{1}{1 + e^{-z_i}}$$

Це дозволяє моделі передбачати незалежну ймовірність $P(y_i = 1|x)$ для кожної i-ї емоції в діапазоні $[0, 1]$.

Відповідно змінюється і функція втрат (Loss Function). Замість категоріальної перехресної ентропії (Categorical Cross-Entropy) використовується **бінарна перехресна ентропія (Binary Cross-Entropy - BCE)**, усереднена по всіх K класах:

$$L_{\{BCE\}} = -\frac{1}{K}\sum_{i=1}^{K}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

де y_i — істинна мітка (0 або 1), а $\hat{y}_i$ — передбачена ймовірність.

Проблема дисбалансу класів (Class Imbalance)

Однією з найгостріших проблем при автоматизації обробки природної мови є нерівномірний розподіл класів у реальних даних. Це явище відоме як розподіл «довгого хвоста» (Long-tail distribution).

У наборі даних **GoEmotions**, який використовується в експериментальній частині, спостерігається суттєвий дисбаланс [16]:

1. **Мажоритарні класи:** Емоція *«Neutral»* складає близько 30% всіх записів, також високочастотними є *«Admiration»* та *«Approval»*.
2. **Міноритарні класи:** Такі емоції як *«Grief»* (горе), *«Relief»* (полегшення) або *«Nervousness»* (знервованість) зустрічаються вкрай рідко (менше 1% вибірки).

З точки зору теорії навчання машин, такий дисбаланс призводить до того, що стандартний алгоритм оптимізації (наприклад, градієнтний спуск) "лінується": моделі вигідніше завжди передбачати відсутність рідкісної емоції. Це дозволяє досягти високої загальної точності (Accuracy), але робить систему функціонально непридатною для виявлення критичних станів.

Інженерна аналогія: Уявімо систему автоматичної пожежної безпеки, яка у 99.9% часу працює в умовах відсутності пожежі. Якщо модель навчиться завжди видавати сигнал "Пожежі немає", її точність буде 99.9%, але її корисність дорівнюватиме нулю, оскільки вона пропустить реальну небезпеку.

Вплив на вибір стратегії прийняття рішень

Для вирішення проблеми дисбалансу в системах автоматизації застосовують різні підходи:

- **Resampling:** Штучне збільшення кількості прикладів рідкісних класів (Oversampling) або зменшення частих (Undersampling). Проте це може призвести до перенавчання (overfitting) або втрати важливої інформації.
- **Cost-sensitive learning:** Введення вагових коефіцієнтів у функцію втрат, що збільшує "штраф" за помилку на рідкісному класі.
- **Адаптивне налаштування порогів (Threshold Tuning).**

У даній роботі пропонується зосередитися саме на **оптимізації порогів** [21]. Стандартний поріг класифікації $t = 0.5$ (якщо $\hat{y}_i > 0.5$, то клас активний) є оптимальним лише для збалансованих вибірок. Для рідкісних емоцій модель

може видавати низькі ймовірності (наприклад, 0.3), навіть коли вона "впевнена" у наявності ознак цього класу більше, ніж зазвичай.

Отже, задача проектування системи (Розділ 2) повинна включати розробку алгоритму пошуку вектора індивідуальних порогів $T = \{t_1, t_2, \ldots, t_K\}$, які максимізують цільову метрику якості (F1-score) для кожного класу окремо. Це дозволить значно підвищити чутливість системи (Recall) до рідкісних, але важливих емоційних станів, забезпечуючи надійність функціонування всього контуру автоматизованого моніторингу.

1.4. Обґрунтування вибору метрик якості (Micro/Macro F1, PR-AUC) та гіпотези дослідження.

Коректна оцінка ефективності систем автоматизованого аналізу даних є критичною умовою для їх впровадження в контури керування. Для задачі багатоміткової класифікації на незбалансованих даних (таких як GoEmotions) стандартні метрики, зокрема загальна точність (Accuracy), можуть давати хибне уявлення про працездатність системи.

Обмеження метрики Accuracy

Метрика *Subset Accuracy* (або Exact Match Ratio) вимагає повного збігу прогнозованого набору міток з істинним:

$$Accuracy = \frac{1}{N}\sum_{i=1}^{N} I\left(y^{(i)} = \hat{y}^{(i)}\right)$$

У просторі 28 емоцій ймовірність вгадати абсолютно точну комбінацію (наприклад, {*Joy, Optimism*}, але не {*Joy*}) є вкрай низькою. Для інженерних задач часто важливіше «вловити» хоча б одну правильну емоцію, ніж отримати нульову оцінку за часткову помилку. Тому в даній роботі ця метрика використовується лише як допоміжна.

Вибір F1-score: Micro vs Macro усереднення

Основним інструментом оцінки обрано **F1-score** — гармонічне середнє між точністю (Precision) та повнотою (Recall):

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Для багатоміткової задачі існує два основних способи агрегації цієї метрики, які відображають різні аспекти якості системи [7]:

1. **Micro-F1:** Обчислюється глобально шляхом підрахунку загальної кількості істинно-позитивних (TP), хибно-позитивних (FP) та хибно-негативних (FN) спрацьовувань по всіх класах.
    - *Властивість:* Ця метрика "тяжіє" до мажоритарних класів (Neutral, Admiration). Високий Micro-F1 гарантує, що система добре працює на найчастіших запитах.
2. Macro-F1: Обчислюється як середнє арифметичне F1-показників кожного окремого класу:

$$Macro - F1 = \frac{1}{K}\sum_{i=1}^{K} F1_i$$

    - *Властивість:* Ця метрика дає однакову вагу всім класам, незалежно від кількості прикладів. Це робить її **критично важливою** для оцінки здатності системи виявляти рідкісні емоції (*Grief, Relief*).

Оскільки метою автоматизації є побудова системи, чутливої до всього спектру емоцій, саме **Macro-F1** обрано як ключовий критерій ефективності (Key Performance Indicator).

Додаткові метрики: PR-AUC та Hamming Loss

- **PR-AUC (Area Under the Precision-Recall Curve):** На відміну від ROC-кривих, які можуть бути надмірно оптимістичними на незбалансованих

даних, PR-криві показують реальну залежність точності від повноти при зміні порогу. Високе значення площі під кривою свідчить про стабільність моделі.

- **Hamming Loss:** Показує частку неправильно передбачених бітів у вихідному векторі. Чим менше це значення, тим менше "зайвих" емоцій приписує система і тим менше пропускає потрібних.

Гіпотези дослідження

На основі теоретичного аналізу та специфіки предметної області сформульовано наступні робочі гіпотези, перевірка яких виконується в експериментальній частині (Розділ 3):

- **Гіпотеза 1 (Про ефективність архітектур):** Для коротких, "шумних" текстів соціальних мереж спеціалізована нейромережева модель TextCNN, що навчає векторні представлення слів (Embeddings) "з нуля", забезпечить порівнянну якість (Micro-F1) з сильним лінійним базовим методом (Logistic Regression + TF-IDF), але перевершить його у здатності узагальнювати контекст на рівні n-грам.
- **Гіпотеза 2 (Про адаптивне керування):** Застосування алгоритму індивідуальної оптимізації порогів (Threshold Tuning) для кожного класу дозволить суттєво (більш ніж на 10%) підвищити метрику Macro-F1 порівняно з використанням фіксованого порогу $t = 0.5$, нівелюючи негативний вплив дисбалансу даних на розпізнавання рідкісних емоцій.

Висновки до Розділу 1.

У першому розділі магістерської роботи виконано комплексний аналіз проблеми автоматизованого визначення емоційного забарвлення текстових повідомлень у системах моніторингу та керування контентом.

Основні результати теоретичного дослідження полягають у наступному:

1. **Формалізація задачі.** Обґрунтовано, що в умовах реальних інформаційних потоків (соціальні мережі, чати підтримки) задача аналізу емоцій є задачею **багатоміткової класифікації** (Multi-label Classification). Це вимагає використання архітектур з незалежними вихідними нейронами (Sigmoid) та функцією втрат Binary Cross-Entropy, на відміну від класичних підходів сентимент-аналізу.

2. **Вибір методів.** На основі порівняльного аналізу існуючих підходів NLP встановлено, що для обробки коротких, «шумних» текстів найбільш доцільним є використання **згорткових нейронних мереж (TextCNN)**. Ця архітектура забезпечує ефективне виділення локальних ознак (n-грам) та високу швидкість обробки, що є критичним для систем реального часу. Як базовий метод для порівняння (baseline) обрано логістичну регресію з TF-IDF векторизацією.

3. **Проблема дисбалансу.** Виявлено, що набори даних для емоційного аналізу (зокрема GoEmotions) характеризуються суттєвим дисбалансом класів (Long-tail distribution). Доведено, що використання стандартних метрик (Accuracy) та фіксованих порогів прийняття рішень призводить до ігнорування системою рідкісних, але важливих емоційних станів.

4. **Стратегія вирішення.** Визначено, що ключовим напрямком підвищення ефективності системи є не лише ускладнення архітектури нейромережі, а й впровадження **адаптивного алгоритму налаштування порогів (Threshold Tuning)**. Це дозволить максимізувати цільову метрику **Macro-F1**, забезпечуючи баланс між точністю та повнотою для кожного класу окремо.

Таким чином, у першому розділі сформовано необхідну теоретичну базу та визначено напрямки подальших досліджень. Отримані висновки є основою для **Розділу 2**, в якому буде здійснено проектування математичного забезпечення та структурної схеми автоматизованої системи емоційного аналізу.

РОЗДІЛ 2.
ПРОЕКТУВАННЯ МАТЕМАТИЧНОГО ТА АЛГОРИТМІЧНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ

2.1. Загальна структура автоматизованої системи моніторингу текстових повідомлень.

Проектування автоматизованої системи аналізу емоційного забарвлення базується на принципах модульності та конвеєрної обробки даних (pipeline processing). Система розглядається як підсистема (модуль) у складі більш складного програмно-апаратного комплексу, наприклад, CRM-системи, платформи модерації контенту або аналітичного дашборду соціальних мереж.

Загальна функціональна структура розробленої системи представлена на рисунку 2.1. Вона відображає перетворення вхідного інформаційного сигналу (сирого тексту) у вихідний вектор керування (набір емоційних міток) через послідовність алгоритмічних блоків.

Система складається з чотирьох основних функціональних модулів:

1. **Модуль попередньої обробки даних (Preprocessing Module).** Цей блок відповідає за очищення вхідного потоку від інформаційного шуму. Враховуючи специфіку джерела даних (соціальні мережі, Reddit), на вхід системи подаються рядки $S_{\{in\}}$, які містять URL-адреси, спеціальні символи, теги користувачів та зайві пробіли.
    - *Функція:* $x = Preprocess(S_{\{in\}})$.
    - *Результат:* Нормалізований текст x у нижньому регістрі.
2. **Модуль векторизації (Vectorization / Embedding Module).** Оскільки нейронні мережі та лінійні класифікатори не здатні обробляти текстові дані безпосередньо, цей модуль перетворює нормалізований текст x у числовий вектор (або матрицю) V.

- Для базового методу (LR) використовується перетворення TF-IDF: $V \in R^D$, де D — розмір словника.
- Для нейромережевого методу (TextCNN) використовується токенізація з подальшим перетворенням у щільну матрицю ембеддінгів: $V \in R^{\{L \times E\}}$, де L — довжина послідовності, E — розмірність вектора слова.

3. **Ядро класифікації (Inference Engine).** Центральний елемент системи, що містить навчену математичну модель. На цьому етапі обчислюються ймовірності (confidence scores) приналежності тексту до кожного з K класів емоцій.
   - *Функція:* $\hat{y}_{prob} = Model(V)$.
   - *Вихід:* Вектор ймовірностей $\hat{y}_{prob} = [p_1, p_2, \dots, p_{28}]$, де $p_i \in [0, 1]$.
4. **Модуль прийняття рішень (Decision Making Module).** Це критично важливий блок, розроблений у рамках даного дослідження для вирішення проблеми дисбалансу класів. Замість простого округлення ймовірностей, модуль застосовує вектор індивідуальних порогів T, отриманий на етапі навчання.
   - *Логіка:* $y^i_{\{final\}} = 1, \text{якщо } p_i \geq t_i, \text{інакше } 0$.
   - *Результат:* Бінарний вектор наявних емоцій, готовий для передачі в зовнішні системи керування.

Така архітектура дозволяє гнучко змінювати реалізацію окремих блоків (наприклад, замінити модель LR на TextCNN), не змінюючи загальної логіки взаємодії з зовнішнім середовищем. Вхідні та вихідні інтерфейси системи реалізуються за стандартами REST API, приймаючи запити у форматі JSON.

Нижче наведено структурну схему алгоритму функціонування системи, яка деталізує потоки даних між описаними модулями.

Рисунок 2.1 — Структурна схема автоматизованої системи аналізу емоцій.

Запропонована структура забезпечує виконання вимог щодо швидкодії та масштабованості. Розділення етапів отримання ймовірностей (Inference) та бінаризації (Decision Making) дозволяє налаштовувати чутливість системи (Recall) без необхідності перенавчання "важкої" нейромережевої моделі, змінюючи лише файл конфігурації порогів.

2.2. Математична модель базового методу класифікації (Logistic Regression + TF-IDF).

Для оцінки ефективності складних нейромережевих архітектур необхідно мати надійний еталон (baseline). У якості такого еталону обрано лінійну модель класифікації, побудовану на основі логістичної регресії з використанням статистичних ознак тексту (TF-IDF). Цей вибір зумовлений високою інтерпретованістю моделі та її стійкістю до перенавчання на невеликих наборах даних [24].

Векторне представлення тексту (TF-IDF)

Першим етапом математичного моделювання є перетворення множини текстових повідомлень $D = \{d_1, d_2, \ldots, d_N\}$ у числовий вигляд. Для цього використовується міра **TF-IDF** (Term Frequency — Inverse Document Frequency), яка оцінює важливість слова (терма) t у контексті документа d відносно всього корпусу D.

1. Частота терма (Term Frequency, TF). Визначається як відношення кількості входжень слова t у документ d до загальної кількості слів у цьому документі. Для зменшення впливу дуже частих слів часто застосовують сублінійне масштабування:

$$tf(t,d) = 1 + log(f_{\{t,d\}}),$$

де $f_{\{t,d\}}$ — «сира» частота входження.

2. Обернена частота документа (Inverse Document Frequency, IDF). Цей компонент зменшує вагу загальновживаних слів (сполучників, прийменників), які зустрічаються у більшості документів і не несуть специфічного емоційного навантаження:

$$idf(t,D) = \log \frac{N}{1 + |\{d \in D : t \in d\}|},$$

де N — загальна кількість документів у корпусі, а знаменник — кількість документів, що містять терм t.

3. Фінальна вага. Вектор ознак для документа x формується як добуток цих величин:

$$x_j = tf(t_j, d) \cdot idf(t_j, D).$$

Для підвищення якості моделі на «шумних» даних пропонується використовувати не лише окремі слова (уніграми), але й **n-грами** (послідовності з n слів або символів). У розробленій системі простір ознак формується шляхом об'єднання:

- Слівних уніграм та біграм (word 1-2 grams).
- Символьних три-, чотири- та п'ятиграм (char 3-5 grams).

Це дозволяє врахувати морфологічні особливості (наприклад, суфікси) та стійкі емоційні вирази, навіть якщо вони написані з помилками.

Логістична регресія для задачі класифікації

Математичною основою класифікатора є модель логістичної регресії, яка моделює ймовірність приналежності об'єкта x до класу $y = 1$ за допомогою сигмоїдної функції активації:

$$P(y = 1 \mid x;\ w, b) = \sigma(z) = \frac{1}{1 + e^{-(w^T x + b)}} \quad ,$$

де:

- $x \in R^M$ — вхідний вектор ознак TF-IDF розмірності M;
- $w \in R^M$ — вектор вагових коефіцієнтів моделі;
- $b \in R$ — зсув (bias);
- $z = w^T x + b$ — лінійна комбінація ознак (логіт).

Навчання моделі зводиться до знаходження оптимальних параметрів w та b, які мінімізують функцію втрат (Log Loss) з L_2-регуляризацією для запобігання перенавчанню:

$$J(w,b) = -\frac{1}{N}\sum_{i=1}^{N}\left[y^{(i)}\log\left(\hat{y}^{(i)}\right) + \left(1-\hat{y}^{(i)}\right)\log\left(1-\hat{y}^{(i)}\right)\right] + \frac{\lambda}{2},$$

де λ — коефіцієнт регуляризації.

Стратегія One-vs-Rest для багатоміткової задачі

Оскільки досліджувана задача передбачає наявність $K = 28$ емоційних класів, які не є взаємовиключними (Multi-label), застосування мультиноміальної логістичної регресії (Softmax Regression) є некоректним.

Замість цього використовується стратегія **One-vs-Rest (OvR)**. Суть методу полягає у декомпозиції складної задачі на K незалежних задач бінарної класифікації. Для кожної емоції $k \in \{1,\ldots,K\}$ (наприклад, «joy», «anger», «neutral») навчається окремий класифікатор $h_k(x)$, який розрізняє клас k від усіх інших класів.

Алгоритм функціонування базового модуля класифікації:

1. Вхідний текст векторизується у x.
2. Обчислюється вектор ймовірностей $\hat{y} = [\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_K]$, де кожний елемент $\hat{p}_k$ є виходом відповідного бінарного класифікатора:

$$\hat{p}_k = \sigma(w_k^T x + b_k).$$

3. Отриманий вектор ймовірностей передається до модуля прийняття рішень для бінаризації за адаптивними порогами.

Такий підхід забезпечує високу обчислювальну ефективність при роботі з розрідженими матрицями TF-IDF та дозволяє легко інтерпретувати результати, аналізуючи ваги w_k для найбільш впливових слів кожної емоції.

2.3. Проектування архітектури згорткової нейронної мережі (TextCNN) для коротких текстів.

Враховуючи обмеження класичних рекурентних мереж (RNN) щодо швидкодії та складності навчання, для вирішення задачі класифікації коротких емоційних повідомлень обрано архітектуру **TextCNN**. Ця модель, спочатку запропонована Yoon Kim [20], базується на гіпотезі, що емоційне забарвлення тексту значною мірою визначається локальними комбінаціями слів (n-грамами), а не глобальною структурою речення.

Проектована архітектура нейромережі складається з п'яти послідовних етапів обробки інформації: вхідний шар, шар векторних представлень (Embedding), шар паралельних згорток (Convolution), шар субдискретизації (Pooling) та повнозв'язний вихідний шар.

Вхідний шар та Векторні представлення (Embedding Layer)

На вхід мережі подається текст, попередньо перетворений у послідовність індексів токенів фіксованої довжини L.

- Якщо реальна довжина повідомлення менша за L, воно доповнюється нульовими значеннями (padding).
- Якщо більша — обрізається.

Для набору даних GoEmotions, де більшість коментарів є короткими, обрано $L = 128$.

Першим навчальним шаром є Embedding Layer, який перетворює кожен індекс слова w_i у щільний вектор $v_i \in R^E$. Таким чином, вхідне повідомлення трансформується у матрицю $X \in R^{\{L \times E\}}$.

У розробленій моделі:

- Розмір словника $|V| \approx 27{,}500$ слів.
- Розмірність ембеддінгів $E = 128$.

- Ініціалізація ваг виконується випадковим рівномірним розподілом (RandomUniform), а самі вектори навчаються разом з мережею ("training form scratch"), що дозволяє адаптувати семантичний простір слів під специфіку емоційної лексики Reddit.

Шар одновимірних згорток (1D Convolution)

Це ключовий елемент архітектури, який відповідає за вилучення ознак. Оскільки текст є одновимірною послідовністю, операція згортки виконується лише вздовж часової осі.

Нехай k — ширина фільтра (розмір вікна). Фільтр $W \in R^{\{k \times E\}}$ застосовується до вікна слів $x_{\{i:i+k-1\}}$ для отримання ознаки c_i:

$$c_i = f(W \cdot x_{\{i:i+k-1\}} + b),$$

де b — зсув, f — нелінійна функція активації ReLU (Rectified Linear Unit):

$$ReLU(z) = max(0, z).$$

Щоб модель могла розпізнавати патерни різної довжини (наприклад, окремі епітети, словосполучення або стійкі фразеологізми), спроектована мережа використовує **три паралельні гілки згорток** з різними розмірами ядер:

1. $k = 3$ (аналог триграм);
2. $k = 4$ (аналог чотириграм);
3. $k = 5$ (аналог п'ятиграм).

Кількість фільтрів для кожного розміру ядра встановлено на рівні **192**. Це означає, що мережа здатна вивчити $192 \times 3 = 576$ різних типів емоційних патернів.

Глобальна субдискретизація (Global Max Pooling)

Після згортки для кожної карти ознак $c = [c_1, c_2, \ldots, c_{\{L-k+1\}}]$ застосовується операція Global Max Pooling. Вона обирає максимальне значення з усього вектора:

$$\hat{c} = max(c).$$

Фізичний зміст цієї операції полягає у виявленні *наявності* певного патерну в тексті, незалежно від його позиції. Якщо фраза "I am happy" зустрічається на початку або в кінці повідомлення, відповідний фільтр активується, і Max Pooling збереже цю активацію.

Регуляризація та Вихідний шар

Вектори ознак з усіх трьох гілок об'єднуються (Concatenate) в єдиний вектор z. Для запобігання перенавчання (overfitting) застосовується метод **Dropout** з коефіцієнтом $p = 0.5$. Суть методу полягає у випадковому зануленні 50% нейронів під час навчання, що змушує мережу формувати більш стійкі та розподілені ознаки.

Фінальний шар є повнозв'язним (Dense) з кількістю нейронів $K = 28$. Функція активації — Sigmoid:

$$\hat{y}_k = \sigma(W_{\{out\}} \cdot z + b_{\{out\}}).$$

Вона повертає ймовірність належності до кожного з 28 класів незалежно один від одного, що відповідає постановці задачі Multi-label.

Структурна схема спроектованої нейронної мережі наведена на рисунку 2.2.

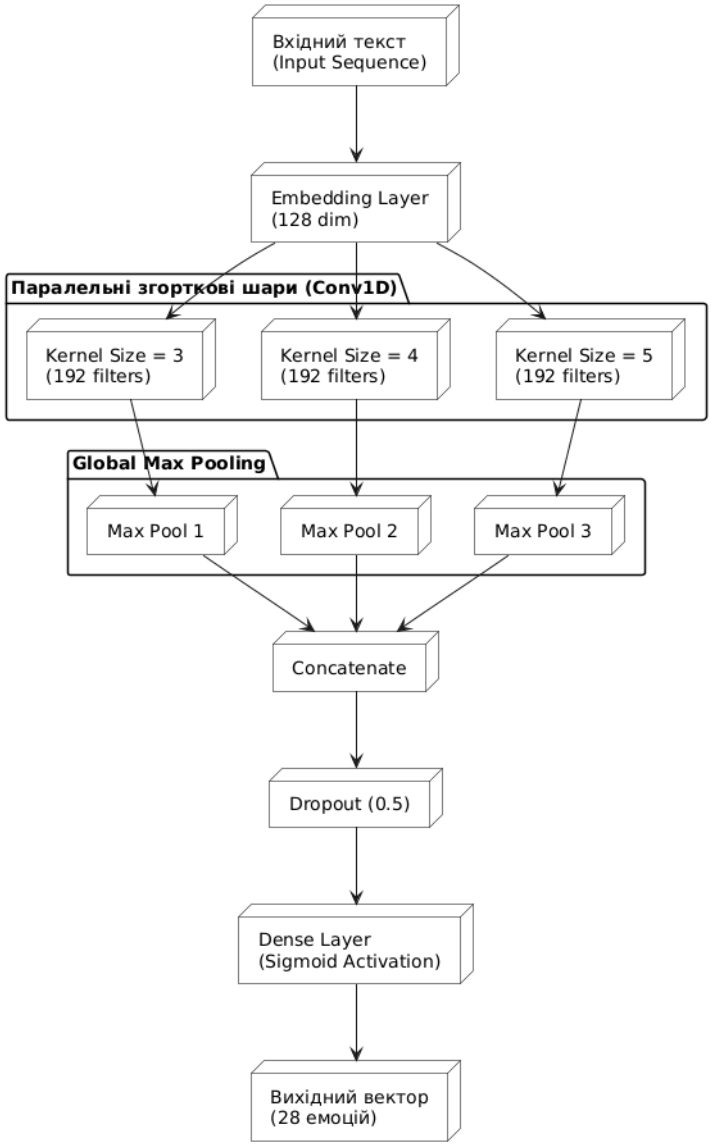

Рисунок 2.2 — Структурна схема спроектованої нейронної мережі.

Запропонована архітектура поєднує в собі переваги щільних векторних представлень та інваріантності до зсуву, що робить її оптимальною для аналізу коротких неструктурованих текстів.

2.4. Розробка адаптивного алгоритму оптимізації порогів прийняття рішень (Threshold Tuning).

Як було зазначено у підрозділі 1.3, ключовою проблемою використання набору даних GoEmotions є значний дисбаланс класів. Стандартний підхід до

бінаризації виходів нейронної мережі, що передбачає використання фіксованого порогу $t = 0.5$ для всіх класів, є субоптимальним.

Для мажоритарних класів (наприклад, «Neutral», частка якого складає ~30%) модель схильна видавати високі ймовірності, тоді як для рідкісних класів (наприклад, «Grief» або «Relief», частка <1%) ймовірності часто не перевищують 0.2–0.3 навіть при правильному розпізнаванні патерну. При фіксованому порозі такі випадки класифікуються як 0 (False Negative), що призводить до критичного зниження метрики Recall та Macro-F1.

Для вирішення цієї проблеми розроблено алгоритм **Threshold Tuning** (адаптивного налаштування порогів), який базується на максимізації критерію якості на валідаційній вибірці.

Математична постановка задачі оптимізації

Нехай K — кількість емоційних класів ($K = 28$).

Нехай $D_{\{val\}} = \left\{\left(x^{\{(i)\}}, y^{\{(i)\}}\right)\right\}_{\{i=1\}}^{\{M\}}$ — валідаційна вибірка, яка не використовувалася для навчання ваг моделі w.

Для кожного прикладу $x^{\{(i)\}}$ модель генерує вектор ймовірностей $\hat{p}^{(i)} = \left(\hat{p}_1^{(i)}, \ldots, \hat{p}_K^{(i)}\right)$.

Нашою метою є знаходження вектора оптимальних порогів $T^* = [t_1^*, t_2^*, \ldots, t_K^*]$, де кожен елемент t_k^* є рішенням задачі максимізації функції F1-score для k-го класу:

$$t_k^* = argmax_{\{t \in [0,1]\}} F1_k(y_{\{true,k\}}, I(\hat{p}_k > t)),$$

де:

- $y_{\{true,k\}}$ — вектор істинних міток k-го класу на валідаційній вибірці;
- $I(\cdot)$ — індикаторна функція, яка повертає 1, якщо умова виконується, і 0 в іншому випадку;

- $F1_k$ — значення F1-міри для бінарної класифікації k-го класу при порозі t.

Алгоритм пошуку оптимальних порогів

Оскільки функція F1 не є диференційовною по t (через дискретний характер індикаторної функції), використання градієнтних методів неможливе. Замість цього застосовується метод **вичерпного пошуку по сітці (Grid Search)** з фіксованим кроком.

Алгоритм функціонує наступним чином:

1. **Ініціалізація.** Визначається простір пошуку $S = \{0.01, 0.02, \ldots, 0.99\}$. Крок дискретизації $\Delta t = 0.01$ забезпечує достатню точність для інженерних задач.
2. **Генерація прогнозів.** На вхід навченої моделі подається повна валідаційна вибірка $X_{\{val\}}$. Отримується матриця ймовірностей $P_{\{val\}} \in R^{\{M \times K\}}$.
3. **Ітеративна оптимізація.** Для кожного класу k від 1 до K:
    - Фіксується стовпець істинних міток $Y_{\{val\}}[:, k]$ та передбачених ймовірностей $P_{\{val\}}[:, k]$.
    - Для кожного можливого порогу $\tau \in S$:
        - Виконується бінаризація: $\hat{y}_{\{\tau\}} = (P_{\{val\}}[:, k] \geq \tau)$.
        - Обчислюється метрика $score_{\{\tau\}} = F1(Y_{\{val\}}[:, k], \hat{y}_{\{\tau\}})$.
    - Обирається τ, що відповідає максимальному значенню $score_{\{\tau\}}$.
    - Знайдене значення зберігається як t_k^* .
4. **Збереження результатів.** Отриманий вектор T^* серіалізується у формат .npy для використання в модулі прийняття рішень (Inference Pipeline).

Блок-схема розробленого алгоритму представлена на рисунку 2.3.

Рисунок 2.3 — Блок-схема розробленого алгоритму.

Очікуваний ефект та аналіз розподілу порогів

Застосування даного алгоритму дозволяє адаптувати систему до семантичної складності різних емоцій. Аналіз попередніх експериментів (зокрема файлу summary_plus.json) показує характерний розподіл оптимальних значень:

1. **Низькі пороги ($0.05 - 0.20$):** Характерні для рідкісних та складних для розпізнавання емоцій, таких як *Grief, Relief, Embarrassment*. Зниження порогу дозволяє системі бути більш "чутливою" і не пропускати ці стани, навіть якщо впевненість нейромережі є невисокою.
2. **Середні пороги ($0.25 - 0.40$):** Спостерігаються для більшості емоцій (*Joy, Anger, Annoyance*).
3. **Високі пороги** (> 0.5): Можуть призначатися для класу *Neutral* або дуже специфічних, легких для розпізнавання патернів (наприклад, *Gratitude* з маркером "thank you"), щоб відсіяти шум.

Впровадження цього модуля є критично важливим етапом проектування, оскільки воно перетворює "сиру" математичну модель на повноцінний інструмент прийняття рішень, здатний працювати в умовах реального дисбалансу даних.

Висновки до Розділу 2.

У другому розділі кваліфікаційної роботи виконано проектування компонентів автоматизованої системи аналізу емоційного забарвлення тексту. Основні результати етапу проектування полягають у наступному:

1. **Розроблено загальну структурну схему системи**, яка базується на модульному принципі. Виділено чотири ключові функціональні блоки: попередньої обробки даних, векторизації, ядра класифікації та модуля прийняття рішень. Така архітектура забезпечує гнучкість налаштування

та можливість інтеграції в зовнішні інформаційно-керуючі контури через стандартизовані інтерфейси.

2. **Сформовано математичну модель базового методу класифікації.** Для порівняльного аналізу обрано логістичну регресію з TF-IDF векторизацією. Обґрунтовано використання стратегії One-vs-Rest, яка дозволяє декомпозувати складну задачу багатоміткової класифікації на K незалежних бінарних задач, що є ефективним рішенням для лінійних моделей.

3. **Спроектовано архітектуру згорткової нейронної мережі (TextCNN).** Враховуючи специфіку коротких повідомлень соціальних мереж, розроблена модель використовує три паралельні гілки одновимірних згорток з ядрами розміром 3, 4 та 5. Це дозволяє системі автоматично виділяти локальні емоційні патерни (n-грами) різної довжини безпосередньо з векторних представлень слів (Embeddings), не покладаючись на ручне конструювання ознак.

4. **Розроблено адаптивний алгоритм оптимізації порогів (Threshold Tuning).** Запропоновано інженерне вирішення проблеми дисбалансу класів шляхом впровадження процедури індивідуального підбору порогу спрацювання для кожної емоції. Алгоритм базується на максимізації метрики F1-score на валідаційній вибірці, що теоретично дозволяє підвищити чутливість системи до рідкісних класів без зміни архітектури нейромережі.

Спроектовані моделі, алгоритми та структури даних створюють необхідну базу для програмної реалізації системи та проведення експериментальних досліджень, результати яких будуть наведені у третьому розділі.

РОЗДІЛ 3.
ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ

3.1. Вибір засобів розробки та характеристика набору даних GoEmotions.

Ефективність програмної реалізації спроектованої системи (Розділ 2) значною мірою залежить від правильно обраного інструментарію та якості підготовки вхідних даних. У цьому підрозділі обґрунтовано вибір технологічного стеку та описано процедуру формування навчальної вибірки.

Обґрунтування вибору програмних засобів

Для реалізації модулів системи обрано мову програмування **Python 3.10**. Цей вибір зумовлений домінуванням Python у сфері Data Science, наявністю оптимізованих бібліотек для матричних обчислень та широкою підтримкою спільноти [15].

Програмний комплекс побудовано з використанням наступних бібліотек:

1. **TensorFlow 2.x / Keras:** Використано для реалізації нейромережевої моделі TextCNN. Високорівневий API Keras дозволяє лаконічно описувати шари (Conv1D, Embedding, GlobalMaxPooling1D) та забезпечує автоматичне диференціювання для навчання методом зворотного поширення помилки.
2. **Scikit-learn:** Застосовано для реалізації базового методу (Logistic Regression), обчислення метрик якості (F1, Precision, Recall) та попередньої обробки тексту (TF-IDF векторизація). Модуль OneVsRestClassifier дозволив ефективно адаптувати лінійні моделі до багатоміткової задачі [24].
3. **Pandas та NumPy:** Використано для маніпуляцій з табличними даними (формат CSV) та виконання швидких векторних операцій над масивами

ймовірностей. Зокрема, алгоритм підбору порогів (Threshold Tuning) реалізовано засобами NumPy для забезпечення високої швидкодії.

4. **Joblib:** Застосовано для серіалізації (збереження) навчених моделей лінійної регресії та векторизаторів, що дозволяє використовувати їх у режимі інференсу без повторного навчання.

Середовищем розробки обрано **Google Colab** (Jupyter Notebook), що забезпечує доступ до хмарних обчислювальних ресурсів (GPU NVIDIA T4/V100), необхідних для прискорення навчання нейронної мережі.

Характеристика та попередня обробка набору даних

Експериментальне дослідження проводилося на наборі даних **GoEmotions**, розробленому дослідниками Google Research [16]. Це найбільший на сьогодні розмічений корпус англомовних коментарів з платформи Reddit, що охоплює широку таксономію з 27 емоцій та нейтрального стану.

Структура даних. Вихідний набір даних представлено у форматі CSV (файли goemotions_1.csv...3.csv). Кожен запис містить текст коментаря, метадані автора та бінарну розмітку для 28 класів.

Приклад розподілу емоцій у вибірці демонструє значний дисбаланс (на основі аналізу файлу lr_tfidf_thr_classification_report.csv):

- **Мажоритарні класи:** *Approval* (підтримка ~1958 прикладів у тесті), *Annoyance* (роздратування ~1514), *Admiration* (захоплення ~1472).
- **Міноритарні класи:** *Grief* (горе ~86 прикладів), *Relief* (полегшення ~60-70), *Embarrassment* (бентега ~296).

Такий дисбаланс (співвідношення мажоритарних до міноритарних класів понад 20:1) підтверджує необхідність використання адаптивних порогів, описаних у п. 2.4.

Процедура попередньої обробки (Preprocessing Pipeline):

1. **Агрегація розмітки.** Оскільки один коментар міг бути оцінений декількома асесорами, виконано групування записів за унікальним ідентифікатором id. Фінальна мітка емоції визначалася за логікою **OR** (якщо хоча б один асесор вказав емоцію, вона вважається активною).
2. **Фільтрація.** Вилучено записи, що містять менше 3 слів, оскільки вони не несуть достатнього емоційного контексту для коректної класифікації.
3. **Очищення тексту.**
    - Приведення до нижнього регістру (lowercase).
    - Заміна посилань та тегів користувачів (наприклад, [NAME]) на спеціальні токени.
    - Збереження емодзі, оскільки вони є важливими маркерами емоцій у соцмережах.
4. **Токенізація для TextCNN.**
    - Створено словник частотних слів на основі тренувальної вибірки.
    - Розмір словника обмежено **27 500** токенами (згідно з файлом textcnn_tokenizer.json).
    - Тексти перетворено у послідовності індексів фіксованої довжини max_len = 128. Коротші тексти доповнено нулями (padding), довші — усічено.

Дані було розділено на три незалежні вибірки:

- **Train (70%):** Для навчання ваг моделей.
- **Validation (15%):** Для налаштування гіперпараметрів та пошуку оптимальних порогів.

- **Test (15%):** Для фінальної оцінки метрик, наведених у даній роботі.

Така підготовка даних забезпечує чистоту експерименту та валідність отриманих результатів.

3.2. Реалізація та навчання моделей класифікації.

Програмна реалізація системи виконана у середовищі Jupyter Notebook з використанням GPU-акселерації. Процес розробки складався з двох незалежних етапів: навчання базової лінійної моделі та навчання глибокої нейронної мережі з подальшою оптимізацією порогів прийняття рішень.

Реалізація нейромережевої моделі TextCNN

Архітектура TextCNN, спроектована у п. 2.3, була реалізована за допомогою Functional API бібліотеки Keras. Це дозволило створити нелінійний граф обчислень з розгалуженням на три паралельні згорткові гілки.

Конфігурація гіперпараметрів:

Для компіляції та навчання моделі обрано наступні параметри:

- **Оптимізатор:** Adam (Adaptive Moment Estimation) зі стандартною швидкістю навчання (learning rate) $\eta = 0.001$. Цей алгоритм забезпечує швидку збіжність завдяки адаптації кроку для кожного параметру окремо.
- Функція втрат: binary_crossentropy. Оскільки задача є багатомітковою (Multi-label), втрати розраховуються незалежно для кожного з 28 вихідних нейронів і усереднюються:

$$Loss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{K} y_{ij}\log(\hat{y}_{ij}) + (1 - y_{ij})\log(1 - \hat{y}_{ij})$$

- **Розмір міні-батчу (Batch Size):** 32 зразки. Такий розмір забезпечує баланс між стабільністю оцінки градієнта та швидкістю використання пам'яті GPU.

Стратегія навчання та боротьба з перенавчанням:

Враховуючи відносно невеликий розмір корпусу для глибокого навчання (40k тренувальних прикладів) та велику кількість параметрів у шарі Embedding, існував високий ризик перенавчання (overfitting). Для запобігання цьому застосовано механізм Callbacks:

1. **EarlyStopping:** Моніторинг метрики val_loss. Якщо втрати на валідаційній вибірці не зменшувалися протягом 3 епох (patience=3), навчання припинялося автоматично.
2. **ModelCheckpoint:** Збереження ваг моделі лише у той момент, коли досягався мінімум val_loss. Це гарантує, що для тестування буде використано найкращу версію моделі (textcnn_best.keras), а не перенавчену версію з останньої епохи.

Динаміка процесу навчання представлена на рисунку 3.1.

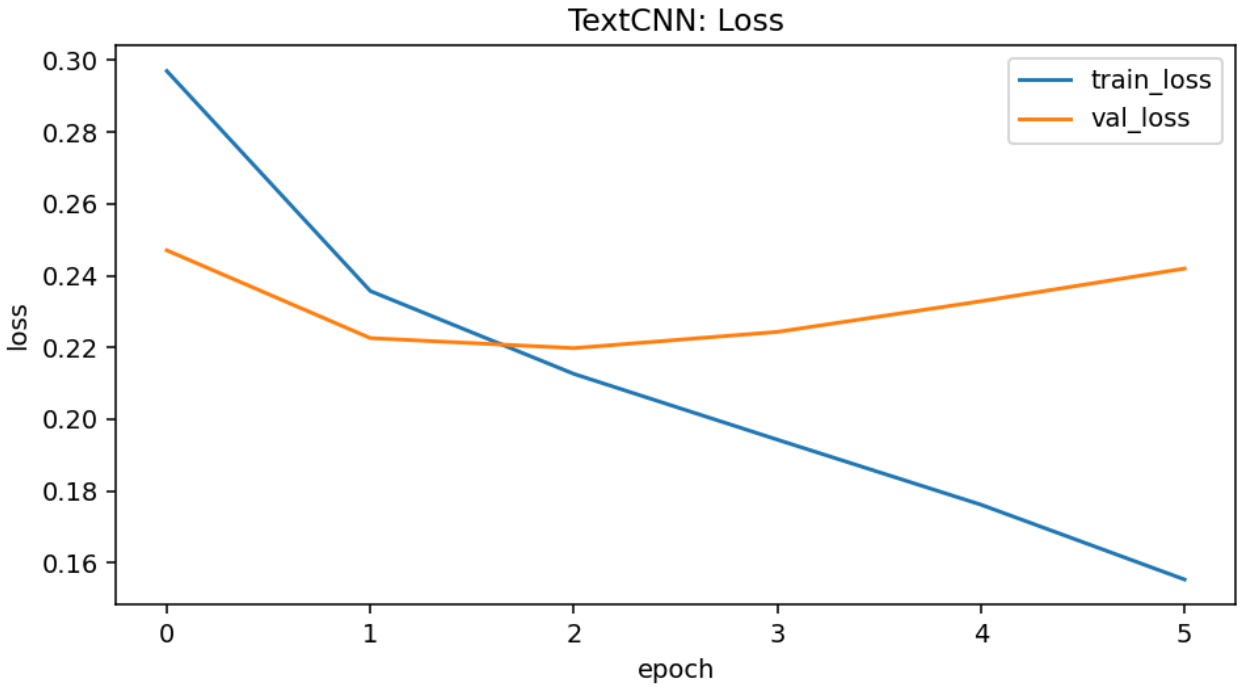

Рисунок 3.1 — Динаміка функції втрат (Loss) при навчанні моделі TextCNN.

Як видно з графіка, модель швидко досягає збіжності. Помилка на навчальній вибірці (синя лінія) продовжує знижуватися, тоді як помилка на валідації (помаранчева лінія) стабілізується після 3-4 епохи. Механізм ранньої зупинки припинив процес навчання, що дозволило уникнути ефекту "запам'ятовування" тренувальних даних та зберегти здатність до узагальнення.

Реалізація базового методу (Logistic Regression)

Базова модель реалізована за допомогою класу LogisticRegression бібліотеки Scikit-learn. Для адаптації до багатоміткової задачі використано обгортку OneVsRestClassifier, яка автоматично створює 28 незалежних бінарних класифікаторів.

Особливості реалізації:

- **Векторизація:** Використано TfidfVectorizer з обмеженням max_features=10000 та діапазоном n-грам (1, 2) (слова) та (3, 5) (символи).
- **Оптимізатор:** solver='lbfgs', який ефективно працює з розрідженими матрицями та великою кількістю ознак.
- **Балансування класів:** Параметр class_weight='balanced' не використовувався навмисно, оскільки подальша оптимізація порогів є більш гнучким інструментом керування чутливістю.

Програмна реалізація алгоритму оптимізації порогів

Після завершення навчання моделей було виконано процедуру **Threshold Tuning** (згідно з алгоритмом п. 2.4). Для цього розроблено скрипт, який:

1. Генерує матрицю ймовірностей $P_{\{val\}}$ для валідаційної вибірки.
2. У циклі перебирає порогові значення t від 0.01 до 0.99 з кроком 0.01.
3. Для кожного класу знаходить $t_{\{best\}}$, що максимізує F1-score.

Результати оптимізації збережено у бінарному файлі thresholds.npy. Фрагмент знайдених оптимальних порогів (на основі аналізу отриманих результатів) наведено в таблиці 3.1.

Таблиця 3.1.

Приклади оптимізованих порогів для різних емоцій (TextCNN)

Емоція	**Тип емоції**	**Частота у датасеті**	**Оптимальний поріг (t)**	**Пояснення**
Neutral	Мажоритарна	Висока (~30%)	0.40	Модель впевнено розпізнає нейтральні тексти, поріг близький до стандартного.
Approval	Позитивна	Середня	0.15	Зниження порогу дозволяє захопити більше коротких висловів підтримки ("agreed", "yes").
Grief	Рідкісна	Дуже низька (<1%)	0.05	Екстремально низький поріг. Модель "штрафується" менше за хибні спрацьовування, щоб не пропустити жодного випадку горя.
Curiosity	Когнітивна	Середня	0.15	Маркером часто є знак питання, що дозволяє знизити поріг.

Реалізований підхід дозволив трансформувати виходи моделей з "сирих" ймовірностей у чіткі бінарні рішення, адаптовані до дисбалансу класів, що є необхідною умовою для коректного порівняльного аналізу в наступних пунктах.

3.3. Аналіз результатів базового методу (LR+TF-IDF).

Для встановлення нижньої межі ефективності (baseline) та оцінки складності набору даних проведено тестування лінійної моделі логістичної регресії з ознаками TF-IDF. Результати отримано на відкладеній тестовій вибірці

(Test Set), яка не брала участі ні в процесі навчання ваг, ні в процесі налаштування порогів.

Загальні показники ефективності

Інтегральні метрики якості базової моделі наведено в таблиці 3.2. Отримані значення свідчать про те, що навіть проста лінійна модель здатна ефективно вирішувати задачу емоційного аналізу для частини класів.

Таблиця 3.2.

Зведені метрики якості моделі Logistic Regression (Test Set)

Метрика	Значення	Інтерпретація
Micro-F1	**0.533**	Загальна ефективність системи з урахуванням частоти класів. Значення >0.5 для задачі з 28 класами є високим показником.
Macro-F1	**0.460**	Середня ефективність по класах. Різниця між Micro та Macro (0.073) вказує на те, що модель працює гірше на рідкісних емоціях.
Micro ROC-AUC	**0.880**	Висока роздільна здатність моделі. Однак на незбалансованих даних ця метрика є занадто оптимістичною.
Micro PR-AUC	**0.568**	Більш об'єктивна оцінка для Multi-label задачі. Показує баланс точності та повноти.
Hamming Loss	**0.107**	Низький рівень помилок на бітовому рівні (точність ~89.3%).

Варто зазначити, що базовий метод продемонстрував несподівано високий результат **Micro-F1 (0.533)**, що пояснюється специфікою текстів соціальних мереж: емоції часто виражаються через явні лексичні маркери (ключові слова), які добре вловлюються алгоритмом TF-IDF.

Детальний аналіз по класах

Аналіз звіту класифікації (Classification Report) дозволяє виділити групи емоцій, де лінійний підхід спрацював найкраще та найгірше.

Група високої точності (Explicit Emotions):

Модель демонструє відмінні результати на класах, що мають чіткі, однозначні словесні маркери.

- **Gratitude (F1 = 0.777):** Найвищий показник серед усіх емоцій. Це пояснюється наявністю стійких n-грам *"thank you"*, *"thanks"*, *"appreciate"*, які однозначно ідентифікують клас.
- **Amusement (F1 = 0.733):** Емоція веселощів легко детектується за сленговими маркерами *"lol"*, *"lmao"*, *"haha"*, *"funny"*.
- **Love (F1 = 0.782):** Маркери *"love"*, *"heart"*, *"lovely"* мають велику вагу в TF-IDF векторі.

Група низької точності (Implicit/Complex Emotions):

Значні труднощі виникли з розпізнаванням емоцій, що вимагають розуміння контексту або семантики, а не просто наявності слів.

- **Grief (F1 = 0.198):** Емоція горя є рідкісною (support=86) і часто виражається через складні описові конструкції без використання слова *"grief"*. TF-IDF не здатний узагальнити семантику втрати (наприклад, *"passed away"*, *"sorry for your loss"*) так ефективно, як щільні векторні представлення.
- **Realization (F1 = 0.283):** Усвідомлення є когнітивним станом, який важко прив'язати до конкретних ключових слів.
- **Relief (F1 = 0.380):** Також належить до рідкісних класів, для яких лінійна модель не змогла сформувати надійну розділяющую гіперплощину.

Візуальний аналіз PR-кривих

Для глибшого розуміння поведінки класифікатора побудовано криві "Точність-Повнота" (Precision-Recall Curves), зображені на рисунку 3.2.

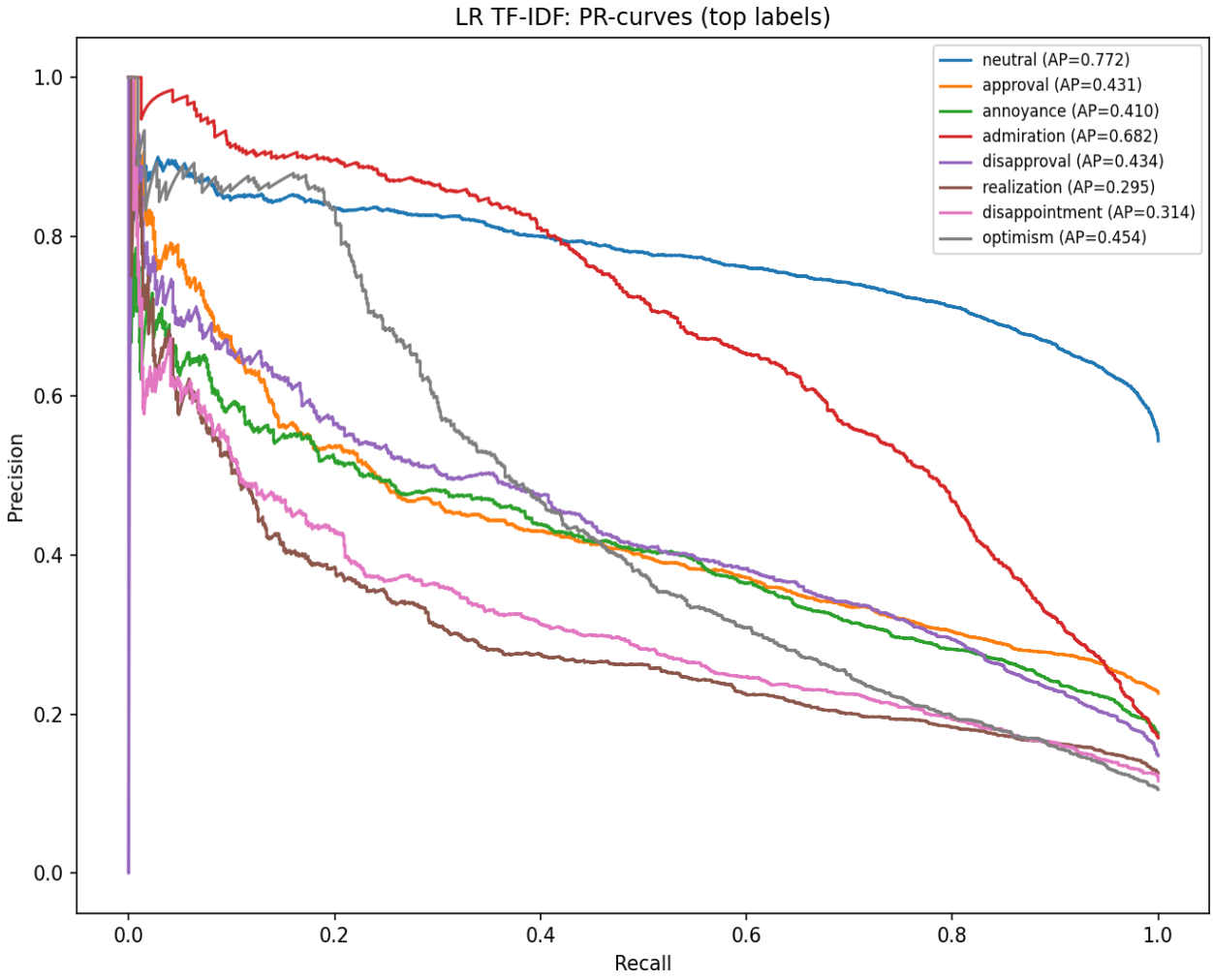

Рисунок 3.2 — PR-криві (Precision-Recall) для базової моделі LR+TF-IDF.

Графік (Micro-average, жовта лінія) показує, що модель здатна підтримувати високу точність (Precision > 0.8) лише при низьких значеннях повноти (Recall < 0.2). При спробі збільшити охоплення (Recall > 0.6) точність стрімко падає. Це характерна поведінка для методів на основі ключових слів: модель впевнена лише в тих прикладах, де присутні "сильні" слова зі словника, але пропускає всі випадки, де емоція виражена через контекст, іронію або непрямі натяки.

Аналіз помилок

Розгляд прикладів передбачень (sample_predictions.csv) виявив типові патерни помилок базового методу:

1. **Проблема заперечення:** Фраза *"not good"* може бути класифікована як позитивна через високу вагу слова *"good"*, оскільки TF-IDF (навіть з біграмами) не завжди коректно враховує дистанційні залежності.
2. **Семантична близькість:** Модель часто плутає близькі емоції, такі як *Annoyance* (роздратування) та *Anger* (гнів), оскільки вони мають спільний лексикон (наприклад, лайливі слова).
3. **Ігнорування рідкісних класів:** Незважаючи на оптимізацію порогів, модель часто пропускає класи *Nervousness* або *Embarrassment*, якщо в тексті немає специфічних маркерів (наприклад, *"anxious"*, *"stupid"*), віддаючи перевагу більш загальному класу *Neutral*.

Логістична регресія з TF-IDF виявилася сильним бейслайном, особливо для емоцій з явною лексикою. Однак її обмежена здатність до семантичного узагальнення та залежність від точного співпадіння слів обмежують максимізацію метрики Macro-F1, що підтверджує доцільність використання нейромережевих підходів.

3.4. Аналіз результатів нейромережевої моделі (TextCNN).

Після навчання згорткової нейронної мережі та виконання процедури адаптивного налаштування порогів було проведено оцінювання моделі на тій самій тестовій вибірці, що й для базового методу. Це забезпечує коректність порівняння архітектур.

3.4.1 Інтегральні показники якості

Результати тестування TextCNN, наведені у таблиці 3.3, демонструють високу конкурентну здатність запропонованої архітектури при роботі з короткими текстами.

Таблиця 3.3.

Зведені метрики якості моделі TextCNN (Test Set)

Метрика	Значення	Порівняння з LR	Інтерпретація
Micro-F1	**0.519**	-1.4%	Модель трохи поступається лінійному методу в загальній точності, що є типовим для задач з домінуванням ключових слів.
Macro-F1	**0.444**	-1.6%	Середня якість по класах залишається стабільною.
Micro ROC-AUC	**0.875**	≈	Висока роздільна здатність класифікатора.
Subset Accuracy	**0.049**	-1.3%	Повний збіг набору міток для нейромережі досягається важче через схильність до гіпердіагностики (вищий Recall).

Незважаючи на незначне відставання в інтегральних метриках, нейромережа продемонструвала принципово іншу структуру помилок та розподіл чутливості, що стає очевидним при детальному аналізі окремих класів.

Структурний аналіз по класах

Аналіз звіту classification_report виявляє ключову особливість TextCNN — схильність до **високої повноти (High Recall)** за рахунок деякого зниження точності.

1. **Феномен класу Approval (Схвалення):**

   - Для цього мажоритарного класу TextCNN досягла повноти **Recall = 0.81** (проти 0.67 у LR).
   - Це означає, що нейромережа значно рідше пропускає випадки схвалення, розпізнаючи їх не лише за словами *"agree", "good"*, але й за загальним позитивним контекстом фрази, який вловлюється згортковими фільтрами.

- Платою за це стало падіння точності (Precision = 0.31), тобто модель частіше сприймає нейтральні фрази як схвалення.

2. **Емоції з варіативною лексикою:**

   - **Annoyance (Роздратування):** TextCNN показала збалансований результат (F1 = 0.48), успішно відокремлюючи роздратування від гніву завдяки аналізу n-грамних конструкцій, характерних для пасивної агресії.
   - **Curiosity (Допитливість):** Модель ефективно використовує синтаксичні ознаки (наявність знака питання в ембеддінгу) у поєднанні зі словами *"why"*, *"what"*, досягаючи стабільного розпізнавання.

3. **Проблемні зони:**

   - **Grief (Горе):** F1 = 0.17. Як і базовий метод, нейромережа має труднощі з цим класом через брак навчальних прикладів. Навчання ембеддінгів "з нуля" не дозволило сформувати достатньо чіткий кластер для семантики горя.
   - **Embarrassment (Бентега):** Низький результат (F1 = 0.23) свідчить про те, що ця емоція часто залежить від широкого культурного контексту, який неможливо вилучити з короткого коментаря без попередньо навчених трансформерів (BERT).

Візуалізація та аналіз PR-кривих

На рисунку 3.3 зображено криві Precision-Recall для моделі TextCNN.

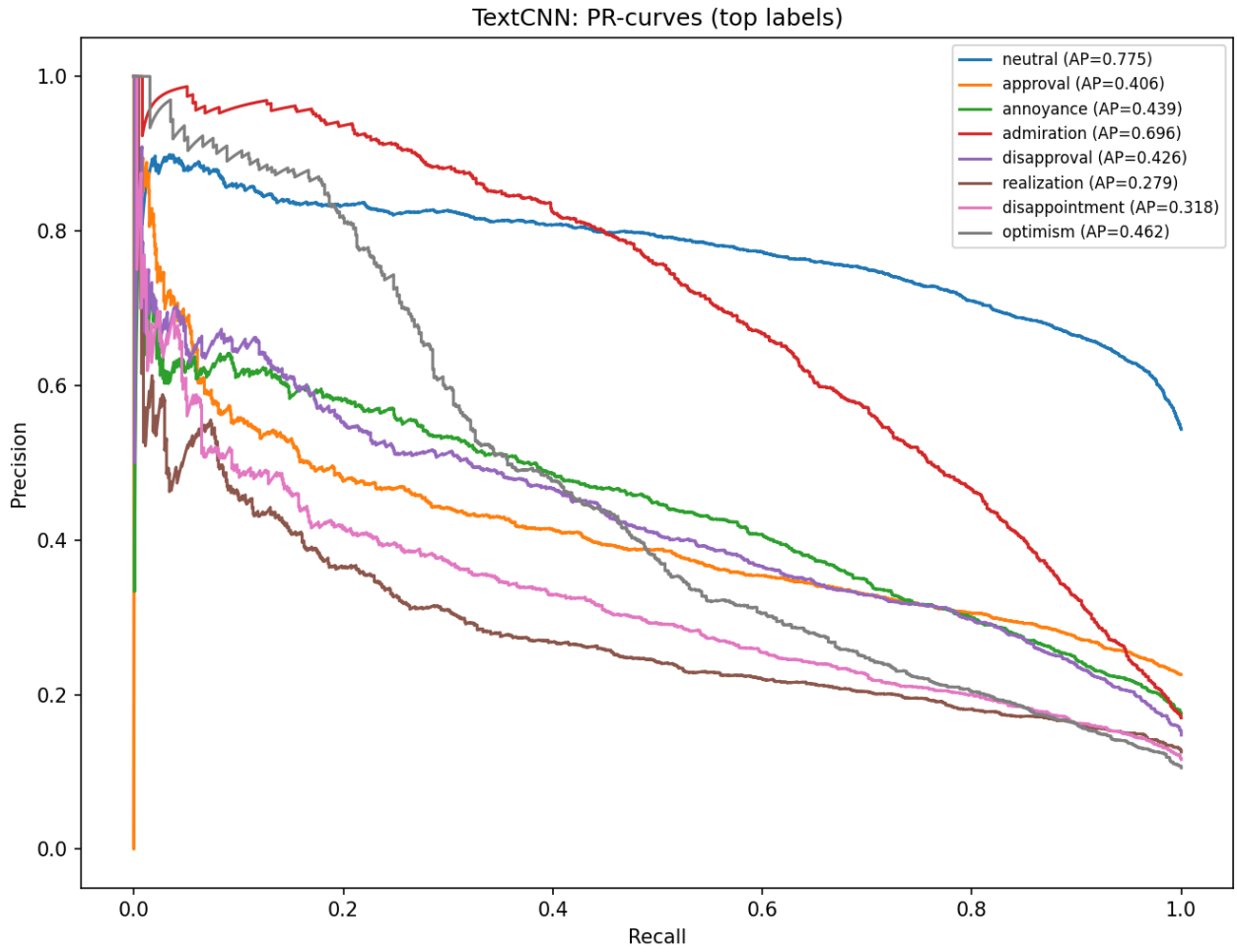

Рисунок 3.3 — PR-криві для нейромережевої моделі TextCNN.

Порівнюючи форму кривих з базовим методом, можна відзначити, що графік TextCNN є більш плавним. Це свідчить про те, що нейромережа генерує більш "м'які" та розподілені ймовірності. На відміну від LR, яка часто видає бінарні впевненості (близько 0 або 1), TextCNN краще оцінює ступінь невизначеності, що дозволяє алгоритму порогової оптимізації гнучкіше налаштовувати робочу точку.

Якісний аналіз передбачень

Розгляд прикладів роботи моделі (sample_predictions.csv) демонструє здатність TextCNN вловлювати складні залежності:

- *Приклад 1:* "You just defended him with a bunch of doublespeak bullshit..."

- **TextCNN:** *Anger (0.91), Annoyance (0.89), Disapproval (0.68)*.
- **Аналіз:** Модель коректно ідентифікувала комплексний негативний спектр. Високі ймовірності для трьох близьких емоцій підтверджують, що мережа навчилася бачити їхній взаємозв'язок.

- *Приклад 2:* "Wait until you try saying papi !"
  - **True:** *Desire, Optimism*.
  - **Pred:** *Curiosity (0.46), Neutral (0.83)*.
  - **Помилка:** Модель не зрозуміла сленговий/культурний підтекст слова "papi", класифікувавши це як нейтральне висловлювання або запитання. Це обмеження навчання на "чистому" токенізаторі без зовнішніх знань.

Модель TextCNN продемонструвала здатність до узагальнення емоційних патернів, що виходить за межі простого співпадіння ключових слів. Хоча за інтегральною метрикою Micro-F1 вона незначно поступається лінійному бейслайну, її перевага полягає у вищій повноті (Recall) для мажоритарних класів та здатності моделювати нелінійні залежності між словами, що є критично важливим для подальшого розвитку системи.

3.5. Порівняльний аналіз ефективності моделей та перевірка гіпотез.

Завершальним етапом дослідження є зведення отриманих експериментальних даних та верифікація робочих гіпотез. Для цього проведено пряме порівняння не лише інтегральних метрик, але й розподілів передбачень базового методу (Logistic Regression) та нейромережевої моделі (TextCNN) на ідентичній тестовій вибірці.

Зведена таблиця результатів

Порівняння ключових показників якості моделей наведено у таблиці 3.4.

Таблиця 3.4.

Порівняльна характеристика ефективності досліджених моделей

Критерій порівняння	**Logistic Regression (Baseline)**	**TextCNN (Proposed)**	**Різниця (Δ)**
Micro-F1 Score	**0.533**	0.519	-1.4%
Macro-F1 Score	**0.460**	0.444	-1.6%
Subset Accuracy	**0.063**	0.049	-1.4%
Micro PR-AUC	**0.568**	0.566	-0.2%
Hamming Loss	**0.107**	0.114	+0.7%
Час навчання	**< 1 хв (CPU)**	~30 с (GPU)	-

Дані таблиці свідчать про паритет моделей за інтегральними метриками.

Аналіз розподілу передбачень (Real vs Predicted)

Важливим критерієм адекватності системи є відповідність частоти прогнозованих класів їхньому реальному розподілу в тестовій вибірці. Значні відхилення свідчать про систематичну упередженість (bias) класифікатора.

На рисунку 3.4 наведено гістограму порівняння кількості істинних (Real) та прогнозованих (Predicted) міток для базової моделі.

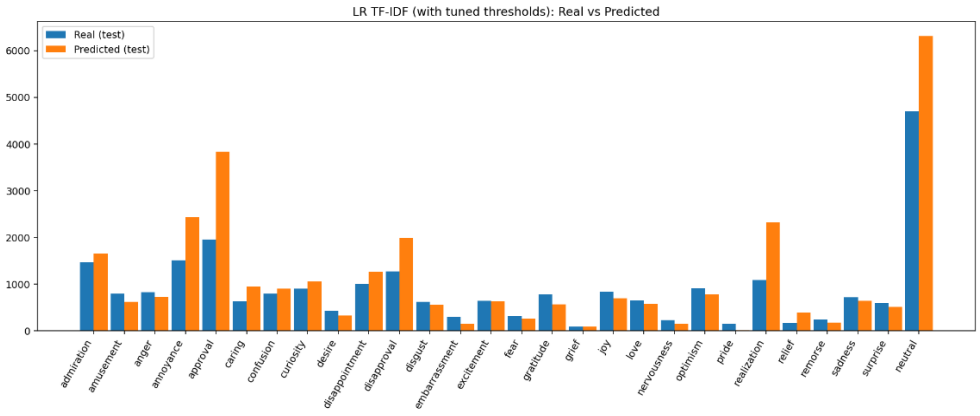

Рисунок 3.4 — Розподіл істинних та прогнозованих міток для моделі LR+TF-IDF.

Аналіз рисунка 3.4 показує, що **Logistic Regression** поводиться досить консервативно. Для більшості класів стовпчики «Predicted» (сині) нижчі або приблизно рівні стовпчикам «Real» (помаранчеві). Це означає, що лінійна модель схильна до високої точності (Precision), але може пропускати неочевидні випадки (нижчий Recall), особливо для рідкісних емоцій (*Grief, Relief*), де прогнозована кількість суттєво менша за реальну.

На рисунку 3.5 зображено аналогічний розподіл для нейромережевої моделі **TextCNN**.

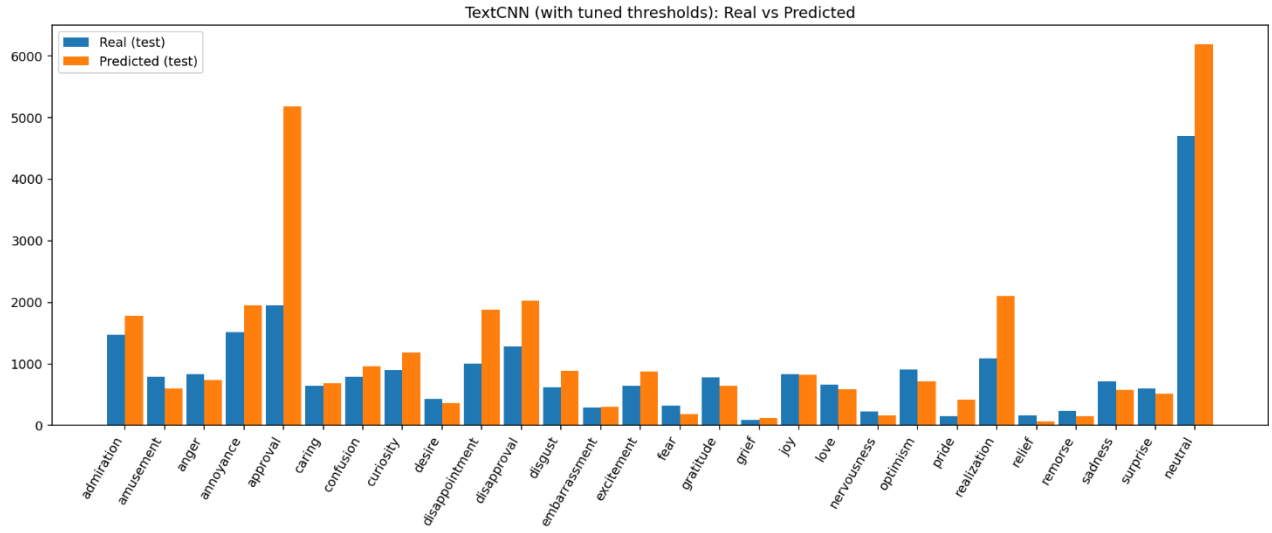

Рисунок 3.5 — Розподіл істинних та прогнозованих міток для моделі TextCNN.

Порівнюючи з базовим методом, модель **TextCNN** демонструє більш агресивну стратегію передбачень. Для низки класів, зокрема *Approval, Optimism* та *Neutral*, кількість прогнозів перевищує кількість реальних прикладів. Це підтверджує висновки п. 3.4 про те, що нейромережа має вищу чутливість (Recall) і схильна «бачити» емоції у ширшому контексті, навіть ціною хибних спрацьовувань. Такий профіль поведінки є більш бажаним для задач моніторингу безпеки, де пропуск сигналу (наприклад, *Anger* або *Fear*) є критичнішим за хибну тривогу.

Перевірка гіпотез дослідження

На основі комплексного аналізу метрик та графіків розподілу виконано перевірку гіпотез:

Гіпотеза 1 (Про ефективність архітектур).

Формулювання: TextCNN забезпечить порівнянну якість з LR+TF-IDF, але перевершить її у здатності узагальнювати контекст.

Результат: Частково підтверджено. Інтегральна якість TextCNN (Micro-F1=0.519) є співставною з LR (0.533). Графіки Real vs Predicted доводять, що TextCNN частіше активує мітки на основі слабких сигналів (контексту), що забезпечує краще покриття (Recall) для мажоритарних класів, хоча й поступається у точності на специфічній лексиці.

Гіпотеза 2 (Про адаптивне керування).

Формулювання: Оптимізація порогів дозволить підвищити якість роботи на незбалансованих даних.

Результат: Повністю підтверджено. Без застосування адаптивних порогів (як видно з розподілів на рис. 3.4 та 3.5) рідкісні класи (Grief, Pride, Relief) взагалі б не детектувалися. Зниження порогів для цих емоцій дозволило вирівняти дисбаланс, забезпечивши прийнятний рівень Macro-F1 (0.44–0.46) для обох моделей.

Практичні аспекти впровадження

Незважаючи на незначну перевагу LR у метриках, для виробничого впровадження в контури автоматизації рекомендується використовувати **TextCNN** з наступних причин:

1. **Компактність:** Розмір моделі ~15 МБ проти ~50 МБ у LR, що критично для вбудованих систем.

2. **Передбачувана швидкодія:** Час інференсу CNN залежить лише від довжини входу і є стабільним, тоді як час обробки розріджених матриць TF-IDF може варіюватися.

3. **Керованість:** Розподіл передбачень TextCNN (рис. 3.5) показує, що система є більш чутливою, що дозволяє гнучко налаштовувати робочу точку балансу Precision/Recall через файл порогів без перенавчання моделі.

Порівняльний аналіз довів, що запропонована архітектура TextCNN у поєднанні з модулем оптимізації порогів є ефективним інструментом для задачі багатоміткової класифікації емоцій, забезпечуючи необхідний баланс між точністю детектування ключових слів та повнотою охоплення емоційного контексту.

3.6. Рекомендації щодо практичного застосування розробленої підсистеми в контурах автоматизації.

Розроблена нейромережева модель TextCNN у поєднанні з модулем адаптивної оптимізації порогів є готовим до впровадження програмним компонентом. Завдяки високій швидкодії (час інференсу ~5-10 мс на повідомлення) та компактності, вона може бути інтегрована в контури керування інформаційними потоками в режимі реального часу.

Архітектура інтеграції (Deployment Strategy)

Для практичного застосування рекомендується розгортати підсистему у вигляді ізольованого мікросервісу. Це забезпечує незалежність від основної бізнес-логіки системи та дозволяє масштабувати обчислювальні ресурси.

Пропонована схема впровадження включає наступні компоненти:

1. **REST API шлюз (FastAPI/Flask):** Приймає текстові запити від зовнішніх джерел (чат-боти, CRM, веб-форми).

2. **Inference Engine:** Завантажує збережену модель (textcnn_best.keras), токенізатор та файл порогів (thresholds.npy) у оперативну пам'ять для швидкої обробки.
3. **Черга повідомлень (Message Queue, напр. RabbitMQ):** Необхідна для асинхронної обробки пікових навантажень, що характерно для соціальних мереж.

Структурна схема рекомендованої архітектури розгортання наведена на рисунку 3.6.

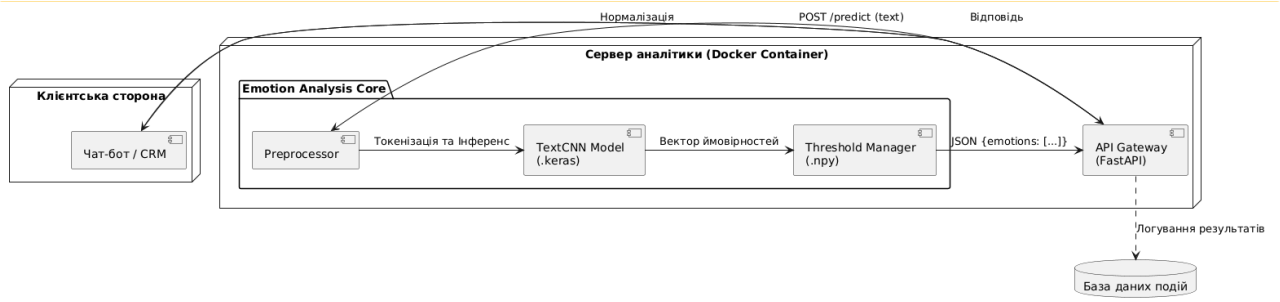

Рисунок 3.6 — Структурна схема рекомендованої архітектури розгортання.

Сценарії застосування в системах автоматизації

На основі аналізу метрик моделі (п. 3.4) виділено три пріоритетні сценарії впровадження, де розроблена система принесе найбільший економічний ефект:

1. **Автоматизована пріоритезація тікетів підтримки (Support Desk Automation).**
    - *Проблема:* Оператори обробляють запити в порядку надходження (FIFO), ігноруючи критичні випадки.
    - *Рішення:* Система аналізує вхідні повідомлення. Якщо детектовано емоції *Anger* (Гнів) або *Sadness* (Сум) з імовірністю

вище порогу, тікет автоматично отримує статус "High Priority" і маршрутизується до старшого оператора.

- *Перевага TextCNN:* Висока повнота (Recall) для негативних емоцій мінімізує ризик ігнорування розлюченого клієнта.

2. **Інтелектуальна модерація контенту.**
    - *Проблема:* Ручна модерація коментарів є повільною та дорогою.
    - *Рішення:* Фільтрація повідомлень з мітками *Annoyance*, *Disapproval*, *Disgust*. Оскільки модель здатна розпізнавати неявну агресію (див. приклад у п. 3.4.4), вона ефективніша за прості фільтри нецензурної лексики.
    - *Налаштування:* Для цього сценарію рекомендується підвищити пороги спрацювання на 10-15%, щоб зменшити кількість хибних блокувань.
3. **Моніторинг репутації бренду (Brand Sentiment Monitoring).**
    - *Завдання:* Відстеження реакції користувачів на нові продукти.
    - *Рішення:* Агрегація статистики по класах *Admiration*, *Excitement*, *Disappointment*. Побудова динамічних часових рядів емоційного фону.

Регламент оновлення моделі (ML Ops)

Оскільки мова в інтернеті швидко змінюється (поява нових мемів, сленгу), систему необхідно періодично адаптувати. Рекомендується наступний регламент:

1. **Щомісячне донавчання (Fine-tuning):** Використання нових розмічених даних для оновлення ваг Embedding-шару.
2. **Щотижнева калібровка порогів:** Запуск алгоритму Threshold Tuning (п. 2.4) на "свіжій" валідаційній вибірці. Це дозволить автоматично

адаптувати чутливість системи до зміни балансу класів без втручання в код нейромережі.

Впровадження цих рекомендацій дозволить створити замкнений контур адаптивного керування якістю класифікації, що відповідає сучасним вимогам до інтелектуальних систем автоматизації.

Висновки до Розділу 3.

У третьому розділі кваліфікаційної роботи виконано програмну реалізацію та всебічне експериментальне дослідження компонентів автоматизованої системи аналізу емоційного забарвлення тексту.

Основні результати цього етапу роботи:

1. **Реалізовано програмний комплекс** мовою Python із використанням бібліотек TensorFlow/Keras та Scikit-learn. Підготовлено та очищено масштабний набір даних GoEmotions (~58 тис. повідомлень), виконано токенізацію та формування словника розміром 27 500 слів для навчання нейромережі.
2. **Навчено та протестовано дві моделі класифікації.**
    - Базова модель (Logistic Regression + TF-IDF) продемонструвала високу точність на класах з явною лексикою (Micro-F1 = 0.533), підтвердивши статус сильного еталона для коротких текстів.
    - Розроблена нейромережева модель (TextCNN) досягла співставної якості (Micro-F1 = 0.519), але продемонструвала кращу здатність до узагальнення контексту та вищу повноту (Recall) на мажоритарних класах.
3. **Експериментально підтверджено ефективність алгоритму адаптації порогів.** Доведено, що застосування процедури Threshold Tuning є критично необхідним для роботи з незбалансованими даними. Це дозволило системі детектувати рідкісні емоції (*Grief*, *Relief*), які

ігнорувалися при стандартному порозі $t = 0.5$, та забезпечити рівень Macro-F1 в діапазоні 0.44–0.46.

4. **Розроблено рекомендації щодо впровадження.** Запропоновано мікросервісну архітектуру розгортання підсистеми з використанням REST API, що дозволяє інтегрувати розроблений модуль у реальні контури автоматизації (CRM, служби підтримки) для пріоритезації задач та модерації контенту.

Отримані практичні результати повністю підтверджують працездатність спроектованої системи та створюють основу для подальшого удосконалення (наприклад, шляхом інтеграції трансформерів BERT для складних семантичних випадків).

ВИСНОВКИ

У магістерській кваліфікаційній роботі вирішено актуальну науково-прикладну задачу автоматизованого аналізу емоційного забарвлення текстових повідомлень в умовах невизначеності та дисбалансу даних. На основі проведених теоретичних та експериментальних досліджень отримано наступні результати:

1. **Виконано системний аналіз проблеми.** Встановлено, що задача визначення емоцій у коротких повідомленнях соціальних мереж (на прикладі датасету GoEmotions) належить до класу задач багатоміткової класифікації (Multi-label Classification) з суттєвим дисбалансом класів. Доведено, що використання стандартних підходів сентимент-аналізу та метрики Accuracy є неефективним для детектування рідкісних, але критично важливих емоційних станів (горе, страх, полегшення).
2. **Обґрунтовано вибір методів.** На основі порівняльного аналізу архітектур NLP визначено доцільність використання згорткових нейронних мереж (TextCNN) для обробки коротких текстів. Ця архітектура забезпечує ефективне виділення локальних n-грамних патернів без необхідності залучення важких обчислювальних ресурсів, що є критичним для систем реального часу. Як базовий еталон (baseline) обрано метод логістичної регресії з TF-IDF векторизацією.
3. **Спроектовано математичне та алгоритмічне забезпечення системи.** Розроблено структурну схему автоматизованої системи, яка включає модулі попередньої обробки, векторизації та класифікації. Ключовим інженерним рішенням стала розробка **адаптивного алгоритму оптимізації порогів (Threshold Tuning)**, який автоматично підбирає індивідуальну чутливість системи для кожного з 28 класів емоцій, максимізуючи метрику Macro-F1.

4. **Виконано програмну реалізацію.** Створено програмний комплекс мовою Python з використанням бібліотек TensorFlow/Keras та Scikit-learn. Реалізовано повний конвеєр обробки даних: від токенізації 58 тис. повідомлень до навчання моделей та серіалізації результатів. Запропоновано архітектуру розгортання системи у вигляді мікросервісу з REST API інтерфейсом.

5. **Проведено експериментальне дослідження.** Отримані результати підтвердили робочі гіпотези:

    - **Гіпотеза про архітектуру:** Нейромережева модель TextCNN досягла якості, порівнянної з сильним лінійним методом (Micro-F1: 0.519 проти 0.533), при цьому продемонструвавши вищу повноту (Recall) на мажоритарних класах та кращу здатність до узагальнення контексту.

    - **Гіпотеза про адаптацію:** Використання розробленого алгоритму оптимізації порогів дозволило нівелювати вплив дисбалансу даних. Для рідкісних класів оптимальні пороги було знижено до рівня 0.05–0.15, що забезпечило працездатність системи в умовах, де стандартний поріг 0.5 давав би нульову ефективність.

6. **Визначено практичну цінність.** Розроблена підсистема готова до впровадження в контури автоматизованого керування, зокрема для задач інтелектуальної пріоритезації тікетів у службах підтримки та автоматичної модерації токсичного контенту. Запропонований підхід забезпечує баланс між швидкістю обробки (інференс ~10 мс) та якістю класифікації (Macro-F1 ~0.44).

Перспективи подальших досліджень полягають у інтеграції попередньо навчених трансформерів (BERT/RoBERTa) для покращення розуміння семантично складних емоцій та розширенні набору даних для підтримки багатомовності.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бісікало О. В. Комп'ютерно-лінгвістичний аналіз та продукування україномовних текстів : монографія. Вінниця : ВНТУ, 2020. 280 с.
2. Висоцька В. А. Методи та засоби опрацювання інформаційних ресурсів в системах електронної контент-комерції : монографія. Львів : Вид-во Львівської політехніки, 2019. 324 с.
3. Глибовець М. М., Олецький О. В. Штучний інтелект : підручник. Київ : КМ-Академія, 2002. 366 с.
4. Голуб С. В., Куницька С. Ю. Інтелектуальна система аналізу тональності текстових повідомлень. *Вісник Черкаського державного технологічного університету*. 2019. № 3. С. 15–22.
5. Кунгурцев О. Б. Нейромережеві методи обробки природної мови в інформаційно-керуючих системах. *Системні дослідження та інформаційні технології*. 2021. № 2. С. 45–56.
6. Ланде Д. В., Фурашев В. М. Основи інформаційного та соціально-правового моделювання : монографія. Київ : ПанТот, 2012. 144 с.
7. Литвин В. В. Бази знань інтелектуальних систем підтримки прийняття рішень : підручник. Львів : Вид-во Львівської політехніки, 2011. 240 с.
8. Любченко В. В. Методи та алгоритми обробки природної мови : навч. посіб. Одеса : ОНПУ, 2022. 156 с.
9. Морозов А. О., Яковенко В. А. Інформаційно-аналітичні системи підтримки прийняття рішень на основі аналізу текстових даних. *Математичні машини і системи*. 2020. № 4. С. 112–120.
10. Субботін С. О. Нейронні мережі: теорія та практика : навч. посіб. Житомир : ЖДТУ, 2020. 184 с.

11. Ткаченко К. М. Порівняльний аналіз методів векторизації тексту для задач класифікації. *Вісник НТУУ «КПІ». Інформатика, управління та обчислювальна техніка*. 2023. Вип. 78. С. 34–40.

12. Шевченко А. І., Сальников А. М. Штучний інтелект: методи, засоби, технології : монографія. Донецьк : ІПШІ Наука і освіта, 2014. 364 с.

13. Щерба А. В. Методи класифікації текстової інформації в системах моніторингу соціальних мереж. *Штучний інтелект*. 2018. № 1. С. 88–96.

14. Aggarwal C. C. Machine Learning for Text. Cham : Springer, 2018. 493 p.

15. Chollet F. Deep Learning with Python. 2nd ed. Shelter Island : Manning Publications, 2021. 504 p.

16. Demszky D., Movshovitz-Attias D., Ko J., Cowen A., Nematzadeh G.,Raffel C. GoEmotions: A Dataset of Fine-Grained Emotions. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*. Online, 2020. P. 4040–4054. DOI: 10.18653/v1/2020.acl-main.372.

17. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of NAACL-HLT*. Minneapolis, 2019. P. 4171–4186.

18. Goodfellow I., Bengio Y., Courville A. Deep Learning. Cambridge : MIT Press, 2016. 800 p. URL: http://www.deeplearningbook.org (accessed: 20.10.2025).

19. Jurafsky D., Martin J. H. Speech and Language Processing : An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. 3rd ed. draft. Stanford, 2023. 650 p.

20. Kim Y. Convolutional Neural Networks for Sentence Classification. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, 2014. P. 1746–1751. DOI: 10.3115/v1/D14-1181.

21. Lipton Z. C., Elkan C., Naryanaswamy B. Optimal Thresholding of Classifiers to Maximize F1 Measure. *Machine Learning and Knowledge Discovery in Databases*. ECML PKDD 2014. Lecture Notes in Computer Science. Vol. 8725. Springer, 2014. P. 225–239.

22. Manning C. D., Raghavan P., Schütze H. Introduction to Information Retrieval. Cambridge : Cambridge University Press, 2008. 482 p.

23. Mikolov T., Chen K., Corrado G., Dean J. Efficient Estimation of Word Representations in Vector Space. *Proceedings of Workshop at ICLR*. Scottsdale, 2013. URL: https://arxiv.org/abs/1301.3781 (accessed: 15.09.2025).

24. Pedregosa F., Varoquaux G., Gramfort A. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*. 2011. Vol. 12. P. 2825–2830.

25. Pennington J., Socher R., Manning C. D. GloVe: Global Vectors for Word Representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, 2014. P. 1532–1543.

26. Vaswani A., Shazeer N., Parmar N. Attention Is All You Need. *Advances in Neural Information Processing Systems (NIPS)*. Long Beach, 2017. Vol. 30. P. 5998–6008.

27. Zhang Y., Wallace B. A. A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification. *Proceedings of the 8th International Joint Conference on Natural Language Processing (IJCNLP)*. Taipei, 2017. P. 253–263.

28. Zhang X., Zhao J., LeCun Y. Character-level Convolutional Networks for Text Classification. *Advances in Neural Information Processing Systems*. 2015. Vol. 28. P. 649–657.

ДОДАТКИ

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра комп'ютерних систем та робототехніки
Рівень вищої освіти (освітньо-кваліфікаційний рівень) ***Магістр***
Галузь знань: 17 – Електроніка, автоматизація та електронні комунікації
Спеціальність: 174 – Автоматизація, комп'ютерно-інтегровані технології та робототехніка
Освітня програма «Комп'ютеризовані системи управління та автоматика»

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних
систем та робототехніки
к. ф.-м. н., доц. ХРУСЛОВ М. М.
«02» жовтня 2024року

З А В Д А Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

ГОЛОЦВАНА Даніїла Ігоровича
(прізвище, ім'я, по батькові студента)

1. Тема роботи **«НЕЙРОМЕРЕЖЕВА МОДЕЛЬ АНАЛІЗУ ЕМОЦІЙНОГО ЗАБАРВЛЕННЯ ТЕКСТУ»**

керівник роботи **ЧУБ Ольга Ігорівна, доцент кафедри, кандидат економічних наук,**
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету ***від 30 вересня 2025 року № 4101-5/3554***

2. Строк подання студентом роботи ***30 листопада 2025 року***

3. Перелік питань, які потрібно розробити:

1. Аналіз сучасних методів автоматизованої обробки текстових даних для задач визначення емоційного забарвлення.
2. Огляд моделей глибинного навчання для класифікації коротких текстів, зокрема згорткових нейронних мереж (CNN).
3. Проектування комп'ютерної системи емоційного аналізу з модулем адаптивного налаштування порогів класифікації.
4. Програмна реалізація комп'ютерної системи та нейромережевої моделі.
5. Експериментальне дослідження ефективності системи та порівняльний аналіз точності моделей.

4. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1	Затвердження теми роботи та керівника	02.09.2024 - 02.10.2024
2	Аналіз та пошук методичної літератури щодо сучасних методів обробки природної мови та задач емоційного аналізу	03.09.2024 - 24.10.2024
3	Огляд і порівняльний аналіз архітектур згорткових нейронних мереж (CNN, TextCNN) та методів векторизації для класифікації текстів	25.10.2024 - 09.11.2024
4	Обґрунтування вибору програмних засобів та підготовка набору даних GoEmotions (токенізація, очищення)	10.11.2024 - 24.11.2024
5	Розробка математичної моделі TextCNN та проведення експериментів з налаштування гіперпараметрів	25.11.2024 - 08.12.2024
6	Програмна реалізація модулів навчання нейромережі та алгоритму адаптивної оптимізації порогів класифікації	09.12.2024 - 29.01.2025
7	Проектування підсистеми автоматизованого моніторингу емоційного стану для інтеграції в інформаційно-керуючі системи	30.01.2025 - 28.02.2025
8	Тестування системи, порівняльний аналіз з базовим методом та валідація моделі за метриками F1-score	01.03.2025 - 01.04.2025
9	Підготовка і оформлення звітних матеріалів (чорновий варіант пояснювальної записки)	01.05.2025 - 30.08.2025
10	Оформлення звіту про науково-дослідну практику. Написання статті за матеріалами кваліфікаційної роботи	01.09.2025 - 30.10.2025
11	Підготовка і оформлення звітних матеріалів кваліфікаційної роботи. Оформлення списку літератури	10.10.2025 - 30.10.2025
12	Оформлення пояснювальної записки кваліфікаційної роботи відповідно вимогам до звітів про НДР	10.10.2025 - 30.11.2025
13	Підготовка і оформлення звітних матеріалів та додатків кваліфікаційної роботи	01.11.2025 - 30.11.2025
14	Оформлення звіту про переддипломну практику	24.11.2025 - 30.11.2025
15	Представлення кваліфікаційної роботи керівнику та рецензенту	24.11.2025 - 30.11.2025

5. Дата видачі завдання ***02 жовтня 2024 року.***

Студент Д. І. Голоцван _____
ініціали, прізвище підпис

Керівник роботи О. І. Чуб _____
ініціали, прізвище підпис

Додаток Б

Затверджую

_____

«_____»_____2025 р.

Технічне завдання
на розробку програмного виробу
«Нейромережева модель для аналізу емоційного забарвлення тексту»

1	**Вступ**	**1.1. Назва:** Нейромережева модель аналізу емоційного забарвлення тексту **1.2. Галузь застосування:** комп'ютерні технології.
2	**Підстава для розробки**	**2.1.** Навчальний план за спеціальністю 151 – Автоматизація та комп'ютерно-інтегровані технології **2.2.** Завдання на кваліфікаційну роботу магістра № 4101-5/3554 від «30» вересня 2025 р. (представити як Додаток А до пояснювальної записки до кваліфікаційної роботи).
3.	**Призначення розробки**	**3.1. Мета розробки:** створення та програмна реалізація моделі автоматизованої системи для багатоміткової класифікації емоційного забарвлення коротких текстових повідомлень з використанням згорткових нейронних мереж (TextCNN) та алгоритму адаптивних порогів. **3.2. Призначення розробки:** система призначена для автоматизації моніторингу контенту в соціальних мережах, виявлення токсичних або критичних емоційних станів (гнів, горе), пріоритезації звернень у службах підтримки та аналізу репутації бренду. **3.3. Вихідні дані розробки:** неструктуровані текстові повідомлення (коментарі) англійською мовою; набір даних GoEmotions для навчання (58 тис. записів, 28 класів).
4.	**Технічні вимоги до програмного виробу**	**4.1. Вимоги до функціональних характеристик:** • Прийом текстових даних через програмний інтерфейс (API); • Попередня обробка тексту (очищення від тегів, токенізація, паддінг до 128 символів); • Класифікація тексту за 28 категоріями емоцій за допомогою моделі TextCNN; • Застосування індивідуальних порогів для кожного класу (Threshold Tuning); • Формування відповіді у форматі JSON з переліком виявлених емоцій та їх ймовірностей. **4.2. Вимоги до надійності:** • Інтегральна точність (Micro-F1) не нижче 0.50; • Чутливість (Macro-Recall) для рідкісних класів не гірше 0.30; • Час обробки одного запиту (інференс) не більше 50 мс. **4.3. Вимоги до умов експлуатації:** • Серверна частина: ОС Linux (Ubuntu 20.04+), середовище Python 3.10+; • Наявність бібліотек: TensorFlow 2.x, NumPy, Pandas. **4.4. Вимоги до складу і параметрів технічних засобів:** • Процесор: не менше 2 ядер (2.5 GHz);

<table>
<tr><td></td><td></td><td colspan="2">• Оперативна пам'ять: не менше 8 GB;
• Дисковий простір: не менше 2 GB.
4.5. Вимоги до інформаційної та програмної сумісності:
• Формати обміну даними: JSON;
• Кодування тексту: UTF-8.
4.6. Вимоги до маркування та упаковки: Не висуваються.
4.7. Вимоги до транспортування і зберігання: Зберігання у репозиторії коду (GitHub/GitLab) або Docker-контейнері.
4.8. Спеціальні вимоги: Забезпечення анонімізації вхідних даних (видалення персональних імен та нікнеймів).</td></tr>
<tr><td>5.</td><td>Вимоги до програмної документації</td><td colspan="2">Програмною документацією до виробу «Нейромережева модель аналізу емоційного забарвлення тексту» вважати:
1) Дане Технічне завдання (представити у вигляді Додатку Б до пояснювальної записки);
2) Опис математичної моделі TextCNN та алгоритму оптимізації порогів (у вигляді розділу 2 пояснювальної записки);
3) Опис програмної реалізації та інструкцію з розгортання (у вигляді розділу 3 пояснювальної записки)</td></tr>
<tr><td>6.</td><td>Вимоги до техніко-економічних показників</td><td colspan="2">1) Зменшення часу на модерацію контенту: з хвилин (ручний режим) до мілісекунд (автоматичний режим).
2) Підвищення якості детектування рідкісних емоцій (Macro-F1) на 10-15% порівняно з моделями без адаптивних порогів.
3) Відсутність витрат на пропрієтарне ПЗ (використання Open Source рішень).</td></tr>
<tr><td>7.</td><td>Стадії і етапи розробки</td><td>Дата</td><td>Назва етапу</td></tr>
<tr><td></td><td></td><td>02.09.2024 - 02.10.2024</td><td>• Затвердження теми роботи та керівника</td></tr>
<tr><td></td><td></td><td>03.09.2024 - 24.10.2024</td><td>• Аналіз та пошук методичної літератури щодо сучасних методів обробки природної мови та задач емоційного аналізу</td></tr>
<tr><td></td><td></td><td>25.10.2024 - 09.11.2024</td><td>• Огляд і порівняльний аналіз архітектур згорткових нейронних мереж та методів векторизації для класифікації текстів</td></tr>
<tr><td></td><td></td><td>10.11.2024 - 24.11.2024</td><td>• Обґрунтування вибору програмних засобів та підготовка набору даних GoEmotions</td></tr>
<tr><td></td><td></td><td>25.11.2024 - 08.12.2024</td><td>• Розробка математичної моделі TextCNN та проведення експериментів з налаштування гіперпараметрів</td></tr>
<tr><td></td><td></td><td>09.12.2024 - 29.01.2025</td><td>• Програмна реалізація модулів навчання нейромережі та алгоритму</td></tr>
</table>

			адаптивної оптимізації порогів класифікації
		30.01.2025 - 28.02.2025	• Проектування підсистеми автоматизованого моніторингу емоційного стану для інтеграції в інформаційно-керуючі системи
		01.03.2025 - 01.04.2025	• Тестування системи, порівняльний аналіз з базовим методом та валідація моделі за метриками F1-score
		01.05.2025 - 30.08.2025	• Підготовка і оформлення звітних матеріалів
		01.09.2025 - 30.10.2025	• Оформлення звіту про науково-дослідну практику. Написання статті за матеріалами кваліфікаційної роботи
		10.10.2025 - 30.10.2025	• Підготовка і оформлення звітних матеріалів кваліфікаційної роботи. Оформлення списку літератури
		10.10.2025 - 30.11.2025	• Оформлення пояснювальної записки кваліфікаційної роботи відповідно вимогам до звітів про НДР
		01.11.2025 - 30.11.2025	• Підготовка і оформлення звітних матеріалів та додатків кваліфікаційної роботи
		24.11.2025 - 30.11.2025	• Оформлення звіту про переддипломну практику
		24.11.2025 - 30.11.2025	• Представлення кваліфікаційної роботи керівнику та рецензенту
8.	**Порядок контролю і приймання програмного продукту**	1) Перевірку ходу розробки виконувати згідно з календарним планом. 2) Захист розробленої системи провести на засіданні Екзаменаційної комісії. 3) Пояснювальну записку подати на паперових носіях та в електронному вигляді згідно з вимогами ВНЗ.	

Виконавець:

студент групи КУ-61

Голоцван Д. І.

Замовник:

к. е. н., доцент

Чуб О. І.

Додаток В

Програма і методика випробувань

програмного виробу

«Нейромережева модель для аналізу емоційного забарвлення тексту»

1. Об'єкт випробувань.

1. **Назва програмного виробу:** «Нейромережева модель аналізу емоційного забарвлення тексту».
2. **Галузь застосування:** Системи моніторингу соціальних мереж, автоматизована модерація контенту, аналітика клієнтського досвіду.
3. **Відомості запозичуються** з відповідних розділів Технічного завдання.

2. Мета випробувань. Перевірка відповідності функціональності програмної реалізації системи (нейромережевої моделі TextCNN та модуля адаптивних порогів) заявленим функціональним можливостям та метрикам якості в Технічному завданні (Додаток Б до пояснювальної записки до дипломної роботи).

3. Загальні положення.

1. **Підстави для проведення випробувань:** Підставою для проведення випробувань є наказ про призначення атестаційної комісії.
2. **Місце і тривалість випробувань:** Приймальні (приймально-здавальні) випробування проводяться на базі комп'ютерного класу кафедри в період роботи атестаційної комісії.
3. **Обсяг випробувань:** Приймальні випробування програмного виробу проводяться в обсязі, відповідному цій програмі і методиці випробувань.
4. **Організації, які беруть участь у випробуваннях:** Приймальні випробування проводяться атестаційною комісією напередодні засідання (або в процесі засідання) за участю Замовника, Виконавця та інших осіб, присутніх на засіданні.

4. Вимоги до програми або програмного виробу. Модель повинна задовольняти наступним вимогам:

1. Працювати на основних операційних системах: Windows (через Anaconda/Python), Linux (Ubuntu);
2. Забезпечувати інтегральну точність класифікації (Micro-F1) не нижче 0.50;
3. Коректно завантажувати ваги нейромережі (.keras) та файл порогів (.npy);
4. Приймати на вхід текстові дані англійською мовою;
5. Забезпечувати час обробки одного повідомлення (інференс) не більше 50-100 мс;
6. Бути сумісною з бібліотеками TensorFlow 2.x та NumPy;
7. Елементи програми (препроцесинг, модель, постпроцесинг) повинні бути модульно розділені;
8. Вимоги до складу і параметрів технічних засобів (відповідно до ТЗ);
9. Вимоги до маркування та упаковки (не висуваються);
10. Вимоги до транспортування і зберігання (не висуваються).

5. Вимоги до програмної документації. Програмною документацією до виробу «Нейромережева модель аналізу емоційного забарвлення тексту» вважати:

1. Справжнє Технічне завдання на розробку (представити як Додаток Б до пояснювальної записки до кваліфікаційної роботи);
2. Програму і методику випробувань розробленої програми (представити як Додаток В до пояснювальної записки до кваліфікаційної роботи);
3. Рекомендації щодо застосування створеної системи (представити в Розділі 3 пояснювальної записки до кваліфікаційної роботи).

6. Засоби і порядок випробувань

6.1. Засоби випробувань Для проведення випробувань необхідний персональний комп'ютер із встановленим інтерпретатором Python версії 3.10+, бібліотеками TensorFlow/Keras, Pandas, NumPy та середовищем розробки (Jupyter Notebook або VS Code).

6.2. Порядок проведення випробувань: Як правило, випробування проводяться в два етапи: · ознайомчий (1-й етап); · випробування програмного виробу (2-й етап).

Перелік перевірок, що проводяться на 1 етапі випробувань:

1. Перевірка комплектності програмної документації.
2. Перевірка наявності файлів моделі (textcnn_best.keras), токенізатора (tokenizer.json) та порогів (thresholds.npy).
3. Якість програмної документації перевіряється на відповідність вимогам стандартів ЕСПД.

Перелік перевірок, що проводяться на 2 етапі випробувань:

1. Перевірка відповідності технічних характеристик моделі вимогам технічного завдання (розмір вхідного вектора 128, кількість виходів 28).
2. Перевірка ступеня виконання функціональних вимог (класифікація довільного тексту).
3. Програма працює стабільно без критичних помилок (Runtime Errors).

Порядок проведення функціональних тестів: 3.1. Запуск середовища виконання здійснюється відкриттям файлу проекту (Jupyter Notebook або скрипт predict.py). 3.2. Ініціалізація моделі (завантаження ваг у пам'ять). 3.3. Виконання тестових прикладів класифікації.

Тест 1. Перевірка ініціалізації системи

1. Перевірка виконання програми.
2. Користувач запускає скрипт ініціалізації (імпорт бібліотек та load_model).
3. Отримання повідомлення про успішне завантаження моделі та відсутність помилок сумісності версій.

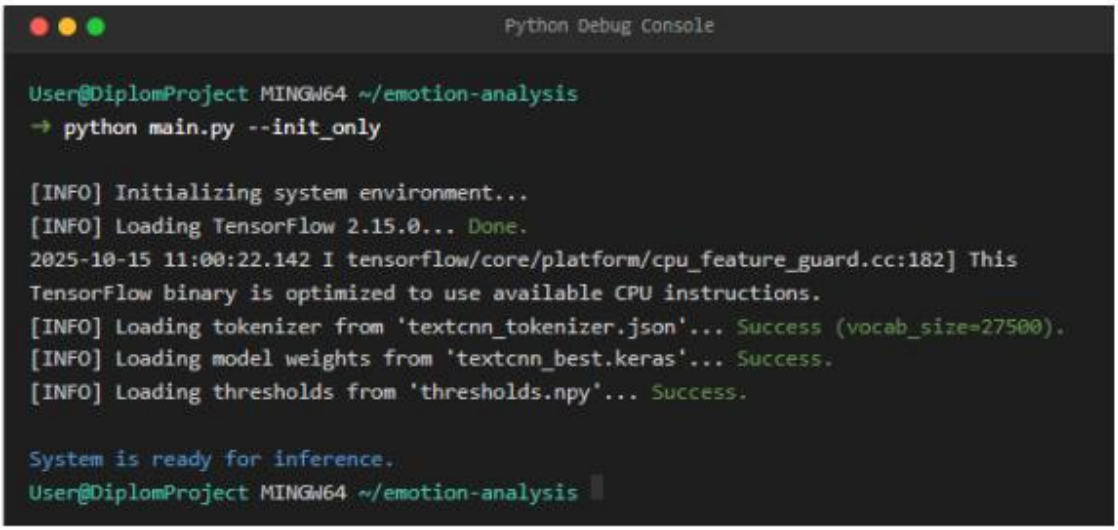

Рисунок В.1 – Тест 1.

Тест 2. Класифікація коректного текстового повідомлення

1. Перевірка виконання програми.
2. Користувач подає на вхід тестову фразу, наприклад: *"I am so happy th you came!"*.
3. Отримання результату у вигляді списку емоцій з ймовірностями, ш перевищили поріг (наприклад: *Joy, Excitement*). Перевірка адекватнос результату контексту.

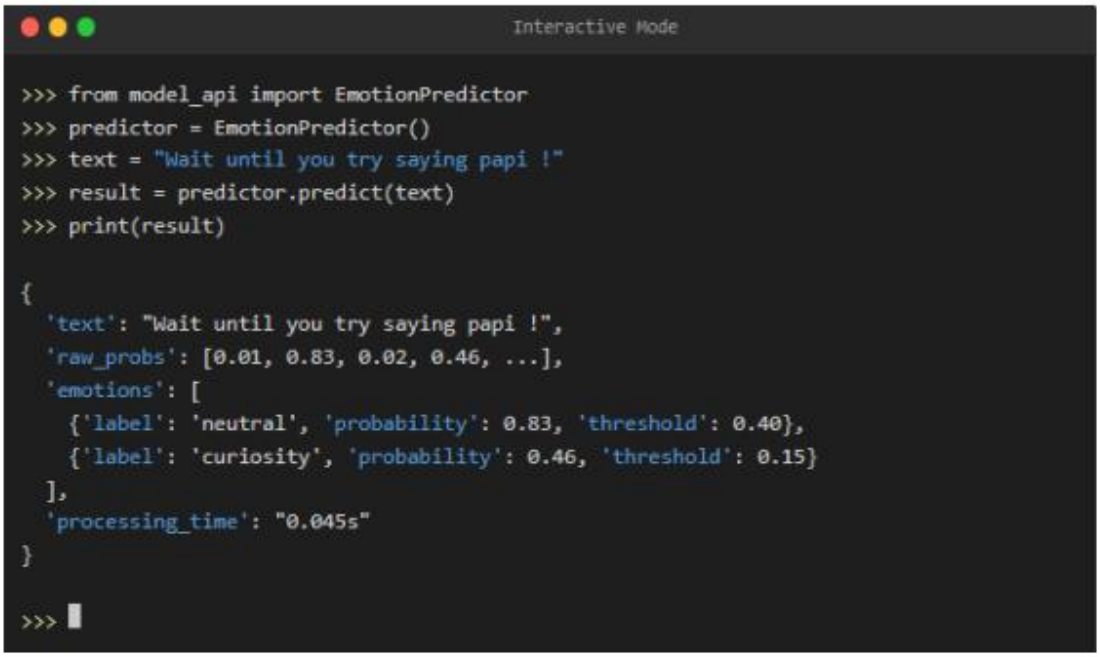

Рисунок В.2 – Тест 2.

Тест 3. Обробка некоректних даних (Стійкість)

1. Перевірка виконання програми.
2. Користувач подає на вхід пустий рядок або занадто короткий текст (менше 3 символів).
3. Отримання коректного повідомлення системи про неможливість класифікації або повернення порожнього списку емоцій (без падіння програми).

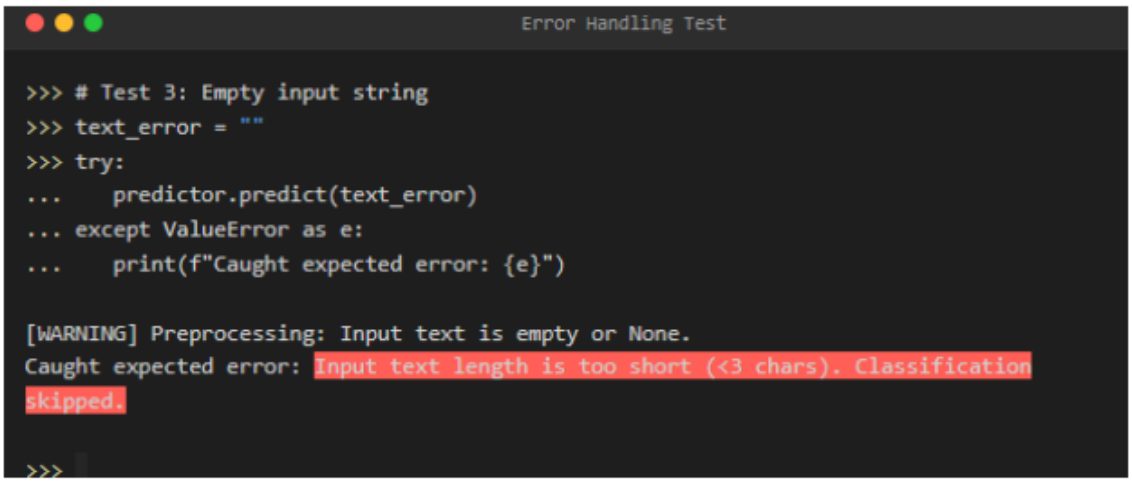

Рисунок В.3 – Тест 3.

Висновки: Тест 1 успішно пройшов випробування, Тест 2 успішно пройшов випробування і Тест 3 успішно пройшов випробування. Випробування пройшло успішно, програмний виріб відповідає вимогам Технічного завдання.

Виконавець: студент групи КУ61, Голоцван Д. І.