КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра технологій управління

Спеціальність 122 – Комп'ютерні науки, освітня програма «Інформаційна аналітика та впливи»

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему:

"Розробка технології емоційного аналізу за допомогою методів науки про дані та алгоритмів машинного навчання**"**

Студента 2-го курсу групи ІАВ-21

Пелішенка Владислава Святославовича
(прізвище, ім'я, по батькові)

_____
(підпис студента)

Науковий керівник:

К.е.н., доцент к.т.у
(науковий ступінь, вчене звання)

Мірошниченко Ігор Вікторович
(прізвище, ім'я, по батькові)

_____ _____
(дата) (підпис)

Попередній захист:		
(Висновок: «До захисту в Екзаменаційній комісії»)		
Завідувач кафедри технологій управління _____ _____		
(підпис)	(прізвище, ініціали)	(дата)

Київ – 2025

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій

Кафедра технологій управління
Освітньо-кваліфікаційний рівень Магістр
Спеціальність 122 – Комп'ютерні науки
Освітня програма Інформаційна аналітика та впливи

ЗАТВЕРДЖУЮ

Завідувач кафедри,

професор

Морозов В.В.

_____

«_____»

2025 року

ЗАВДАННЯ НА ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Студент Пелішенко Владислав Святославович

Група __ІАВ-11__

1. Тема кваліфікаційної роботи

Розробка технології емоційного аналізу за допомогою методів науки про дані та алгоритмів машинного навчання_____

_____

Затверджено наказом від «___» _____ 2025 р. № _____

2. Строк подання студентом готової роботи – "___" _____ 2025 р.

3. Цільова установка та вихідні дані до роботи

Здійснити аналіз вихідних даних на їхню збалансованість, провести процес балансування класів за необхідності. Використати сучасні технології класифікації текстових даних, а саме методи глибокого навчання для розробки технології емоційного аналізу. Вихідні дані включають текстові повідомлення користувачів та завчасно визначені класи.

4. Зміст роботи

Зміст роботи передбачає аналіз літературних джерел щодо методів емоційного аналізу, а саме аналіз методів класифікації, які включають методи машинного навчання та глибокого навчання, пошук та обробку вхідних даних, розробку покращеної моделі класифікації на основі моделі глибокого навчання LSTM. Окрім цього передбачено проведення тестування розробленї моделі, створення інструкції з розгортання розробленої моделі за допомогою хмарних сервісів та аналіз перспектив подальших досліджень.

5. Перелік графічних матеріалів (слайдів)

Загалом робота містить 17 рисунків, слайдів. Перелік слайдів: актуальність обраної теми (1 слайд), об'єкт та предмет (1 слайд), мета та задачі дослідження (1 слайд), обрана модель (1 слайд), процес побудови моделі (2 слайди), оцінка моделі(3 слайди), отримані результати дослідження (1 слайд), висновки (1 слайд).

6. Календарний план виконання роботи

№ з/п	Назва частин роботи	%	Виконання роботи	
			За планом	Фактично

Дата видачі завдання «___» _____ 2025 р.

Керівник роботи доцент к.т.у., Мірошниченко Ігор Вікторович
(посада, прізвище, ім’я, по батькові)

_____
(підпис)

Завдання прийняв до виконання студент групи _____

_____
(прізвище, ім’я, по батькові)

_____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва частин роботи	%	Виконання роботи	
			За планом	Фактично
1.	Вибір теми дипломної роботи	3	01.10.24	01.10.24
2.	Протокол кафедри ТУ про затвердження тем дипломних робіт та призначення наукових керівників	2	27.12.24	27.12.24
3.	Формування переліку нормативних матеріалів, літератури з проблематики дипломної роботи	10	08.01.25	07.01.25
4.	Складання розгорнутого плану кваліфікаційної роботи	5	18.01.25	18.01.25
5.	Ознайомлення наукового керівника з розгорнутим планом кваліфікаційної роботи. Внесення змін.	5	19.01.25 - 20.01.25	20.01.25
6.	Підготовка розділу 1 «Історія, розвиток та аналіз існуючих рішень для емоційного аналізу»	10	12.02.25	13.02.25
7.	Підготовка розділу 2 «Формалізація методу емоційного аналізу»	14	08.03.25	08.03.25
8.	Підготовка розділу 3 «Розробка методу емоційного аналізу»	14	20.03.25	20.03.25
9.	Підготовка розділу 4 «Впровадження розробленої технології»	13	15.04.25	15.04.25
10.	Оформлення кваліфікаційної роботи. Підготовка висновків і пропозицій	15	25.04.25	25.04.25
11.	Передача кваліфікаційної роботи науковому керівникові	2	01.05.25	01.05.25
12.	Передача кваліфікаційної роботи рецензенту для рецензування	2	04.05.25	04.05.25
13.	Попередній захист кваліфікаційної роботи	5	10.05.25	10.05.25

ЗМІСТ

АНОТАЦІЯ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра технологій управління
Спеціальність – 122, Комп'ютерні науки, освітня
програма "Інформаційна аналітика та впливи"

Дипломна робота магістра Пелішенка Владислава Святославовича

Тема роботи – «Розробка технології емоційного аналізу за допомогою методів науки про даті та алгоритмів машинного навчання».

Мета дипломної роботи магістра – розробка покращеної технології емоційного аналізу за допомогою методів науки про дані та алгоритмів машинного навчання.

Об'єкт дослідження – процеси емоційного аналізу користувачів.

Предмет дослідження – методи науки про дані та алгоритми машинного навчання для емоційного аналізу.

Наукова новизна роботи – розроблено удосконалену технологію емоційного аналізу за допомогою алгоритмів машинного навчання, яка відрізняється від існуючих впровадженням завчасно натренованого корпусу векторних представлень слів, регуляризатора та вкладання слів.

У роботі досліджуються існуючі підходи до емоційного аналізу за допомогою алгоритмів машинного навчання. Розробляється удосконалена технологія емоційного аналізу за допомогою нейронних мереж, а також проводиться обґрунтування доцільності впровадження запропонованої технології. Наводяться рекомендації щодо практичного впровадження технології.

Дипломна робота складається зі вступу, основної частини, яка включає чотири розділи, висновків та списку використаних джерел. Всього налічує 92 сторінки, перелік посилань з 36 джерел на 4 сторінках та 3 додатків.

Ключові слова: емоційний аналіз, нейронні мережі, обробка природньої мови, навчання з розширеною пам’яттю.

ПЕРЕЛІК ВИКОРИСТАНИХ СКОРОЧЕНЬ

LSTM – Long-Short Term Memory

RNN – Recurrent Neural Network

КРМ – кваліфікаційна робота магістра

HR – Human Resources

NLP – Natural Language Processing

GloVe - Global Vectors for Word Representation

MCC - Matthews Correlation Coefficient

MAL – Memory Augmented Learning

GRU - Gated Recurrent Units

ВСТУП

У сучасному інформаційному суспільстві, де щодня генерується величезна кількість текстових даних у соціальних мережах, блогах, відгуках та коментарях, надзвичайно важливою стає здатність автоматизовано визначати емоційний зміст повідомлень. Ефективний емоційний аналіз відкриває широкі можливості для вдосконалення сервісів зворотного зв’язку, моніторингу громадської думки, виявлення соціальних настроїв, а також для створення інтелектуальних систем підтримки прийняття рішень.

Традиційні методи обробки текстової інформації, що базуються на ручному аналізі або використанні простих евристичних підходів, виявляються недостатньо гнучкими та масштабованими в умовах стрімкого зростання обсягів даних. У зв’язку з цим особливого значення набувають методи науки про дані та алгоритми машинного навчання, які дозволяють виявляти приховані закономірності у великих текстових корпусах та здійснювати високоточний аналіз емоційного стану користувачів.

Застосування технологій глибокого навчання забезпечує можливість моделювати складні нелінійні залежності між текстовими характеристиками та емоційними ознаками, що значно підвищує якість автоматичного емоційного аналізу. Завдяки цьому підходу можна створювати моделі, здатні не лише розпізнавати базові емоції, але й відрізняти тонкі відтінки емоційних станів, що важливо для практичних застосувань у сфері маркетингу, охорони здоров’я, освіти, кібербезпеки та розробки користувацьких сервісів.

У зв'язку з цим дослідження та розробка технологій емоційного аналізу текстів на основі сучасних аналітичних і обчислювальних підходів є надзвичайно актуальною та перспективною темою, що має як наукову, так і прикладну цінність.

Метою даної кваліфікаційної роботи магістра є розробка покращеної технології автоматичного емоційного аналізу тексту із застосуванням методів

науки про дані та алгоритмів машинного навчання, зокрема технологій глибокого навчання.

Завдання кваліфікаційної робот магістра – оцінити практичну сферу застосування методології науки про дані, визначити завдання проєкту, провести аналіз літератури, розробити модель мовою програмування Python, яка буде виконувати задачу емоційного аналізу тексту, провести тестування розробленої моделі та оформити отримані результати у формі звіту.

Об’єктом дослідження у кваліфікаційній роботі є процеси автоматизованого емоційного аналізу текстової інформації на підприємствах.

Предметом дослідження є методи науки про дані та алгоритми машинного навчання, орієнтовані на обробку тексту та аналіз емоційного контексту.

До основних методів дослідження, використаних у роботі, належать:

- збір і попередня обробка текстових даних;
- токенізація та векторизація тексту із застосуванням попередньо натренованих моделей (наприклад, GloVe);
- побудова, навчання та оптимізація моделей машинного навчання, включаючи застосування регуляризації та спеціалізованих функцій втрат для підвищення точності класифікації;
- порівняльний аналіз результатів і тестування на незалежних даних.

Наукова новизна одержаних результатів полягає в розробці моделі глибокого навчання, що удосконалює раніше розроблені рішення за допомогою використання завчасно сформованого корпусу векторизованих слів, регуляризації та ембеддингу для підвищення ефективності роботи.

Практична цінність одержаних результатів полягає в отриманні удосконаленої моделі емоційного аналізу тексту з вищою ефективністю роботи. Розроблена модель може бути використана на підприємствах для аналізу відгуку

цільової аудиторії на впроваження змін і прийняття відповідних до результатів рішень.

Результати роботи апробовано на Міжнародній науковій конференції *Information Technology and Implementation(Sattelite): Conference Proceedings,* 21 листопада 2024 року, м. Київ; за підсумками представлено одну публікацію.

РОЗДІЛ 1. КЛАСИФІКАЦІЯ ТА АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ ДЛЯ ЕМОЦІЙНОГО АНАЛІЗУ

1.1. Класифікація підходів до емоційного аналізу

Емоційний аналіз тексту (sentiment analysis або opinion mining) охоплює низку методів, спрямованих на автоматичне визначення емоційного змісту повідомлень. Існуючі підходи до емоційного аналізу можна умовно поділити на кілька основних категорій: лексиконно-орієнтовані методи, методи на основі правил, класичні алгоритми машинного навчання, глибоке навчання, векторні представлення слів та контекстуальні трансформерні моделі.

1. Лексиконно-орієнтовані методи

Це найпростіший та найраніший підхід, що ґрунтується на використанні заздалегідь підготовлених емоційних словників, в яких кожному слову приписується певна полярність (позитивна, негативна, нейтральна) або конкретна емоція. Оцінка тексту здійснюється шляхом підрахунку відповідних слів.

Основні словники:

1. SentiWordNet – надає кожному слову оцінку позитивності, негативності та об'єктивності;
2. WordNet-Affect – додає до WordNet позначки афективної категорії (страх, гнів, радість тощо);
3. NRC Emotion Lexicon – класифікує слова за 8 базовими емоціями та двома полярностями.

Переваги:

1. Простота реалізації;
2. Не потребує навчання моделей.

Недоліки:

1. Не враховується контекст;
2. Складнощі з новими словами, сленгом, сарказмом, полісемією.

2. Методи на основі правил

Цей підхід поєднує словникові дані з граматичними та логічними правилами для аналізу структури речення. Наприклад, враховуються:

1. Негатори (заперечення змінює полярність: «не хороший»);
2. Підсилювачі та пом'якшувачі (дуже, надзвичайно, трохи);
3. Пунктуація, емодзі, великі літери.

Приклад реалізації:

— VADER (Valence Aware Dictionary and sEntiment Reasoner) — адаптований для текстів із соціальних мереж, враховує специфіку неформального мовлення.

3. Класичне машинне навчання

Методи цієї категорії використовують алгоритми класифікації, яким подаються векторизовані тексти на основі частот слів (Bag-of-Words, TF-IDF тощо).[31]

Популярні алгоритми:

1. Наївний баєсівський класифікатор (Naive Bayes);
2. Метод опорних векторів (SVM);
3. Логістична регресія;
4. Рішення дерев/Random Forest.

Переваги:

1. Гнучкість у роботі з різними джерелами текстів;
2. Можливість налаштування параметрів моделі.

Недоліки:

1. Не враховується порядок слів;
2. Вимагає ручної інженерії ознак.

4. Глибоке навчання (Deep Learning)

Глибокі нейронні мережі дозволяють автоматично вивчати представлення тексту без явної векторизації чи лексиконів.

Основні моделі:

1. RNN (Recurrent Neural Network) – враховує порядок слів;
2. LSTM (Long Short-Term Memory) – краще працює з довгими послідовностями;
3. CNN (Convolutional Neural Network) – виділяє локальні шаблони в тексті.

Переваги:

1. Краще розуміння контексту;
2. Вища точність;
3. Автоматичне навчання ознак.

Недоліки:

1. Потребують великих обсягів даних і ресурсів;
2. Складність налагодження та інтерпретації.

5. Векторні представлення слів (word embeddings)

Для покращення якості розпізнавання смислу слів застосовуються розподілені векторні представлення, що дозволяють моделі враховувати семантичну схожість між словами.

Найвідоміші моделі:

1. Word2Vec – базується на контексті вікна слів;
2. GloVe – використовує глобальну статистику спільного вживання;
3. FastText – враховує морфологічну структуру слів.

Ці вектори слів зазвичай передаються на вхід класичним або нейронним моделям.

6. Контекстуальні трансформери

Починаючи з 2018 року, нове покоління моделей на базі трансформерної архітектури радикально підвищило якість емоційного аналізу.[11]

Основні моделі:

1. BERT (Bidirectional Encoder Representations from Transformers);
2. RoBERTa, DistilBERT, GPT, XLNet.

Переваги:

1. Повноцінне розуміння контексту;
2. Висока точність навіть без донавчання;
3. Можливість доадаптації на специфічних корпусах (fine-tuning).

Ці моделі стали новим стандартом у багатьох NLP-задачах, зокрема й в емоційному аналізі.

Підходи до емоційного аналізу пройшли шлях від простих лексиконів до глибоко контекстуальних трансформерів. Кожен з методів має свої переваги та недоліки і вибір конкретного підходу залежить від поставленого завдання, обсягу даних, необхідної точності та доступних ресурсів. Сучасні системи емоційного аналізу нерідко комбінують кілька підходів, включаючи гібридні та мультимодальні рішення.

1.2. Аналіз існуючих рішень для емоційного аналізу

Для подальшої роботи над розробкою рішенння для емоційного аналізу тексту необхідно провести аналіз наявих сучасних рішень даної проблеми. Останніми роками значного поширення набули рішення, що базуються на методах машинного та глибокого навчання. Зі зростанням обсягів неструктурованих текстових даних та вдосконаленням обчислювальних потужностей, такі підходи стали основою для побудови ефективних моделей, здатних виявляти емоційні ознаки в мовленні з високою точністю. На відміну від традиційних підходів, сучасні алгоритми можуть автоматично навчатися на великих обсягах даних, виявляти складні мовні шаблони та адаптуватися до нових контекстів. Тому під час дослідження нинішнього стану існуючих рішень було розглянуто методи машинного та глибокого навчання. Найбільш поширеними рішеннями, які використовують глибоке навчання та нейронні мережі є:

LSTM - тип рекурентних нейронних мереж (RNN), здатний зберігати довготривалі залежності в послідовностях даних. Він складаються з осередків пам'яті, які контролюють потік інформації через механізми затворів, дозволяючи моделі ефективно обробляти послідовні дані, такі як текст.

Переваги:

1. Ефективно моделюють довготривалі залежності в тексті.
2. Підходять для обробки послідовних даних, таких як природна мова.

Недоліки:

1. Високі обчислювальні витрати та тривалий час навчання.
2. Складність у паралельній обробці даних.

BiLSTM - складається з двох LSTM-мереж, що обробляють послідовність даних у прямому та зворотному напрямках. Це дозволяє моделі враховувати як

попередній, так і наступний контекст для кожного елемента послідовності, що особливо корисно для задач емоційного аналізу .

Переваги:

1. Краще розуміння контексту завдяки двонапрямленій обробці.
2. Покращена точність у порівнянні з однонапрямленими моделями.

Недоліки:

1. Збільшена складність моделі та потреба в більшій кількості ресурсів.
2. Триваліший час навчання.

CNN для тексту – модель глибокого навчання, що використовує згорткові шари для виявлення локальних шаблонів у тексті, таких як n-грамні фрази. Вони застосовують фільтри для виявлення ключових особливостей у фіксованих вікнах тексту, що дозволяє ефективно класифікувати короткі текстові фрагменти.

Переваги:

1. Швидке навчання та ефективна обробка даних.
2. Добре виявляють локальні залежності в тексті.

Недоліки:

1. Обмежена здатність моделювати довготривалі залежності.
2. Менш ефективні для аналізу довгих текстів.

BERT - трансформерна модель, яка навчається передбачати замасковані слова в реченні, використовуючи двонапрямлений контекст. Це дозволяє моделі глибоко розуміти значення слів у контексті всього речення [1].

Переваги:

1. Висока точність у задачах емоційного аналізу.
2. Можливість донавчання на специфічних наборах даних.

Недоліки:

1. Високі вимоги до обчислювальних ресурсів.
2. Складність у налаштуванні та інтеграції.

RoBERTa - покращена версією BERT, яка оптимізує процес навчання шляхом використання більшого обсягу даних і усунення обмежень, таких як передбачення наступного речення. Це дозволяє досягти кращої продуктивності в задачах емоційного аналізу .

Переваги:

1. Покращена точність порівняно з оригінальним BERT.
2. Краще узагальнення на нових даних.

Недоліки:

1. Ще більші вимоги до ресурсів, ніж у BERT.
2. Складність у розгортанні для реальних застосувань.

Трансформери використовують механізм самопильності (self-attention) для обробки всіх слів у послідовності одночасно, що дозволяє моделі враховувати залежності між будь-якими словами незалежно від їх відстані.

Переваги:

1. Здатність моделювати довготривалі залежності.
2. Паралельна обробка даних.

Недоліки:

1. Високі вимоги до обчислювальних ресурсів.
2. Потреба в великих обсягах даних для навчання.

Також для емоційного аналізу тексту використовуються такі класичні методи машинного навчання як:

Наївний Байєсівський класифікатор - базується на теоремі Байєса та припускає незалежність між ознаками. У контексті аналізу сентименту, кожне слово в тексті розглядається як незалежне, імовірності яких використовуються для визначення загальної ймовірності належності тексту до певного класу (позитивного чи негативного) [5].

Переваги:

1. Простота реалізації та швидкість навчання.
2. Ефективність при роботі з великими обсягами даних.

Недоліки:

1. Припущення про незалежність ознак часто не відповідає реальності, що може знижувати точність.
2. Менш ефективний при обробці складних мовних конструкцій.

Логістична регресія - це лінійний класифікатор, який прогнозує ймовірність належності об'єкта до одного з двох класів. Вона застосовує логістичну функцію для моделювання залежності між вхідними ознаками та ймовірністю позитивного результату.

Переваги:

1. Добре працює при наявності лінійної залежності між ознаками та класами.
2. Має інтерпретовані коефіцієнти, що дозволяє зрозуміти вплив кожної ознаки.

Недоліки:

1. Обмежена в моделюванні нелінійних залежностей.
2. Чутлива до мультиколінеарності між ознаками.

Метод опорних векторів (SVM) – це алгоритм, що шукає гіперплощину, яка найкраще розділяє дані різних класів, максимізуючи відстань (маржу) між найближчими точками різних класів (опорними векторами).

Переваги:

1. Ефективний при високій розмірності ознак.
2. Гнучкість завдяки використанню різних ядерних функцій для моделювання нелінійних залежностей.

Недоліки:

1. Високі обчислювальні витрати при роботі з великими наборами даних.
2. Чутливість до вибору ядра та параметрів моделі.

K-найближчих сусідів (K-NN) – алгоритм, що класифікує об'єкт на основі класів його k найближчих сусідів у просторі ознак. Відстань між об'єктами зазвичай вимірюється за допомогою метрики, такої як евклідова відстань.

Переваги:

1. Простота реалізації та відсутність необхідності в явному навчанні моделі.
2. Гнучкість у виборі функції відстані та кількості сусідів.

Недоліки:

1. Високі обчислювальні витрати при класифікації нових об'єктів, особливо при великих обсягах даних.
2. Чутливість до шуму та нерівномірності розподілу даних.

Випадковий ліс (Random Forest) - це ансамблевий метод, що створює множину дерев рішень під час навчання. Кожне дерево будується на випадковій підмножині ознак та даних, а класифікація здійснюється шляхом голосування більшості рішень окремих дерев.

Переваги:

1. Висока точність класифікації та стійкість до перенавчання.
2. Може обробляти великі обсяги даних та працювати з наборами, що мають багато ознак.

Недоліки:

1. Складність інтерпретації результатів через велику кількість дерев.
2. Може бути повільним при обробці дуже великих наборів даних.

Окрім використання зазначених алгоритмів у процесі побудови систем емоційного аналізу тексту важливу роль можуть відігравати спеціалізовані програмні бібліотеки, які забезпечують базову обробку природної мови (Natural Language Processing, NLP), створення векторних представлень тексту, навчання моделей машинного навчання та нейронних мереж. На даний момент існує декілька рішень.

Бібліотека Transformers від компанії Hugging Face є однією з найпопулярніших платформ для роботи з попередньо навченими великими мовними моделями, такими як BERT, RoBERTa, DistilBERT та GPT. Вона забезпечує простий доступ до тисяч моделей для задач класифікації тексту, генерації, перекладу та аналізу емоцій.

Переваги використання Transformers:

1. Наявність великого вибору готових моделей для різних мов і задач.
2. Підтримка як TensorFlow, так і PyTorch.
3. Можливість донавчання моделей на власних даних із мінімальними витратами часу.
4. Активна спільнота та регулярне оновлення.

Недоліки:

1. Високі вимоги до апаратних ресурсів при роботі з великими моделями.

2. Значний час навчання без використання GPU.

Завдяки своїй гнучкості бібліотека Transformers є особливо корисною для побудови сучасних моделей емоційного аналізу, орієнтованих на контекстне розуміння тексту.

NLTK (Natural Language Toolkit) є однією з перших і найвідоміших бібліотек для базової обробки текстових даних. Вона надає широкі можливості для токенізації, стемінгу, лематизації, синтаксичного аналізу речень, а також роботи з корпусами текстів.

Переваги використання NLTK:

1. Великий набір класичних інструментів для попередньої обробки тексту.
2. Можливість швидкої побудови прототипів NLP-проєктів.
3. Вбудовані готові корпуси даних для експериментів.

Недоліки:

1. Порівняно повільна обробка великих обсягів даних.
2. Недостатня оптимізація для сучасних завдань глибокого навчання.

NLTK доцільно використовувати на етапах підготовки тексту до подальшого аналізу або для створення невеликих демонстраційних проєктів.

spaCy — сучасна та високооптимізована бібліотека для швидкої обробки природної мови. Вона розроблена для продуктивного використання у реальних проєктах, що вимагають швидкості та ефективності.

Переваги використання spaCy:

1. Дуже висока швидкість обробки тексту.
2. Підтримка багатьох мов.
3. Можливість легко інтегрувати власні моделі у виробниче середовище.

4. Вбудована підтримка іменованих сутностей (Named Entity Recognition) та частин мови (POS-tagging).

Недоліки:

1. Менша кількість вбудованих корпусів і моделей порівняно з Hugging Face.
2. Більше орієнтована на класичний NLP, ніж на глибоке навчання.

Завдяки своїй продуктивності spaCy часто використовується у проєктах, де потрібна попередня обробка великих об’ємів тексту перед застосуванням складніших моделей.

TextBlob — бібліотека, що забезпечує простий API для базових задач обробки природної мови, зокрема для виконання полярного (sentiment) аналізу тексту.

Переваги використання TextBlob:

1. Дуже простий синтаксис для реалізації задач емоційного аналізу.
2. Вбудовані засоби для визначення тональності тексту (позитивна, негативна, нейтральна).
3. Підходить для невеликих задач або навчальних проєктів.

Недоліки:

1. Обмежена точність і гнучкість при роботі зі складними або специфічними текстами.
2. Підходить лише для базового аналізу без врахування глибокого контексту.

TextBlob може бути використаний для швидкої базової оцінки емоційного тону тексту або як референсний підхід при порівнянні з більш складними моделями.

Таким чином, сучасні бібліотеки для обробки природної мови забезпечують широкий спектр інструментів для реалізації емоційного аналізу тексту.

Вибір конкретної бібліотеки чи алгоритму залежить від складності задачі, вимог до точності, швидкості обробки та доступних обчислювальних ресурсів. У рамках даної роботи основний акцент зроблено на власній побудові моделі на основі нейронних мереж із використанням бібліотеки TensorFlow, проте знання можливостей інших платформ є важливим для розробки комплексних систем у майбутньому.

1.3. Місце емоційного аналізу на підприємствах

Емоційний аналіз тексту, як інструмент аналізу даних, здатний відігравати стратегічно важливу роль у діяльності сучасних підприємств, оскільки дозволяє виявляти неочевидні шаблони у взаємодії з клієнтами, працівниками та ринком загалом. Його застосування дає змогу підприємствам вийти за межі суто кількісного аналізу (наприклад, кількість звернень, середня оцінка задоволеності) і перейти до якісного розуміння того, що саме відчувають споживачі або співробітники.

У сфері обслуговування клієнтів емоційний аналіз може автоматично обробляти великі обсяги звернень у чатах, електронній пошті, соціальних мережах та відгуках, визначаючи негативно або позитивно забарвлені повідомлення. Це дає змогу оперативно виявляти критичні випадки (наприклад, роздратування, злість, розчарування клієнта) ще до того, як ситуація набуде резонансу, та вжити заходів щодо покращення досвіду клієнта.

У маркетинговій діяльності підприємства можуть використовувати емоційний аналіз для вивчення реакцій споживачів на нові продукти, рекламні кампанії або бренд загалом. Наприклад, аналізуючи коментарі у соціальних мережах, можна не лише визначити загальний тон, але й зрозуміти, які саме емоції домінують (захоплення, байдужість, гнів тощо). Це дозволяє маркетологам адаптувати контент під цільову аудиторію, обираючи найбільш емоційно релевантну комунікацію.

У відділах управління персоналом (HR) емоційний аналіз текстів анкет зворотного зв’язку, відкритих відповідей в опитуваннях задоволеності роботою або навіть внутрішньої корпоративної комунікації може виявляти приховане емоційне вигорання, стрес чи напруженість у колективі. Це особливо актуально для великих організацій, де традиційний HR-моніторинг не завжди встигає за динамікою змін настроїв персоналу.

У стратегічному плануванні компанія може застосовувати емоційний аналіз як частину аналітики зворотного зв’язку з ринку. Наприклад, аналізуючи публікації в медіа або відгуки про конкурентів, підприємство може побудувати емоційні карти сприйняття різних гравців на ринку та скоригувати свою стратегію просування або позиціонування.

Як приклад, дослідження впливу соціальних мереж на біотехнологічні компанії показало, що аналіз твітів та інших онлайн-обговорень може допомогти передбачити зміни в ціні акцій та приймати обґрунтовані інвестиційні рішення [8]. Або ж великі фінансові компанії застосовують емоційний аналіз для оцінки настроїв клієнтів під час взаємодії зі службою підтримки. Це дозволяє виявляти потенційних незадоволених клієнтів та проактивно вирішувати їхні проблеми, підвищуючи рівень задоволеності та лояльності [9].

Якщо говорити про базу проходження науково-дослідної практики, то дане підприємство, яке працює в сфері телекомунікацій, часто проводить опитування користувачів задовленістю різних сервісів – від якості інтернет-з’єднання до стабільності роботи мобільного застосунку. Проте зібрані дані використовуються лише для числової оцінки задоволеності користувачів, а такі речі, як побажання чи скарги або ж, наприклад, вказівки користувачів на дивний принцип взаємодії з послугою дуже часто не враховуються через відсутність можливості автоматичної обробки таких даних, в той час як ручний процес може забрати багато часу та людських ресурсів. Саме тому впровадження технології емоційного аналізу тексту, якій для роботи необхідні лише дані може принести значну користь базі практики.

Таким чином, емоційний аналіз тексту є універсальним інструментом, який дозволяє підприємствам не лише реагувати на вже наявні проблеми, а й проактивно формувати позитивний досвід взаємодії з усіма учасниками бізнес-процесів. Його інтеграція у корпоративні системи аналітики може забезпечити підприємству відчутну конкурентну перевагу та сприяти довгостроковому розвитку.

1.4. Вибір та наповнення методології проекту емоційного аналізу

У сфері науки про дані надзвичайно важливою є чітка організація етапів роботи над проєктом — від постановки задачі до впровадження готового рішення. Для цього було розроблено низку методологічних підходів, які регламентують процес аналітики даних: CRISP-DM, SEMMA, KDD, OSEMN, TDSP, ASUM-DM та DataOps. Кожен із них має свою структуру, переваги та сферу застосування.

1.4.1. CRISP-DM

CRISP-DM — одна з найпопулярніших методологій, запропонована у 1996 році консорціумом IBM, NCR, Daimler-Benz та ін. для проєктів з інтелектуального аналізу даних.

Етапи CRISP-DM:

1. Розуміння бізнесу — розуміння бізнес-проблеми та постановка задачі.
2. Розуміння даних — знайомство з даними, первинний аналіз, виявлення проблем.
3. Підготовка даних — очищення, трансформація, об’єднання даних.
4. Моделювання — вибір і побудова моделей машинного навчання.
5. Оцінка — оцінювання якості моделей, перевірка відповідності бізнес-завданню.
6. Розгортання — впровадження моделі у практичне середовище.

Переваги:

— Загальновизнаний стандарт.

— Гнучкість і циклічність (можна повертатись до попередніх етапів).

— Добре підходить для прикладного машинного навчання.

Недоліки:

— Не передбачає етапу управління експлуатацією моделі (MLOps);

— Вимагає ручного контролю на кожному етапі.

CRISP-DM є методологічною основою даної роботи, оскільки вона найкраще структурує весь цикл побудови моделі емоційного аналізу — від постановки задачі до оцінки та інтеграції.

1.4.2. SEMMA

Методологія, запропонована компанією SAS Institute як частина платформи SAS Enterprise Miner.

Етапи SEMMA:

1. Семпл — вибірка релевантних даних.
2. Дослідження — аналіз структури та розподілу.
3. Модифікація — трансформація, очищення, інженерія ознак.
4. Моделювання — побудова та тренування моделей.
5. Оцінка — перевірка якості, тестування моделей.

Переваги:

— Орієнтована саме на роботу з даними.

— Добре поєднується з візуальними інструментами SAS.

Недоліки:

— Мало уваги до бізнес-контексту.

— Обмежена інтеграція з сучасними open-source інструментами.

1.4.3. KDD (Knowledge Discovery in Databases)

Одна з найперших методологій добування знань із баз даних, представлена у 1990-х.

Етапи KDD:

1. Вибір — вибір даних.
2. Попередня обробка — очищення.
3. Трансформація — уніфікація та нормалізація.
4. Добування даних — власне побудова моделей.
5. Інтерпретація/Оцінка — аналіз результатів і формування знань.

Переваги:

— Теоретична основа сучасних підходів.

— Чітко розділяє підготовку та моделювання.

Недоліки:

— Застаріла, не охоплює сучасні потреби життєвого циклу моделей.

— Не враховує автоматизацію та впровадження.

1.4.4. OSEMN (Obtain – Scrub – Explore – Model – Interpret)

Методологія, популяризована компанією DataScience Handbook, більше орієнтована на практичних спеціалістів.

Етапи:

1. Здобуття — отримання даних із різних джерел.
2. Очищення — очищення та перетворення.

3. Дослідження — візуалізація та статистичний аналіз.
4. Моделювання — машинне навчання.
5. Інтерпретація — пояснення результатів.

Переваги:

— Простота та гнучкість.

— Добре підходить для невеликих команд і стартапів.

Недоліки:

— Відсутній формалізований підхід до розгортання та обслуговування.

— Не приділяє уваги бізнес-контексту.

1.4.5. TDSP (Team Data Science Process)

Методологія від Microsoft, спрямована на командну роботу над проектами науки про дані.

Етапи:

1. Розуміння бізнесу
2. Отримання даних та їх розуміння
3. Моделювання
4. Розгортання
5. Прийняття клієнтом

Переваги:

— Орієнтація на розгортання моделей у хмарному середовищі.

— Вбудовані шаблони для Azure, Git, CI/CD.

Недоліки:

— Прив’язка до екосистеми Microsoft.

— Менш гнучка для open-source проєктів.

1.4.6. ASUM-DM

Методологія від IBM як розширення CRISP-DM з урахуванням Big Data та сучасних технологій.

Особливості:

— Підтримує Agile.

— Орієнтована на автоматизацію.

— Включає оцінку ризиків і KPI на кожному етапі.

Переваги:

— Готова до великомасштабних проєктів.

— Формалізує узгодження з бізнес-замовниками.

Недоліки:

— Складна для впровадження в малих проєктах;

— Комерційна специфікація.

1.4.7. DataOps

DataOps — це не методологія аналізу, а підхід до організації роботи з даними, аналог DevOps у розробці.

Компоненти:

— Безперервна інтеграція та доставка (CI/CD).

— Контроль версій даних і моделей.

— Автоматизація перевірок, тестування та оновлення.

Переваги:

— Підвищує якість і стабільність моделей.

— Полегшує підтримку моделей у процесі використання.

Недоліки:

— Потребує технічної інфраструктури (pipelines, orchestration).

— Не описує саму логіку побудови моделей.

У межах даної роботи обрано методологію CRISP-DM як найбільш адаптовану до задачі емоційного аналізу текстів. Вона дозволяє послідовно пройти всі етапи — від формування цілей до тестування моделі — та залишає простір для ітерацій і вдосконалення. Тому згідно з даною методологією було визначено такі завдання на кожен етап:

1. Розуміння бізнес-проблеми: визначення цілей емоційного аналізу, а саме визначення емоційного забарвлення відповідей користувачів. Аналіз потреб та очікувань кінцевого користувача розробленої технології.
2. Розуміння даних: пошук та збір доступних даних, а саме маркованих текстових повідомлень.
3. Підготовка даних: приведення текстових повідомлень до числових представлень однакової довжини.
4. Моделювання: порівняння, вибір та побудова моделі глибокого навчання навчання. Впровадження поліпшень до обраної моделі.
5. Оцінка: оцінювання якості розробленої моделі за допомогою відповідних метрих, перевірка відповідності технології бізнес-завданню.
6. Розгортання: розробока алгоритму розгортання розробленої технології та створення інструкції користувача.

1.5. Необхідні дані

Для проведення емоційного аналізу необхідні дані, що містять висловлювання людей, у яких відображені їхні емоції, настрій або ставлення до певного об'єкта, події чи ситуації. Дана інформація може бути представлена у

різних формах: текст, аудіозапис, відео. В залежності від формату даних можуть змінюватись і моделі. Проте, незалежно від формату, ці дані мають бути достатньо об’ємними та репрезентативними, а також бажано структурованими (тобто містити як сам текст (аудіозапис, відео), так і відповідну емоційну мітку — вручну або автоматично визначену). Оскільки найпопулярнішим форматом передачі та збереження інформації є текст, то в подальшому буде розроблена технологія емоційного аналізу тексту через наявність найбільшої кількості джерел даних саме в форматі тексту.

Основні типи даних, що використовуються:

1. Текст повідомлень — це можуть бути як короткі фрази (наприклад, твіти), так і довгі тексти (відгуки, коментарі, електронні листи).
2. Мітки емоцій — вручну або автоматично позначені емоційні категорії (наприклад: радість, гнів, страх, сум, здивування, любов).
3. Метадані (не обов’язково) — дата публікації, автор, джерело тексту, мова тощо. Вони можуть бути корисними для контекстуалізації емоцій або фільтрації.

Джерела отримання даних:

1. Соціальні мережі (Twitter, Facebook, Reddit) — найпопулярніше джерело для аналізу публічних емоцій з огляду на великий обсяг і неформальний стиль спілкування.
2. Платформи з відгуками (IMDb, TripAdvisor, Amazon, Google Reviews) — дозволяють збирати тексти разом з оцінками, які можна використовувати для інтерпретації емоцій.
3. Чати та звернення до служби підтримки — внутрішні дані компаній, які дають змогу проаналізувати досвід клієнтів у реальному часі.

4. Опитування відкритого типу — анкети, де користувачі можуть вільно висловлювати свої враження, часто застосовуються в HR або соціологічних дослідженнях.
5. Форуми та блоги — джерело тематичних і часто глибших емоційних висловлювань щодо певних проблем або інтересів.
6. Спеціалізовані датасети з відкритим доступом — наприклад:
    I) Emotion Dataset (GoEmotions від Google)
    II) ISEAR (International Survey on Emotion Antecedents and Reactions)
    III) SemEval datasets (конкурсні набори для оцінки NLP-моделей)
    IV) Emotion-Stimulus dataset, DailyDialog, EmotionLines, TweetEval тощо.

Наявність якісних та релевантних джерел є критично важливою умовою для успішного навчання та оцінки моделей емоційного аналізу, особливо якщо планується використання глибокого навчання чи трансформерів, які вимагають великих обсягів даних.

1.6. Постановка задачі

Метою даної роботи є розробка технології емоційного аналізу тексту з використанням методів науки про дані та алгоритмів машинного навчання, орієнтованої на виявлення шести базових емоцій (радість, сум, злість, страх, любов, здивування). Для досягнення цієї мети необхідно вирішити низку задач, серед яких: зібрати та попередньо обробити текстові дані, здійснити анотацію або вибір датасету зі вже розміченими емоційними мітками, провести аналіз сучасних методів машинного та глибокого навчання, реалізувати модель емоційного аналізу на основі LSTM-мережі, здійснити навчання та тестування моделі, а також порівняти її ефективність із класичними алгоритмами машинного навчання. Очікується, що розроблена система зможе ефективно класифікувати емоційний стан тексту та може бути інтегрована у прикладні аналітичні рішення для бізнесу чи соціальних досліджень.

Висновки

У першому розділі було розглянуто еволюцію методів емоційного аналізу, починаючи від лексиконно-орієнтованих підходів до сучасних моделей глибокого навчання. Було проаналізовано як класичні алгоритми машинного навчання, так і нейронні архітектури (LSTM, BiLSTM, GRU, CNN, BERT, RoBERTa), які суттєво підвищили точність і гнучкість аналізу текстів, особливо в умовах природної мови.

Також розглянуто програмні бібліотеки, що забезпечують технічну реалізацію емоційного аналізу — від базових інструментів (NLTK, TextBlob) до потужних платформ для роботи з трансформерами (Transformers від HuggingFace). Проаналізовано роль емоційного аналізу у сучасному бізнес-середовищі, зокрема в таких сферах, як клієнтський сервіс, HR-аналітика, стратегічний маркетинг і соціальні дослідження.

Здійснено порівняння ключових методологій реалізації аналітичних проєктів у сфері Data Science (CRISP-DM, SEMMA, KDD, OSEMN, TDSP, ASUM-DM, DataOps), серед яких у межах цієї роботи було обґрунтовано вибір CRISP-DM як найбільш адаптованої до структури задачі емоційного аналізу.

Таким чином, даний розділ сформував ґрунтовну теоретичну основу для подальшої реалізації практичної частини, визначивши сучасні технології, інструменти та методологічні підходи, які лягли в основу розробки власної моделі емоційного аналізу.

РОЗДІЛ 2. ФОРМАЛІЗАЦІЯ МЕТОДУ ЕМОЦІЙНОГО АНАЛІЗУ

2.1. Порівняння методів емоційного аналізу

У процесі розробки ефективної системи емоційного аналізу тексту особливо важливим етапом є оцінка якості різних підходів до обробки та класифікації текстових даних. З метою виявлення найбільш продуктивного методу, необхідно провести порівняльне тестування алгоритмів, які представляють два основні підходи в машинному навчанні: класичні моделі та моделі, що ґрунтуються на нейронних мережах. До першої групи увійдуть алгоритми, які традиційно використовуються для обробки текстових задач, зокрема метод опорних векторів (SVM), дерева рішень, наївний баєсівський класифікатор та логістична регресія. До другої групи належать моделі глибокого навчання, зокрема рекурентна нейронна мережа на основі LSTM, які продемонстрували високу ефективність в обробці послідовностей та врахуванні контексту та покращена двонаправлена LSTM. Результати тестування класичних методів машинного навчання наведені в таблиці 2.1.

Таблиця 2.1 – Загальний результат оцінки класичних методів

Метод	Точність	Час тренування
Наївний Байєс	83.60%	0.39 секунди
Опорних векторів	84.10%	66.06 секунд
Дерево рішень	76.15%	15.24 секунди
Логістична регресія	84.65%	0.86 секунди

Тепер оглянемо результати методів глибокого навчання.

Таблиця 2.2 – Загальний результат оціни методів глибокого навчання

Метод	Точність	Час тренування
LSTM	91.20%	2 хв. 12 сек.
BiLSTM	89%	9 хв. 13 сек.
BERT	93.20%	3 год. 42 хв.
CNN	91.85%	26 сек.
GRU	92.10%	2 хв. 34 сек.

Виходячи із отриманих результатів можна зробити висновок, що для подальшої розробки технології емоційного аналізу в контексті бази практики слід обрати модель LSTM. Переваги LSTM порівняно з класичними алгоритмами (SVM, Naive Bayes, Decision Tree тощо):

1. Урахування послідовності та контексту: класичні моделі оперують фіксованими векторами ознак (наприклад, TF-IDF), ігноруючи порядок слів. Натомість LSTM, як рекурентна нейронна мережа, «читає» текст як послідовність слів, зберігаючи інформацію про попередні слова, що дозволяє враховувати контекст і смислові зв'язки.
2. Краще розпізнавання складних емоційних патернів: емоції в тексті можуть бути прихованими або залежними від довгих фраз. LSTM здатна «запам’ятовувати» важливі частини тексту навіть через кілька речень, чого не здатні зробити прості лінійні або деревоподібні моделі.
3. Автоматичне виділення ознак**:** LSTM працює з векторизованим представленням слів (наприклад, word2vec або GloVe) та сама навчається виявляти важливі патерни, тоді як класичні методи потребують ручної інженерії ознак або спрощених представлень тексту.

Переваги LSTM у порівнянні з BiLSTM (двонапрямленою LSTM):

1. Менша обчислювальна складність: BiLSTM складається фактично з двох LSTM, які працюють у протилежних напрямках, подвоюючи кількість параметрів і споживання ресурсів. Якщо задача не вимагає дуже точної семантичної обробки (наприклад, не переклад чи генерація), класична LSTM може бути достатньо точною.
2. Швидше навчання та нижчі вимоги до даних: BiLSTM, як і інші складні архітектури, потребує більше прикладів для ефективного навчання. Оскільки доступна вибірка є обмеженою, LSTM з меншою кількістю параметрів покаже кращі результати через менший ризик перенавчання.

3. Простота інтеграції та модифікації: звичайну LSTM-модель легше реалізувати, змінити чи інтегрувати в інші системи.

Переваги LSTM порівняно з BERT та трансформерами:

1. Обчислювальні ресурси та навчання: моделі на основі BERT мають сотні мільйонів параметрів і вимагають значного обсягу пам’яті та обчислювальної потужності (особливо GPU), чого не може собі дозволити база практики оскільки локальні серверні потужності не використовують графічні процесори та більшу частину часу зайняті виконанням запитів та оновленням баз даних підприємства. LSTM — набагато «легша» альтернатива, яка може працювати навіть на середньостатистичному ноутбуці.
2. Навчання з нуля або тонке налаштування: для BERT зазвичай потрібне fine-tuning великої попередньо навченої моделі, що потребує більше технічного досвіду. LSTM можна навчити з нуля на власному наборі даних без необхідності завантаження і налаштування складних моделей.
3. Краще працює на менших датасетах: у випадках, коли даних небагато або їх складно анотувати (що типово для емоційного аналізу з багатьма класами), LSTM переважно показує стабільніші результати, ніж трансформери, схильні до переобучення.

Переваги LSTM порівняно з CNN:

1. Вища стійкість до втрати контексту у складних текстах: LSTM краще справляється із задачами, де важливим є збереження контексту при обробці довгих і складних текстів. CNN, яка працює з локальними вікнами слів через згортки, фактично не має можливості глобально "пам’ятати" контекст всього тексту. У той час як LSTM поступово оновлює свій внутрішній стан при обробці кожного слова, що дозволяє їй утримувати цілісне розуміння тексту.

2. Можливість обробляти тексти різної довжини без втрати якості: LSTM адаптивно працює з текстами різної довжини завдяки своїй рекурентній природі, тоді як CNN потребує чіткої фіксації розміру вхідних послідовностей для ефективної роботи. Це робить LSTM універсальним вибором для задач, де довжина текстів варіюється від кількох слів до кількох абзаців.
3. Більш гнучке моделювання залежностей між словами: через наявність окремих механізмів для "забування" старої інформації і "запам’ятовування" нової, LSTM має більшу гнучкість у визначенні, які саме елементи тексту є релевантними для прийняття рішення. У CNN вікно згортки має фіксований розмір, що обмежує здатність моделі вловлювати залежності між віддаленими словами.

Переваги LSTM порівняно з CNN:

1. Краще збереження довготривалих залежностей у тексті: на відміну від GRU, LSTM має окрему комірку пам'яті та три типи воріт, що дозволяють моделі ефективно запам’ятовувати важливу інформацію протягом тривалих часових проміжків. Завдяки цьому LSTM здатна враховувати емоційні нюанси, які виявляються лише через декілька речень або після складних синтаксичних конструкцій. GRU, через більш спрощену структуру, іноді втрачає таку здатність при обробці довгих послідовностей.

Вибір LSTM є збалансованим рішенням: вона значно потужніша за класичні моделі в задачах, де важливо враховувати послідовність і контекст, але при цьому легша, швидша та менш вимоглива, ніж BiLSTM чи BERT. Вона є ідеальним вибором для проєктів з обмеженими ресурсами, невеликим або середнім обсягом даних, або коли потрібна хороша якість при простій реалізації. Саме тому LSTM часто використовують у прикладних дослідженнях, дипломних роботах і реальних продуктах, де важливий баланс між ефективністю й практичністю.

2.2. Алгоритм роботи LSTM

Для кращого розуміння проблем, які має LSTM потрібно розуміти як працює даний метод глибокого навчання. Варто почати з огляду рекурентних нейронних мереж. Рекурентні нейронні мережі (RNN) призначені для обробки послідовних даних, таких як текст, аудіо або часові ряди. Основна ідея полягає в тому, що мережа має внутрішній стан (прихований стан), який оновлюється на кожному кроці послідовності, дозволяючи враховувати попередню інформацію при обробці поточного елемента. Однак традиційні RNN стикаються з проблемою затухаючих або вибухаючих градієнтів під час навчання, що ускладнює захоплення довгострокових залежностей у даних.

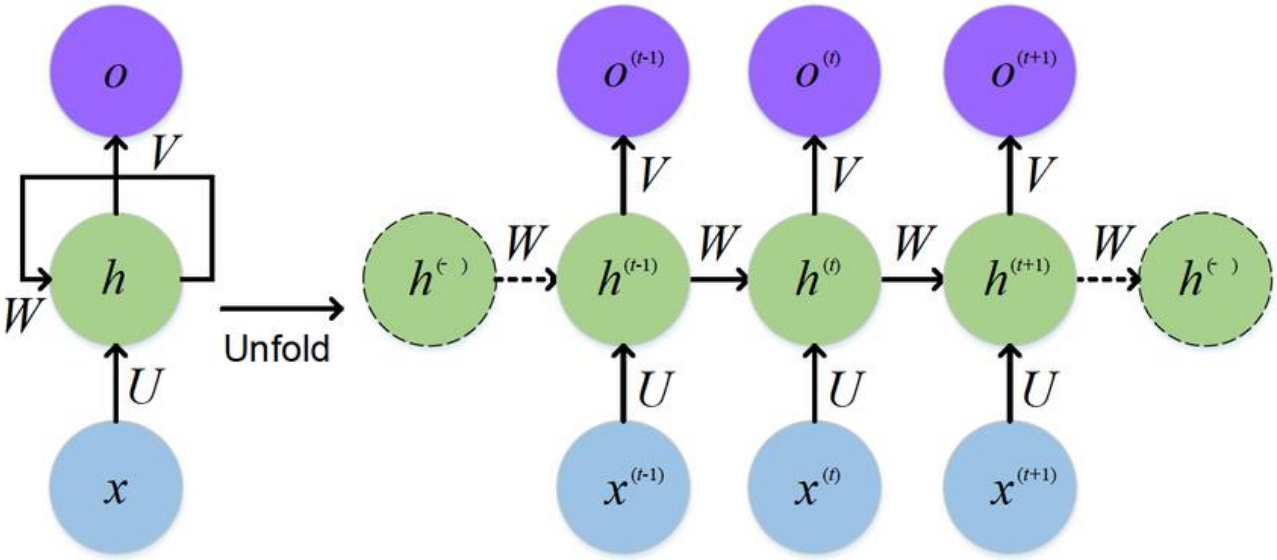


Рисунок 2.1 – Схематичне зображення роботи RNN

Що ж таке затухаючі та вибухаючі градієнти? Під час зворотного поширення помилки градієнти (похідні функції втрат) передаються від вихідного шару до вхідного. Якщо значення градієнтів менше 1, їхнє послідовне множення через шари призводить до експоненційного зменшення. Цей ефект називається затуханням градієнтів

Наслідки:

1. Перші (глибші) шари мережі навчаються дуже повільно або взагалі не навчаються.

2. Мережа не може ефективно захоплювати довгострокові залежності у даних.

Причини**:**

1. Використання активаційних функцій з похідними в межах (0, 1), наприклад, сигмоїда або гіперболічний тангенс.
2. Глибока архітектура мережі з великою кількістю шарів.

Вибухання градієнтів це протилежний процес. Якщо значення градієнтів більше 1, їхнє послідовне множення через шари призводить до експоненційного зростання.

Наслідки:

1. Нестабільне навчання мережі.
2. Переповнення числових значень, що може призвести до помилок у обчисленнях.

Причини:

1. Використання активаційних функцій з великими похідними.
2. Неправильна ініціалізація ваг мережі.

LSTM — це спеціалізований тип RNN, розроблений для подолання проблеми затухаючих градієнтів і здатний ефективно захоплювати довгострокові залежності в послідовностях. Ключовою особливістю LSTM є наявність структури з трьох "воріт".

1. Ворота забування (Forget Gate): визначають, яку інформацію з попереднього стану слід зберегти або забути.
2. Ворота введення (Input Gate): контролюють, яку нову інформацію слід додати до стану пам'яті.

3. Ворота виводу (Output Gate): визначають, яку частину стану пам'яті слід використовувати як вихід.

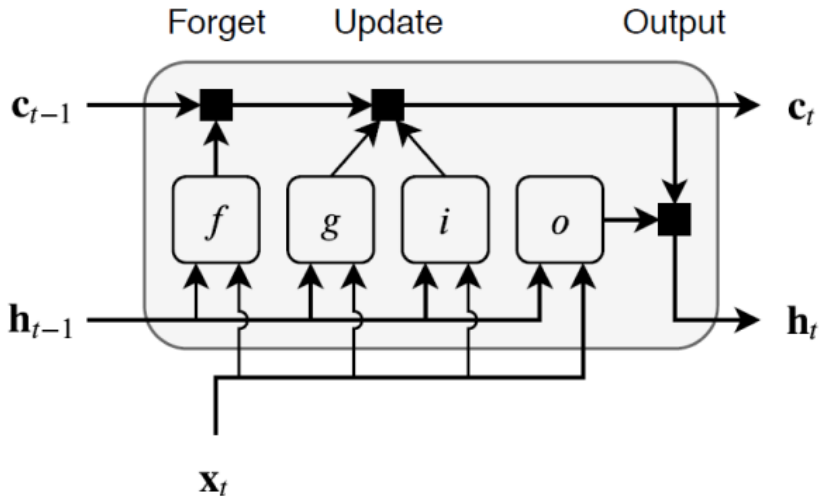


Рис. 2.2 – Схематичне зображення роботи LSTM

Ці ворота дозволяють LSTM вибірково зберігати або забувати інформацію, що робить її ефективною для задач, де важливо враховувати контекст на довгих послідовностях даних.

2.3. Пропоновані поліпшення

Розробка базової моделі на основі LSTM дозволяє досягти певного рівня точності при розв’язанні задачі емоційного аналізу тексту. Проте для підвищення її продуктивності та здатності до узагальнення необхідно здійснити ряд технічних удосконалень. Тому я розглянув ряд поліпшень, що ґрунтуються на сучасних дослідженнях та практичних рекомендаціях, які можна використати при розробці даної моделі глибокого навчання.

Перш за все, використання попередньо навчених векторних представлень слів. Одним із важливих аспектів якості моделі є те, як вона представляє текстові дані у числовій формі. Замість випадкової ініціалізації ембедінгів, доцільно використовувати попередньо навчені векторні представлення, наприклад, GloVe

(Global Vectors for Word Representation). Вектори GloVe навчені на великих корпусах тексту і здатні вловлювати семантичні подібності між словами. Імплементація таких векторів дозволяє моделі отримати глибше розуміння контексту вже на початкових етапах навчання, що може значно покращити якість класифікації емоцій, особливо при обмеженій кількості навчальних даних.

Одним із основних викликів при побудові нейронних мереж є проблема перенавчання (overfitting) — ситуації, коли модель дуже добре запам'ятовує тренувальні дані, але погано узагальнює знання на нові приклади. Для боротьби з перенавчанням застосовується низка методів регуляризації.

Регуляризація — це сукупність технік, спрямованих на обмеження складності моделі з метою покращення її здатності до узагальнення. Основна ідея регуляризації полягає в додаванні штрафу за занадто складні або великі значення ваг моделі у функцію втрат під час навчання. Давайте розглянемо основні регуляризатори, які застосовуються для задач глибокого навчання.

L1-регуляризація (Lasso Regularization). Даний регуляризатор додає до функції втрат суму абсолютних значень ваг мережі. Формально:

$$Loss_{L1} = Loss + \lambda \sum_{i} |w_i|$$

де λ — коефіцієнт регуляризації, а w_i — ваги моделі.

Особливості**:**

1. Сприяє зануленню частини ваг (виродження ваг у нуль).
2. Використовується для побудови розріджених моделей (Sparse models).
3. Може бути корисною при великій кількості неінформативних ознак.

L2-регуляризація (Ridge Regularization). L2-регуляризація додає до функції втрат суму квадратів ваг:

$$Loss_{L1} = Loss + \lambda \sum_i w_i^2$$

Особливості**:**

1. Схильна зменшувати значення ваг до малих, але не до нуля.
2. Забезпечує гладкість розподілу ваг.
3. Часто використовується у нейронних мережах для стабілізації навчання.

L2-регуляризація допомагає зменшити дисперсію моделі, зробивши її менш чутливою до випадкових флуктуацій у тренувальних даних.

ElasticNet (Комбінація L1 та L2). ElasticNet об'єднує переваги L1 та L2 регуляризацій:

$$Loss_{ElasticNet} = Loss + \lambda \sum_i |w_i| + \lambda \sum_i w_i^2$$

Особливості**:**

- Комбінує розрідженість (від L1) та гладкість ваг (від L2).

Застосовується переважно у випадках, коли потрібно балансувати між видаленням зайвих ознак та стабільністю моделі.

Існують також інші методи регуляризації, а саме:

1. Dropout: випадкове "вимикання" певного відсотка нейронів під час навчання, що запобігає перенавчанню шляхом зменшення залежності моделі від окремих нейронів.
2. Early Stopping: припинення навчання тоді, коли точність на валідаційних даних перестає покращуватись.

Для задачі емоційного аналізу серед L1, L2 та ElasticNet досить добре підходить L2-регуляризатор. Під час навчання нейронної мережі з L2-регуляризацією до стандартної функції втрат додається додатковий член, що залежить від квадрату

ваг. У процесі оптимізації модель прагне не лише мінімізувати помилку прогнозу, але й утримувати ваги невеликими.

Це сприяє:

1. Зменшенню дисперсії моделі.
2. Покращенню стійкості до шуму в даних.
3. Уповільненню процесу перенавчання.

Таким чином, застосування L2-регуляризації дозволяє побудувати більш узагальнюючу та стабільну модель без значного ускладнення архітектури.

Наступне можливе поліпшення це балансування класів та збільшення вибірки (Data Augmentation). У задачах емоційної класифікації часто спостерігається дисбаланс класів, коли деякі емоції представлені значно частіше за інші. Це може призводити до упередженості моделі. Для вирішення цієї проблеми можна використати:

1. Oversampling менш представлених класів.
2. Text augmentation — генерація нових текстів за допомогою синонімізації, перефразування або перекладу зворотнім перекладом (back translation).

Ці методи дозволяють розширити навчальний набір та покращити його репрезентативність.

Наступним кроком може бути нормалізація вхідних послідовностей. Застосування попередньої обробки тексту, зокрема лематизації, приведення тексту до нижнього регістру, видалення пунктуації та стоп-слів сприяє зменшенню шуму у даних. Крім того, нормалізація довжини послідовностей (padding або truncation) дозволяє уніфікувати вхідні дані для обробки в батчах.

Також суттєвий вплив на ефективність LSTM-моделі мають її архітектурні параметри:

1. Кількість LSTM-шарів: додавання другого шару може покращити здатність до навчання складніших залежностей, проте збільшує ризик перенавчання.
2. Розмір прихованого стану: збільшення кількості нейронів у прихованих шарах дозволяє моделі зберігати більше інформації, але також ускладнює навчання.

У задачах багатокатегоріальної класифікації, особливо з розбалансованими даними, стандартна функція втрат (categorical crossentropy) може не бути достатньо ефективною. Замість неї можна використати виважену функцію втрат або функцію втрат focal loss. У ході розробки моделі було виявлено проблему потенційної нерівномірності розподілу класів у наборі даних, що може призводити до переваги більш представлених емоційних класів над менш поширеними. Для вирішення цієї проблеми та покращення здатності моделі розпізнавати рідкісні класи було вирішено застосувати функцію втрат Focal Loss замість стандартної функції sparse_categorical_crossentropy.

Focal Loss модифікує базову функцію втрат шляхом додавання коефіцієнта, який зменшує вагу легко класифікованих прикладів і підсилює увагу моделі до складних випадків. Формально функція визначається як:

$$FL(p_t) = -\alpha(1-p_t)^{\gamma}\log(p_t)$$

де:

— p_t — ймовірність передбачення правильного класу,

— α — ваговий коефіцієнт для компенсації незбалансованості,

— γ — параметр фокусування, що посилює акцент на важких для класифікації прикладах.

Стандартними значеннями для даної функції втрат є γ=2.0 та α=0.25, які є типовими для подібних задач.

Удосконалення LSTM-моделі можливе через впровадження як архітектурних змін, так і методів покращення підготовки даних, регуляризації та оптимізації функції втрат. Реалізація запропонованих поліпшень дозволяє підвищити точність моделі, зменшити ймовірність перенавчання та покращити її здатність до генералізації на нових прикладах. На наступному етапі дослідження планується реалізувати ці підходи та порівняти їх ефективність у рамках експериментального дослідження.

2.4. Навчання з розширеною пам’яттю

Окремо варто згадати підхід навчання з рошириною пам’яттю. Навчання з розширеною пам’яттю (memory-augmented learning) — це підхід у машинному та глибокому навчанні, за якого нейронна мережа під час тренування або формування висновків отримує доступ до зовнішньої або внутрішньої пам’яті, в якій зберігається інформація про попередні приклади, їх ознаки або внутрішні стани.[13]

На відміну від звичайних моделей, які не зберігають довгострокової інформації про минулі приклади, memory-augmented моделі можуть запам’ятовувати, узагальнювати та повторно використовувати знання про структуру класів або особливості даних, що дозволяє покращити загальну ефективність, особливо в складних або незбалансованих задачах.

У традиційних нейронних мережах знання про дані "вбудовуються" у ваги під час навчання. Проте в багатьох задачах — таких як класифікація нових або рідкісних класів, розпізнавання емоцій чи few-shot learning — важливо мати механізм гнучкої адаптації до прикладів, які модель уже бачила. Навчання з розширеною пам’яттю дозволяє вирішити це завдяки наявності структури пам’яті, яка зберігає інформацію окремо від ваг мережі.

Існує кілька ключових підходів до реалізації навчання з розширеною пам’яттю, які відрізняються за архітектурою пам’яті, способом доступу до неї та її призначенням.

2.4.1. Нейронні машини Тюрінга

Нейронні машини Тюрінга використовуються для моделювання процесів, які потребують використання тимчасової пам’яті — сортування, копіювання, асоціативне відтворення. Дана архітектура містить два основні блоки:

— Контролер (нейронна мережа, наприклад, LSTM), яка приймає вхідні дані.
— Матриця пам’яті — зовнішня пам’ять розміром N×M, де N — кількість комірок, M — розмір кожної.

Контролер навчається читати з і писати в пам’ять за допомогою диференційовних операцій.

Принцип дії:

— На кожному кроці конролер подає запит і отримує відповідь з пам’яті.

— Контролер змінює вміст пам’яті на основі свого стану.

Переваги:

1. Може реалізовувати алгоритми.
2. Вчиться працювати з пам’яттю як з базою даних.

Недоліки:

1. Дуже складна для тренування.
2. Чутлива до гіперпараметрів.
3. Погано масштабується.

Зазвичай, використовується для задач копіювання, сортування, логічного виведення та алгоритмічного навчання.

2.4.2. Мережі пам’яті

Мережі пам’яті застосовуються для зберігання та використання текстових фактів, наприклад, у задачах запитань і відповідей. Архітектура такого підходу складається з:

— Модуль вводу — обробляє вхід (наприклад, запит).

— Модуль пам’яті — зберігає інформацію (наприклад, речення з тексту).

— Механізм уваги — вибирає релевантні частини пам’яті.

— Модуль виводу — формує відповідь.

Принцип дії:

— Для кожного запиту відбувається пошук релевантних фактів у пам’яті.

— Вибрані факти комбінуються та використовуються для формування відповіді.

— Може здійснювати кілька "кроків розуміння" (multi-hop attention).

Дана архітектура також має власні підвиди:

— End-to-End мережі пам’яті — навчання через зворотне поширення помилки.

— Динамічні мережі пам’яті — розширення з рекурсивним оновленням пам’яті.

Переваги:

1. Ефективні в задачах обробки природної мови.
2. Добре справляються із запитаннями, що потребують "розуміння контексту".

Недоліки:

1. Потребують великої кількості пам’яті.
2. Працюють переважно з текстом.

Типовими задачами для даного підходу є питання-відповіді, діалогові системи та логічне виведення.

2.4.3. Memory-augmented meta learning

Цей підхід використовується для навчання моделей на дуже обмеженій кількості прикладів. Архітектура комбінує:

— Meta-learner (наприклад, LSTM або CNN),

— Модуль пам’яті, який зберігає приклади, згруповані по класах.

Принцип дії:

— Під час навчання модель отримує лише кілька прикладів на клас (наприклад, 1–5).

— Кожен приклад зберігається у пам’яті як вектор ознак.

— Нові приклади порівнюються з уже збереженими.

— Класифікація відбувається на основі схожості до прикладів у пам’яті.

Переваги:

1. Працюють із малою кількістю даних.
2. Гнучкість до нових класів без перенавчання.

Недоліки:

1. Потребують ретельної підготовки пам’яті.
2. Не дуже стійкі до шумів у прикладах.

Типові задачі: класифікація з 1–5 прикладів на клас (few-shot), медичні діагнози, класифікація нових об'єктів.

2.4.4. Епізодична пам’ять

Епізодична пам’ять застосовується для покращення навчання класифікаторів за рахунок зберігання узагальнених характеристик класів.

Архітектура: для кожного класу формується окрема "пам’ять" (наприклад, словник), яка зберігає:

— Приховані стани прикладів.

— Або середні вектори класу (центроїди).

Принцип дії:

— Під час тренування для кожного прикладу обчислюється його вектор (наприклад, з виходу LSTM).

— Цей вектор:

1. Або зберігається у пам’яті класу.
2. Або порівнюється з середнім вектором цього класу.

— Якщо новий вектор віддаляється від центру, модель отримує штраф (regularization loss).

Переваги:

1. Дуже проста реалізація.
2. Легко додається до вже існуючих моделей.
3. Добре працює в багатокласових задачах.

Недоліки:

1. Не масштабується на мільйони класів.
2. Не зберігає контекст усіх прикладів, лише "усереднений сенс".

Типовими задачами є емоційний аналіз, розпізнавання облич, тональна класифікація.

Навчання з розширеною пам’яттю охоплює широке коло підходів — від складних нейронних машин до простих центрів класу. Вибір методу залежить від задачі, обсягу даних, обчислювальних ресурсів та необхідної точності. У практичних умовах, зокрема для задач емоційної класифікації тексту, найбільш доцільними виявляються простіші реалізації пам’яті, які легко інтегруються в

існуючі нейронні архітектури та забезпечують суттєве підвищення узагальнення без втрати ефективності.

2.5. Джерело отримання даних для тренування

Для навчання та тестування моделі емоційного аналізу тексту було використано відкритий набір даних "Emotion", розміщений на платформі Hugging Face під назвою dair-ai/emotion. Цей датасет широко застосовується у дослідженнях з обробки природної мови (NLP), зокрема в задачах емоційної класифікації, і надає збалансований корпус коротких текстових повідомлень англійською мовою (здебільшого — твіти), отримані з соціальної мережі X, які вручну анотовано відповідно до певних емоційних категорій.

Загальний обсяг датасету становить 20 000 записів, які поділено на три підмножини: навчальну — 16 000 зразків, валідаційну — 2 000 зразків, та тестову — 2 000 зразків. Кожен запис містить два основні поля: text — текст повідомлення, та label — одна з шістьох емоційних категорій: joy (радість), sadness (смуток), anger (злість), fear (страх), love (любов) та surprise (здивування).

Цей набір даних є ідеальним для дослідження завдань емоційної класифікації завдяки таким характеристикам:

1. Збалансованість класів — набір містить порівняно рівномірну кількість прикладів для кожної емоції, що дозволяє уникнути упередженості моделі.
2. Реальні приклади текстів — зразки походять з соціальної мережі X, що робить модель придатною для практичного використання;
3. Стандартизована структура — формат набору даних добре підтримується більшістю сучасних бібліотек, таких як datasets від Hugging Face, pandas, PyTorch та TensorFlow.

Обрана структура і якість датасету дозволяє забезпечити надійну базу для навчання та подальшого тестування моделей машинного та глибокого навчання в межах даної дипломної роботи.

2.6. Метрики для оцінки ефективності моделі

Оцінювання ефективності моделей машинного та глибокого навчання є критично важливою складовою розробки інтелектуальних систем. Вибір метрик впливає як на процес побудови моделі, так і на її практичну цінність. Залежно від поставленої задачі — багатокласової класифікації, бінарної класифікації чи регресії — застосовуються відповідні метрики. У задачах емоційного аналізу, які зазвичай є задачами класифікації, особливу роль відіграють точність (accuracy), повнота (recall), точність (precision), F-міра, AUROC та інші.

Точність (Accuracy) — це частка правильно класифікованих прикладів серед усіх прикладів:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

де:

— TP — кількість істинно позитивних передбачень,

— TN — істинно негативні,

— FP — хибно позитивні,

— FN — хибно негативні.

Переваги:

— Простота інтерпретації.

— Добре працює на збалансованих даних.

Недоліки**:**

— Малоефективна для незбалансованих класів (може бути штучно високою при домінуванні одного класу).

Точність може бути достатньою в задачах, де всі класи мають приблизно однакову кількість прикладів.

Точність (Precision) — це частка правильних позитивних передбачень серед усіх передбачених як позитивні певного класу:

$$Precision = \frac{TP}{TP + FP}$$

Переваги:

— Корисна при великій кількості хибнопозитивних передбачень.

Недоліки:

— Не враховує хибнонегативні передбачення (FN), тому в задачах з неповною вибіркою може бути упередженою.

Має корисне застосування у задачах виявлення спаму, де важливо не помилково маркувати "нормальні" повідомлення як спам.

Повнота (Recall) — це частка правильно передбачених позитивних прикладів серед усіх фактичних позитивних певного класу:

$$Recall = \frac{TP}{TP + FN}$$

Переваги:

— Корисна, коли важливо не пропустити позитивні класи.

Недоліки:

— Може бути високою при низькій точності.

Часто застосовується в медичних діагностичних системах, де важливо виявити всі випадки хвороби навіть ціною помилкових класфікацій.

F1-міра — це гармонійне середнє між точністю та повнотою:

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

Переваги:

— Балансує дві метрики.

— Стійка до незбалансованих класів.

Недоліки:

— Не дозволяє віддати перевагу точності або повноті (використовується F2, якщо повнота важливіша).

Використовується в задачах класифікації з незбалансованими класами, зокрема у емоційному аналізі.

Fβ-міра — узагальнення F1, що дозволяє контролювати важливість точності чи повноти:

$$F_\beta = (1 + \beta^2) * \frac{Precision * Recall}{(\beta^2 * Precision) + Recall}$$

— Якщо β > 1 — наголос на повноті;

— Якщо β < 1 — наголос на точності.

Fβ застосовується там, де важливіше виявити всі релевантні приклади, навіть якщо частина буде помилкова.

AUROC (Area Under the ROC Curve) — використовується для оцінювання якості класифікаційної моделі, особливо у випадках бінарної класифікації. Вона поєднує в собі ROC-криву (Receiver Operating Characteristic curve) — графік залежності True Positive Rate (TPR) від False Positive Rate (FPR) — та площу під цією кривою, яка й становить значення AUROC.

$$AUROC = \int_0^1 TPR(FPR)dFPR$$

Визначення компонентів:

— True Positive Rate (TPR) або Recall:

$$TPR = \frac{TP}{TP + FN}$$

— False Positive Rate (FPR):

$$FPR = \frac{FP}{FP + TN}$$

Далі ROC-крива будується шляхом поступової зміни порога, за яким модель вирішує, до якого класу належить приклад. Для кожного значення порога обчислюються значення True Positive Rate (TPR) і False Positive Rate (FPR). Ці пари значень наносяться на графік, де по осі X відкладається FPR, а по осі Y — TPR.

Інтерпретація AUROC:

— AUROC = 1.0: ідеальна модель.

— AUROC = 0.5: модель не відрізняється від випадкового вгадування.

— AUROC < 0.5: модель працює гірше за випадкову (переплутані мітки).

Переваги:

— Дозволяє оцінити модель без жорсткої прив’язки до одного значення.

— Стійка до незбалансованих класів, на відміну від точності.

— Добре демонструє здатність моделі відокремлювати позитивний клас від негативного.

Недоліки:

— Менш інтерпретативна у багатокласових задачах — потребує розширення (наприклад, "one-vs-rest").

— Не завжди співвідноситься з практичною якістю передбачень, якщо конкретний поріг важливий (наприклад, у медичних діагнозах).

Застосовується коли наявний сильний дисбаланс класів.

Матриця неточностей — це табличне представлення реальних і передбачених значень. Для багатокласової класифікації вона дозволяє побачити:

— які класи найчастіше плутаються,

— які класи найкраще визначаються.

Переваги:

— Дає повну картину помилок.

— Універсальна для багатокласових задач.

Matthews Correlation Coefficient (MCC) вважається однією з найнадійніших метрик як для збалансованих, так і для незбалансованих задач.

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Часто застосовується в молекулярній біології та задачах з дуже нерівномірним розподілом класів.

Правильний вибір метрики — критичний для оцінювання якості моделі. У задачах емоційного аналізу, де часто зустрічається незбалансованість даних, точність (accuracy) сама по собі може бути оманливою. Тому для повноцінної оцінки необхідно використовувати комбінацію метрик, зокрема F1-міру, F2, матрицю неточностей та MCC, а також, у разі потреби, AUROC.

2.7. Інформаційні засоби для розробки

У процесі розробки моделі емоційного аналізу тексту на основі архітектури LSTM було використано низку сучасних інформаційних засобів, що забезпечують зручність, гнучкість та ефективність при роботі з даними і побудові моделей машинного навчання. Основною мовою програмування є Python, яка завдяки своїй багатій екосистемі бібліотек є однією з найпопулярніших у галузі обробки природної мови (NLP) та глибокого навчання.

Python — це високорівнева мова програмування загального призначення з відкритим вихідним кодом, яка підтримує різноманітні парадигми програмування, включаючи об'єктно-орієнтоване, процедурне та функціональне. Завдяки своїй простоті синтаксису та потужним бібліотекам Python став основним інструментом для реалізації задач у сфері науки про дані та штучного інтелекту. У рамках даної роботи Python використовується для передобробки даних, побудови моделі LSTM, навчання моделі та аналізу результатів.

pandas — це бібліотека для роботи з табличними даними, що надає високорівневі структури, зокрема DataFrame, для зручного зберігання, маніпуляції та аналізу даних. У межах реалізації LSTM pandas застосовується для завантаження датасету, попередньої обробки тексту, фільтрації, групування та підготовки даних до подачі у модель. Бібліотека дозволяє ефективно обробляти великі обсяги інформації з мінімальними затратами коду.

NumPy (Numerical Python) — це бібліотека для роботи з багатовимірними масивами (array) та виконання складних математичних операцій над ними. У даному проєкті NumPy використовується для виконання різних математичних операцій у процесі підготовки даних та обчислення метрик. Вона забезпечує ефективну взаємодію з іншими бібліотеками, зокрема TensorFlow та scikit-learn.

TensorFlow — це одна з найпотужніших платформ для створення та навчання моделей машинного і глибокого навчання, розроблена компанією Google. Вона підтримує як високорівневий API (через Keras), так і низькорівневий доступ до операцій з тензорами. У даній роботі TensorFlow використовується для побудови та навчання моделі LSTM, визначення архітектури мережі, функції втрат, оптимізатора, регуляризацій, а також для моніторингу результатів тренування. TensorFlow забезпечує гнучкість і масштабованість, дозволяючи працювати як на CPU, так і на GPU.

Scikit-learn — це одна з найпопулярніших бібліотек Python для задач машинного навчання. Вона є відкритим програмним продуктом і базується на бібліотеках NumPy, SciPy та matplotlib, що робить її легко інтегровуваною з іншими інструментами аналітики даних. Scikit-learn призначена для реалізації класичних алгоритмів машинного навчання, які не пов'язані з глибоким навчанням (deep learning), але залишаються ефективними для широкого кола задач: від класифікації й кластеризації до регресії, зменшення розмірності та обробки даних.

Matplotlib — це базова бібліотека для візуалізації даних у середовищі Python, яка дозволяє створювати графіки різної складності: від простих лінійних діаграм до багатовимірних графіків з високим рівнем кастомізації. Matplotlib призначена для гнучкої побудови графіків будь-якої складності: аналітичних, наукових, презентаційних тощо. Вона забезпечує інтерактивність, інтегрується з Jupyter Notebook та підтримує експорт у різні формати (PNG, PDF, SVG, EPS).

Seaborn — це високорівнева бібліотека візуалізації для мови програмування Python, яка базується на основі бібліотеки matplotlib. Вона створена для спрощення побудови статистичних графіків та є надзвичайно корисною при дослідженні та аналізі структури даних у проектах Data Science та машинного навчання. Метою Seaborn є створення естетично привабливих, інформативних та легко інтерпретованих графіків, які дають змогу швидко виявляти закономірності, кореляції, аномалії та розподіли в даних.

Jupyter Notebook — це інтерактивне середовище з відкритим вихідним кодом, призначене для створення, редагування та виконання коду з можливістю поєднання тексту, формул, коду та візуалізацій в одному документі.

Висновки

В даному розділі було детально розглянуто принципи побудови моделей емоційного аналізу тексту на основі сучасних методів машинного та глибокого

навчання. Проведено порівняльний аналіз різних нейронних архітектур, таких як LSTM, GRU та CNN, визначено їхні особливості, переваги й доцільність використання в задачах класифікації емоцій. Особливу увагу приділено моделі LSTM як такій, що найкраще адаптована до обробки послідовностей природної мови завдяки збереженню контекстної інформації.

У межах розділу також було описано обґрунтування використання метрики оцінки якості класифікації, зокрема точності, повноти, F1-міри, F2-міри, AUROC, MCC та матриці неточностей. Проведено аналіз сильних і слабких сторін кожної метрики, що дозволило забезпечити комплексне й об’єктивне оцінювання результатів побудованої моделі.

Окрему увагу було приділено методології навчання з розширеною пам’яттю, яка дає змогу зберігати узагальнені представлення класів у вигляді векторів пам’яті та використовувати їх як додаткову інформацію під час тренування моделі. Описано як класична реалізація цієї концепції, так і її адаптована версія, реалізована в рамках цієї роботи.

Таким чином, другий розділ заклав практичну основу для побудови ефективної моделі емоційного аналізу, визначивши не лише архітектуру і алгоритми навчання, а й критерії оцінювання її якості. Усе це забезпечує підготовку до реалізації експериментальної частини дослідження, що буде висвітлено в наступному розділі.

РОЗДІЛ 3. РОЗРОБКА МЕТОДУ ЕМОЦІЙНОГО АНАЛІЗУ

3.1. Аналіз вхідних даних

Перед побудовою та навчанням моделі машинного навчання надзвичайно важливо провести детальний аналіз вхідних даних. Якість, структура та збалансованість вхідної інформації безпосередньо впливають на здатність моделі адекватно навчатися та узагальнювати закономірності.

3.1.1. Оверсемплінг

Одним з критичних аспектів є перевірка рівномірності розподілу класів у тренувальній вибірці, оскільки дисбаланс між категоріями може призвести до упередженості моделі у бік переважаючих класів.

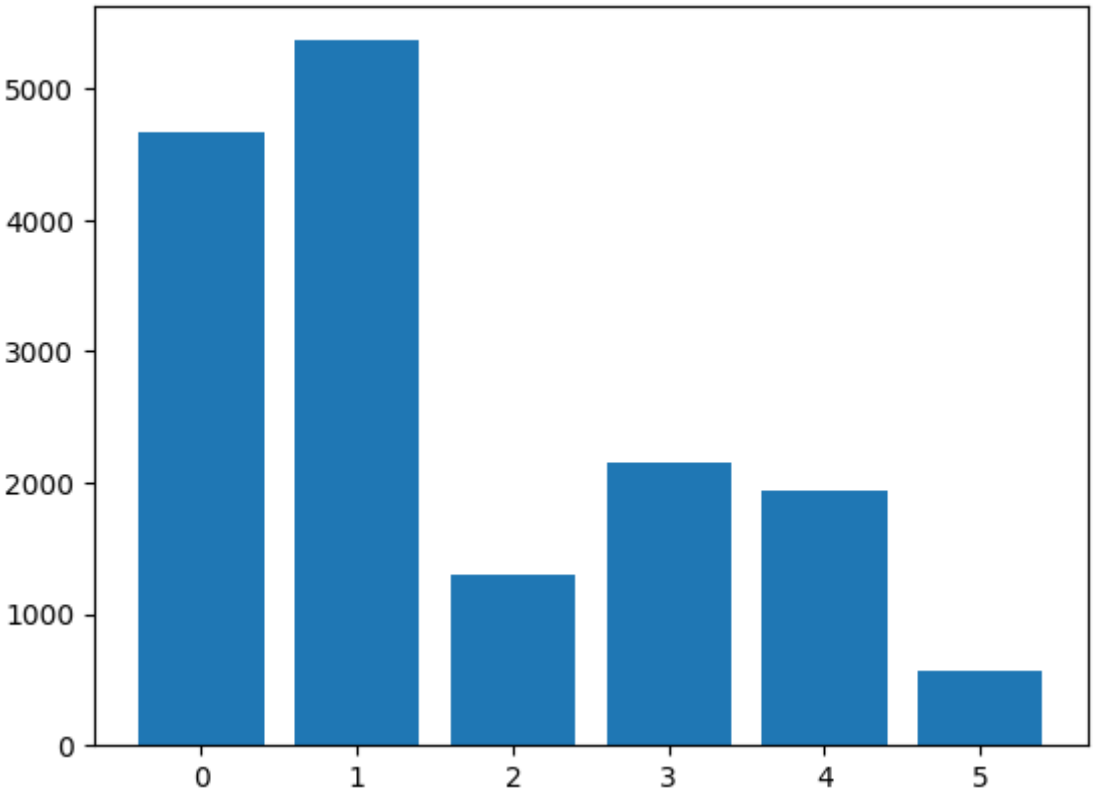


Рисунок 3.1 – Розподіл класів у тренувальному наборі

Як видно із результатів, представлених на рисунку 3.1, класи 2-5 мають значно меншу кількість екземплярів ніж класи 0 та 1. Проте, якщо класи 3 та 4 хоч і мають значно менше прикладів проте їх кількість можна вважати

достатньою, то класи 2 та 5 можна вважати недостатньо представленими. Такий дисбаланс класів може негативно вплинути на здатність моделі правильно навчатися, оскільки вона буде схильна до переваги класів з більшою кількістю прикладів. Це призводить до зниження точності передбачень для рідкісних класів та загального погіршення якості класифікації. Для вирішення цієї проблеми існує кілька основних підходів:

1. Undersampling — зменшення кількості прикладів у переважаючих класах, щоб вирівняти розподіл. Цей метод дозволяє уникнути перенавчання на домінуючі класи, але супроводжується втратою частини цінної інформації.
2. Oversampling — збільшення кількості прикладів у менш представлених класах за рахунок дублювання існуючих або генерації нових прикладів.
3. Зважування втрат (class weighting) — підвищення "вартості" помилок на прикладах рідкісних класів у функції втрат, що спонукає модель більше уваги приділяти цим класам.
4. Синтетичне генерування прикладів (наприклад, SMOTE) — створення нових точок даних для недопредставлених класів шляхом інтерполяції між наявними прикладами.

У даній роботі для боротьби з дисбалансом класів обрано метод оверсемплінгу, як один з найпростіших та ефективних підходів. Його суть полягає у збільшенні кількості прикладів для класів 2 та 5 за допомогою повторного включення наявних екземплярів у навчальну вибірку. Такий підхід дозволяє уникнути втрати даних (на відміну від undersampling) і не вимагає складної модифікації моделі або генерації нових прикладів, як це робиться у SMOTE. Основна перевага оверсемплінгу полягає у його простоті реалізації та збереженні структури оригінальних даних. Крім того, він дозволяє швидко досягти збалансованого розподілу класів, що сприяє покращенню результатів класифікації, особливо у задачах, де важливо враховувати всі категорії. Попри ризик перенавчання, особливо при надмірному дублюванні прикладів,

оверсемплінг залишається надійним методом у ситуаціях, коли кількість даних у недопредставлених класах є критично низькою, як у випадку з класами 2 та 5 у даному дослідженні.

Під час процесу оверсемплінгу я вирішив підвищити кількість екземплярів для класів 2 та 5 до 2000, щоб дані класи були представлені на тому ж рівні, що й 3 та 4.

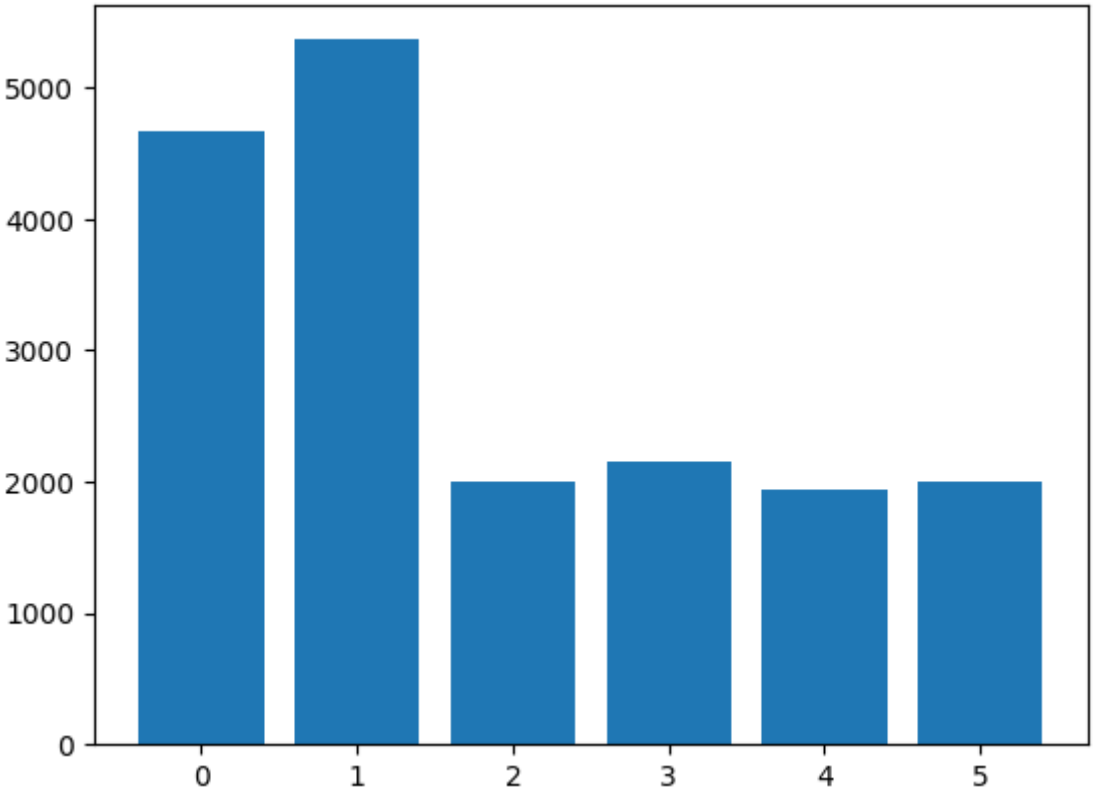


Рисунок 3.2 – Розподіл класів після оверсемплінгу

Згідно із розподілом екземплярів на рисунку 3.2 можна побачити, що кількість прикладів для класів 2 та 5 після процесу оверсемплінгу зросла до рівня класів 3 та 4.

3.1.2. Перетворення тексту у числове представлення

У задачах обробки природної мови за допомогою нейронних мереж, зокрема рекурентних мереж типу LSTM, модель не може напряму працювати з текстовими даними у вигляді звичайного рядка. Тому наступним етапом є перетворення тексту у числове представлення, що відображає структуру й зміст

тексту. Для цього використовується процес токенізації, тобто розбиття тексту на окремі слова або токени та привласнення кожному токену унікального числового індексу. Наприклад, слово "радість" може отримати індекс 17, "сьогодні" — 42 і т.д. В результаті кожен текстовий приклад у наборі даних перетворюється на послідовність чисел, що відображають порядок появи слів у тексті.

Однак після токенізації виникає інша проблема — різна довжина текстів. Один приклад може містити 5 слів, інший — 50. А оскільки нейронні мережі очікують на вхід матрицю фіксованого розміру, необхідно привести всі послідовності до однакової довжини. Для цього застосовується процес, який називається паддінг (padding). Суть його полягає у тому, що до коротших послідовностей додаються спеціальні значення — найчастіше це нулі — щоб усі вектори мали однакову довжину. Наприклад, якщо найдовший текст має 10 токенів, то всі інші послідовності також доповнюються до 10 елементів. Згідно з аналізом текстових повідомлень найдовше повідомлення містить в собі 66 слів, найкоротше всього 2, а медіанне значення дорівнює 17.

3.2. Векторне представлення слів GloVe

Наступним кроком, щоб поліпшити якість моделі до початку розробки самої моделі, є впровадження векторних представлень слів. Для даного завдання було обрано корпус векторизованих слів GloVe. GloVe (Global Vectors for Word Representation) — це метод векторного представлення слів (word embedding), який дозволяє перетворити слова у числові вектори з фіксованою кількістю вимірів. Модель GloVe була розроблена дослідниками зі Стенфордського університету і представлена у 2014 році як альтернатива до інших популярних методів, таких як Word2Vec. Основна мета GloVe — зберегти семантичні та синтаксичні зв’язки між словами у векторному просторі, щоб схожі за значенням слова мали подібні вектори.

На відміну від контекстно-залежних моделей, таких як BERT, GloVe є статичною моделлю: кожному слову у словнику відповідає один вектор,

незалежно від контексту його використання в реченні. Особливістю GloVe є те, що векторизація базується не тільки на локальному контексті слів (як у Word2Vec), а й на глобальній статистиці співзустрічей слів у великому корпусі текстів. Модель будує матрицю співзустрічей (co-occurrence matrix), яка показує, як часто одне слово зустрічається поруч з іншим у певному вікні контексту. Після цього виконується оптимізація, яка намагається знайти векторні представлення слів, що найкраще відображають цю інформацію.

Переваги використання GloVe у задачах емоційного аналізу та інших NLP-задачах:

1. Збереження семантики: слова зі схожим значенням або з однаковим контекстом розміщуються поруч у векторному просторі. Наприклад, вектори для слів «радість» і «щастя» будуть близькими.
2. Покращення якості моделі: векторні представлення GloVe дозволяють моделі отримувати більше інформації про значення слів ще до навчання, що може значно покращити якість класифікації, особливо при обмеженій кількості даних.
3. Ефективність: попередньо натреновані вектори GloVe можуть бути швидко інтегровані у модель без потреби навчання їх з нуля, що економить час і ресурси.
4. Стійкість до лінгвістичного шуму: GloVe дозволяє краще справлятися з синонімами, граматичними варіаціями та іншими особливостями природної мови.

У практичному застосуванні в контексті розроблюваної моделі, GloVe дозволяє кожне слово в послідовності (яка передається на вхід моделі LSTM) представити як вектор заданої довжини, наприклад, 100 або 300 вимірів. Ці вектори використовуються як вхід до нейронної мережі, замінюючи прості числові індекси, що суттєво підвищує інформативність даних для моделі.

3.3. Розробка LSTM моделі

Після підготовки вхідних даних та матриці векторних представлень слів, наступним кроком є побудова самої нейронної мережі. Модель реалізована за допомогою високорівневого API Keras бібліотеки TensorFlow, що забезпечує гнучкість, простоту побудови і навчання нейронних мереж.

```
model = Sequential()
model.add(Embedding(input_dim=len(tokenizer.word_index) + 1, output_dim=embedding_dim,
                    input_length=max_length, weights=[embedding_matrix], trainable=True))
model.add(LSTM(128, dropout=0.3, recurrent_dropout=0.3))
model.add(Dense(64, activation='relu'))
model.add(Dense(6, activation='softmax'))

model.compile(loss='sparse_categorical_crossentropy', optimizer=Adam(learning_rate=0.001), metrics=['accuracy'])
```

Рисунок 3.3 – Код нейронної мережі

Модель створюється як послідовна, що означає лінійне додавання шарів один за одним. Кожен наступний шар отримує на вхід результат попереднього. Архітектура моделі складається з наступних шарів:

1. Embedding-шар

   Першим шаром моделі є Embedding, який перетворює послідовності індексів слів у вектори фіксованої довжини. В даному випадку шар ініціалізується попередньо натренованою матрицею GloVe, яка дозволяє надати кожному слову унікальне векторне представлення на основі статистики співзустрічей у великому текстовому корпусі.

2. LSTM-шар

   Основу моделі складає один прихований шар з рекурентними елементами LSTM (Long Short-Term Memory). Шар має 128 нейронів, які відповідають за запам’ятовування послідовної інформації. Саме LSTM дає змогу моделі розпізнавати довгострокові залежності у вхідному тексті, що критично важливо для задач емоційного аналізу, де значення деяких слів може залежати від контексту попередніх слів у реченні.

3. Повнозв’язний прихований шар (Dense)

   Після шару LSTM йде шар Dense з 64 нейронами та функцією активації

ReLU. Цей шар виконує нелінійне перетворення проміжних ознак, отриманих після LSTM та служить для витягу додаткових абстрактних ознак перед класифікацією.

4. Вихідний шар
   Останнім є шар Dense, який має 6 вихідних нейронів — по одному для кожного з шести класів емоцій. Softmax використовується для перетворення результатів на ймовірнісні значення, які відображають ймовірність того, що приклад належить до кожного з класів.

Перед початком навчання модель необхідно скомпілювати, що означає визначення функції втрат, метрики та оптимізатора:

1. Функція втрат: sparse_categorical_crossentropy — використовується у багатокласових задачах класифікації, коли цільові значення представлені у вигляді індексів класів, а не one-hot векторів.
2. Оптимізатор: Adam з швидкістю навчання 0.001. Adam є адаптивним методом оптимізації, який об'єднує переваги RMSProp і Momentum, забезпечуючи ефективне і швидке зменшення втрат.
3. Метрика: accuracy — стандартна метрика точності, що відображає відсоток правильних передбачень.

3.4. Впровадження навчання з розришеною пам’яттю

Під час розробки обраної моделі було реалізовано механізм навчання з розширеною пам’яттю, який дозволяє моделі зберігати узагальнені представлення для кожного емоційного класу та використовувати цю інформацію під час подальшого навчання. Такий підхід сприяє стабільнішому узагальненню, особливо в умовах складних або близьких між собою емоційних категорій.

Для кожного з шести емоційних класів була реалізована окрема структура пам’яті у вигляді словника Python, де ключем виступає номер класу (мітка), а

значенням — список векторів прихованих станів (hidden states), які генерує LSTM-модель після обробки тексту.

```
self.memory = {i: [] for i in range(num_classes)}
```

Рисунок 3.4 – Структура пам’яті

Ці вектори є щільними представленнями конкретних текстів, що були подані на вхід мережі, й відображають внутрішнє "розуміння" моделі про належність тексту до певної емоційної категорії.

Після кожної обробки батчу даних, модель оновлює свою пам’ять, додаючи до відповідного класу нові приховані вектори як показано на рисунку 3.5.

```
def update_memory(self, hidden_states, labels):
    for h, label in zip(hidden_states, labels):
        label = label.numpy()
        # Додаю новий вектор до пам'яті
        if len(self.memory[label]) < self.memory_limit:
            self.memory[label].append(h)
        else:
            # Якщо пам'ять повна, заміняю найменш важливий вектор
            # Для простоти заміняю найстаріший вектор
            self.memory[label].pop(0)  # Видаляю старий
            self.memory[label].append(h)  # Додаю новий
```

Рисунок 3.5 – Оновлення пам’яті

Таким чином, із часом у кожному емоційному класі накопичується все більше представлень, що дозволяє обчислювати центроїд класу — усереднений вектор, який узагальнює типові ознаки цього емоційного стану.

Під час подальшого навчання модель обчислює відстань між новим вектором (який генерується LSTM для кожного тексту) та усередненим вектором класу з пам’яті. Ця відстань виражається у вигляді MSE (mean squared error) і додається до основної функції втрат як регуляризаційний штраф:

mem_loss += tf.reduce_mean(tf.square(hidden_state - center))

total_loss = loss + λ * mem_loss

де:

— hidden_state — вектор нового прикладу,

— center — центроїд класу з пам’яті,

— λ — ваговий коефіцієнт штрафу (в даній реалізації: λ = 0.01).

Це стимулює модель генерувати представлення, ближчі до узагальненого образу класу, і пригнічує випадкові флуктуації векторів.

Такий підхід дозволяє:

— зменшити розкид векторів одного класу у векторному просторі

— стабілізувати навчання, особливо в умовах класів, які частково перетинаються (наприклад, "сум" і "страх")

— покращити інтерпретованість: середній вектор кожного класу можна вважати його "емоційною сутністю".

Окрім того, під час оцінки ефективності моделі було проведено тестування роботи даного механізму за умови обмеження розміру пам’яті. У випадку обмеження списку векторів до 50 елементів для кожного класу час тренування скоротився вдвічі без значного зниження значень оціночних метрик.

3.5. Впровадження кросс-валідації

У процесі побудови машинного навчання важливо не лише досягти високої точності на навчальних даних, але й переконатися, що модель здатна узагальнювати — тобто демонструвати стабільні результати на нових, раніше не бачених прикладах. Одним із основних інструментів для перевірки узагальнювальної здатності моделі, використаним під час розробки, є кросс-валідація. Кросс-валідація — це метод оцінки якості моделі шляхом її

багаторазового тренування і тестування на різних підмножинах даних. Замість того, щоб фіксувати один розподіл на тренувальний та валідаційний набори, кросс-валідація дозволяє більш надійно і об'єктивно оцінити продуктивність моделі на різних частинах даних. Найпоширенішим видом кросс-валідації є K-Fold Cross-Validation, де:

1. Весь датасет розбивається на *K* приблизно рівних частин.
2. Процес повторюється *K* разів:
    I) На кожній ітерації одна із частин використовується як валідаційна, а решта *(K-1)* частин — для навчання.
    II) Модель тренується на навчальній частині і оцінюється на валідаційній.
3. Результати кожної ітерації (наприклад, точність) зберігаються.
4. Після завершення всіх ітерацій обчислюється середнє значення метрик, яке дає об'єктивну оцінку якості моделі.

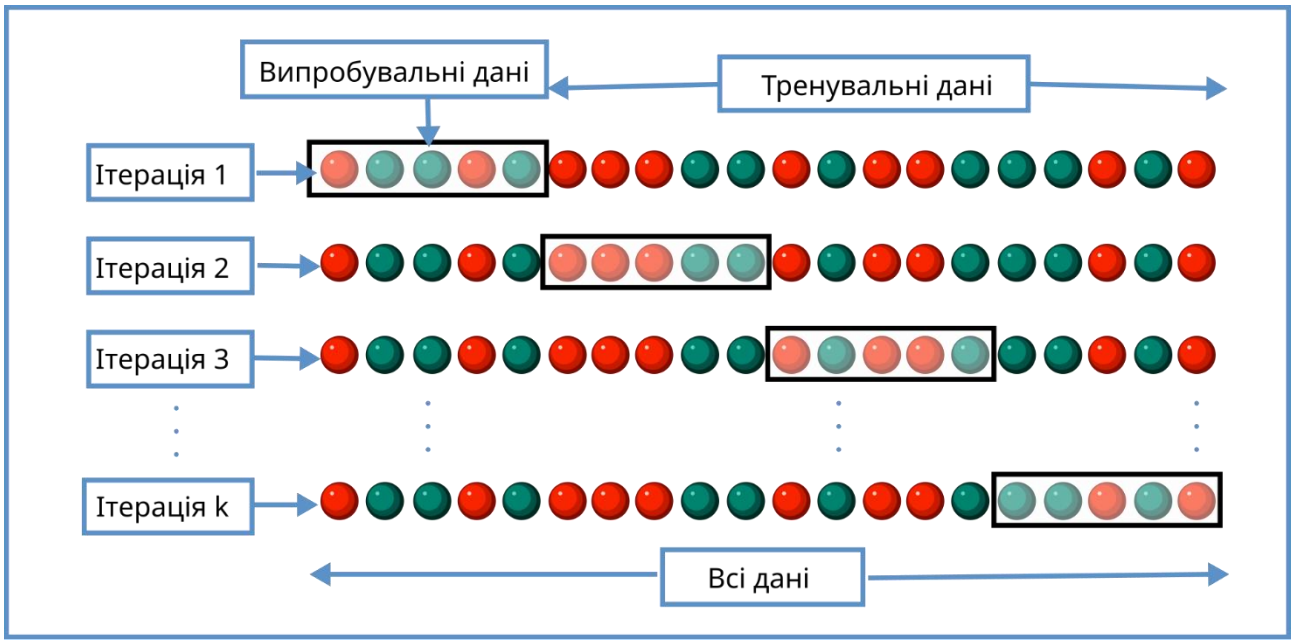


Рисунок 3.6 – Схема роботи кросс-валідації

В розробленій моделі використовується Stratified K-Fold Cross-Validation з 5-ма частинами (K=5). Це означає, що кожна частина зберігає приблизно однакове співвідношення класів, як і в повному датасеті, що особливо важливо для недопредставлених класів.

Використання кросс-валідації надає низку важливих переваг:

1. Зменшення впливу випадковості: модель оцінюється на різних підмножинах, що дозволяє уникнути залежності від конкретного розбиття.
2. Більш точна оцінка: середнє значення метрик з усіх частин краще відображає реальну продуктивність моделі.
3. Підбір параметрів: кросс-валідація часто використовується для перевірки різних гіперпараметрів без ризику перенавчання на одному фіксованому наборі.
4. Робота з обмеженою кількістю даних: дозволяє максимально використати наявні дані — кожен приклад потрапляє і в тренувальний, і у валідаційний набір хоча б один раз.

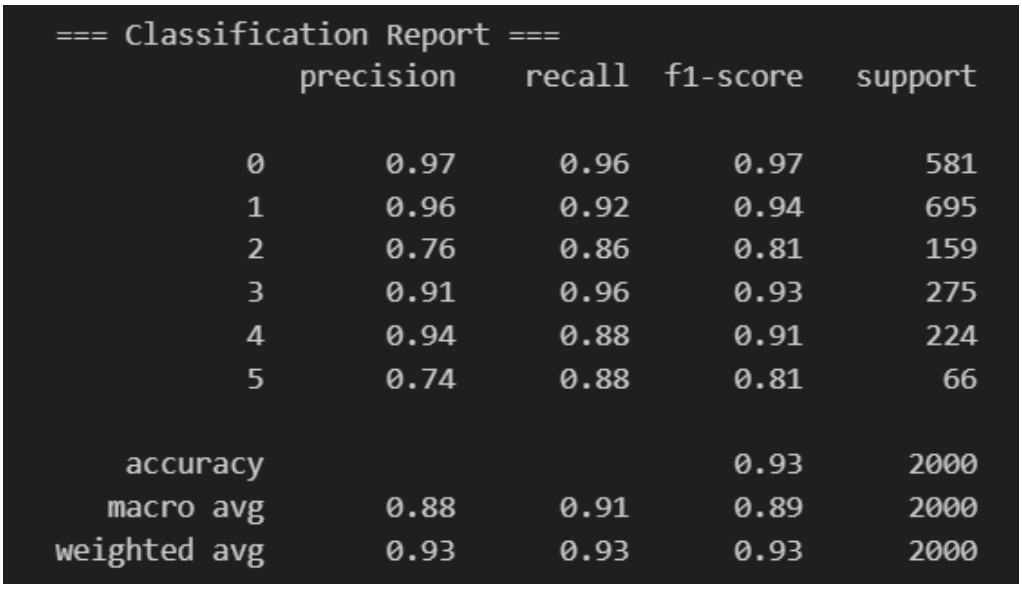
Таким чином, кросс-валідація є ключовим інструментом для оцінки якості та стабільності розробленої моделі.

3.6. Тестування та оцінка розробленої моделі

Після завершення процесу кросс-валідації та фінального навчання моделі, було проведено її оцінювання на валідаційних та тестових даних. З метою перевірки здатності моделі до узагальнення, використовувалась 5-Fold кросс-валідація, у межах якої середнє значення валідаційної точності склало 94.21%.

На основі отриманих результатів модель була повторно навчена на повному об'єднаному тренувальному наборі (який включає в себе початкові тренувальні та валідаційні дані) та протестована на відкладеному тестовому наборі. В результаті було отримано наступні метрики:

1. Accuracy (точність класифікації) - загальна частка правильних передбачень серед усіх зразків. Результат: 0.9310.
2. Precision (макросереднє) - показує, яка частина передбачених як певний клас дійсно належить до нього. Висока precision означає, що модель робить мало хибнопозитивних передбачень. Результат: 0.8807.
3. Recall (макросереднє) - показує, яку частку з усіх прикладів певного класу модель змогла правильно розпізнати. Високий recall означає, що модель майже не пропускає потрібні випадки. Результат: 0.9086.
4. F1-score (макросереднє) - це середнє значення між precision та recall. Використовується тоді, коли обидві метрики важливі. Результат: 0.8929.
5. F2-score (макросереднє) - схожа на F1, але більше уваги приділяє recall. Корисна тоді, коли важливо не пропустити жодного позитивного прикладу, навіть якщо буде більше помилкових передбачень. Результат: 0.9019.
6. MCC (Коефіцієнт кореляції Метью) - значення коливається від -1 (повна помилка) до 1 (ідеальна модель), 0 означає випадкове передбачення. Результат: 0.9042.
7. AUROC – значення в діапазоні від 0 до 1. Результат: 0.9960.



=== Classification Report ===

	precision	recall	f1-score	support
0	0.97	0.96	0.97	581
1	0.96	0.92	0.94	695
2	0.76	0.86	0.81	159
3	0.91	0.96	0.93	275
4	0.94	0.88	0.91	224
5	0.74	0.88	0.81	66
accuracy			0.93	2000
macro avg	0.88	0.91	0.89	2000
weighted avg	0.93	0.93	0.93	2000

Рисунок 3.7 – Звіт з класифікації покращеної моделі

Також було згенеровано детальний звіт з класифікації, який демонструє ефективність моделі для кожного з класів окремо на рисунку 3.4. Дані метрики дозволяють комплексно оцінити як загальну ефективність моделі, так і її збалансованість у розпізнаванні кожної категорії.

Окрім цього на рисунку 3.8 зображений загальний прогрес точності моделі під час кросс-валідації та фінального навчання.

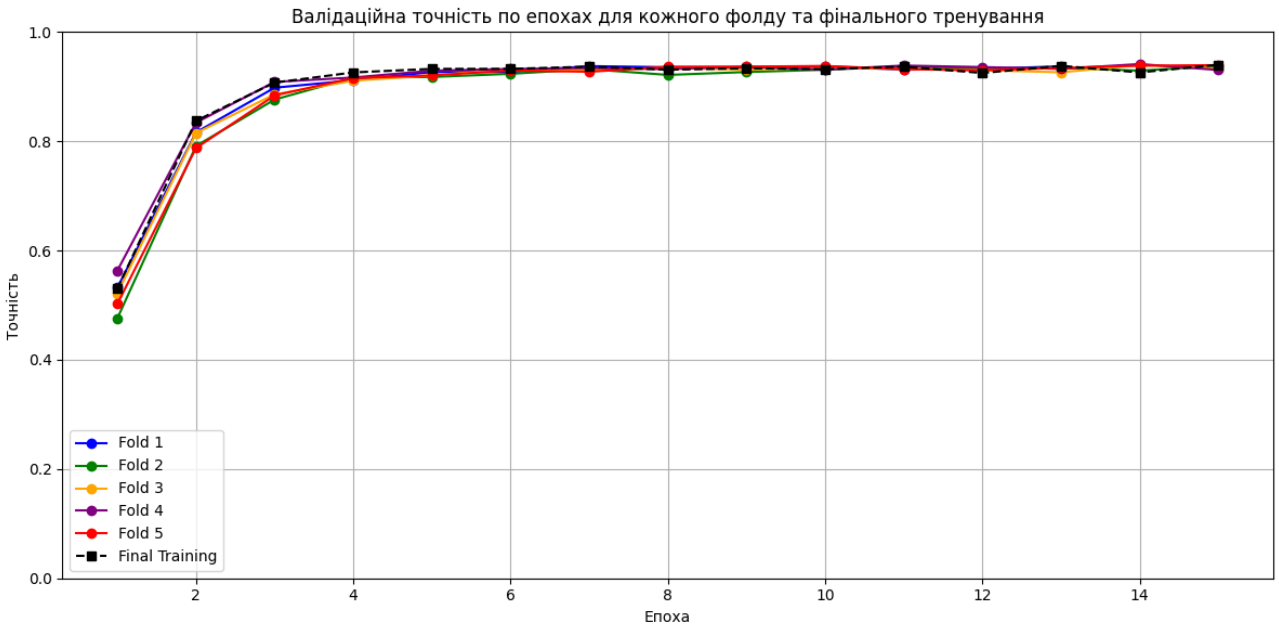


Рисунок 3.8 – Прогрес валідаційної точності моделі

На рисунку 3.9 представлено графік зміни функції втрат на тренувальних та валідаційних даних протягом епох фінального тренування моделі.

Як видно з графіка:

1. На початкових етапах навчання значення функції втрат як на тренувальній, так і на валідаційній вибірках стрімко зменшуються, що свідчить про швидке пристосування моделі до основних закономірностей даних.
2. Починаючи з приблизно 4–5 епохи, темпи зменшення втрат сповільнюються, а криві тренувальної та валідаційної втрат стабілізуються на низькому рівні.

3. Значення валідаційної втрати залишається близьким до тренувальної протягом усього процесу навчання, без суттєвого збільшення на останніх етапах, що свідчить про відсутність перенавчання (overfitting).
4. Остаточні значення втрат є дуже низькими (близько 0.02–0.03), що демонструє високу якість побудованої моделі на як тренувальних, так і на валідаційних даних.

Таким чином, аналіз графіка втрат підтверджує стабільність навчального процесу моделі та її здатність ефективно узагальнювати закономірності у даних без суттєвого перенавчання.

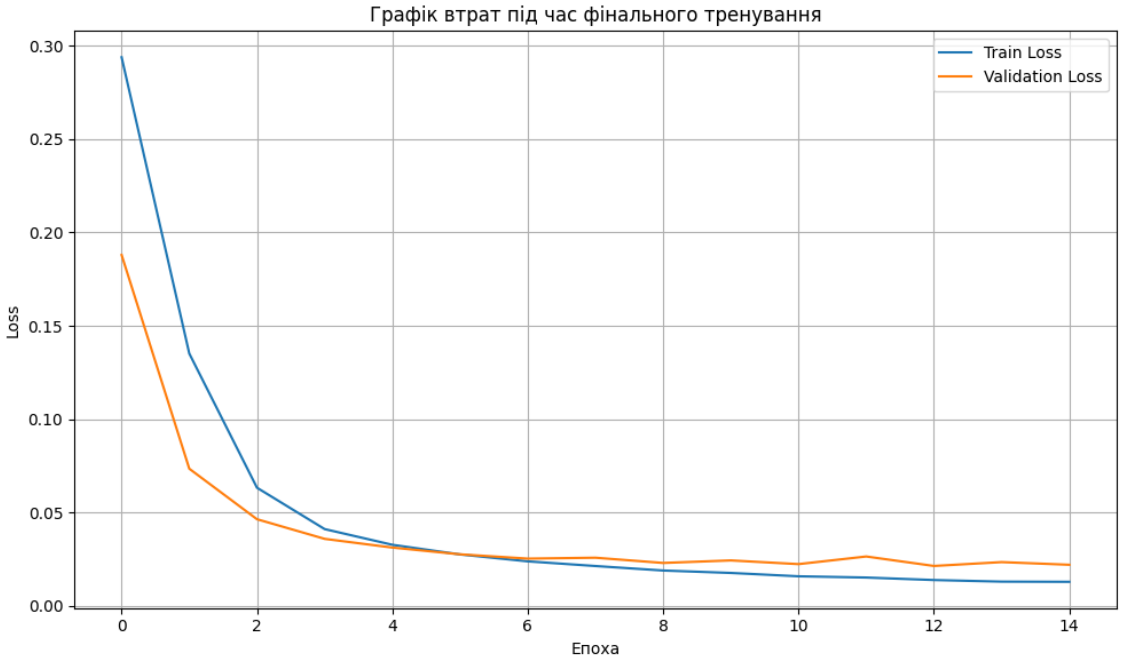


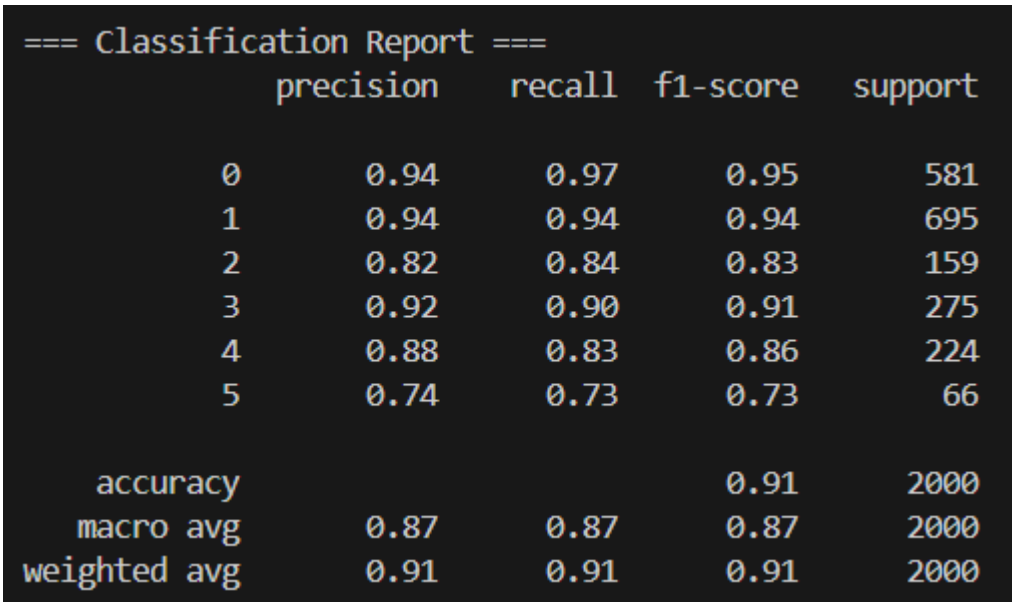
Рисунок 3.9 – Прогрес функції втрат

Для порівняння результатів поліпшеної моделі було проведено тестування стандартної LSTM-моделі без описаних раніше змін. Після завершення навчання було отримано такі результати:

1. Accuracy (точність класифікації): 0.9145
2. Precision (макросереднє): 0.8728
3. Recall (макросереднє): 0.8678
4. F1-score (макросереднє): 0.8701

5. F2-score (макросереднє): 0.8687
6. MCC (Matthews Correlation Coefficient): 0.8868
7. AUROC: 0.8940

Звіт з класифікації надав такі значення:



```
=== Classification Report ===
              precision    recall  f1-score   support

           0       0.94      0.97      0.95       581
           1       0.94      0.94      0.94       695
           2       0.82      0.84      0.83       159
           3       0.92      0.90      0.91       275
           4       0.88      0.83      0.86       224
           5       0.74      0.73      0.73        66

    accuracy                           0.91      2000
   macro avg       0.87      0.87      0.87      2000
weighted avg       0.91      0.91      0.91      2000
```

Рисунок 3.10 – Звіт з класифікації стандартної моделі

Покращена LSTM-модель демонструє стабільне зростання за всіма ключовими метриками, особливо помітне покращення recall та F2-метрики, що вказує на здатність моделі краще розпізнавати істинно позитивні випадки. Це особливо важливо в задачах, де визначення позитивного класу як хибно негативного коштує дорого.

Окрім того, для кращої демонстрації результатів роботи моделі було сформовано матрицю невідповідностей, яку можна побачити на рисунку 3.11.

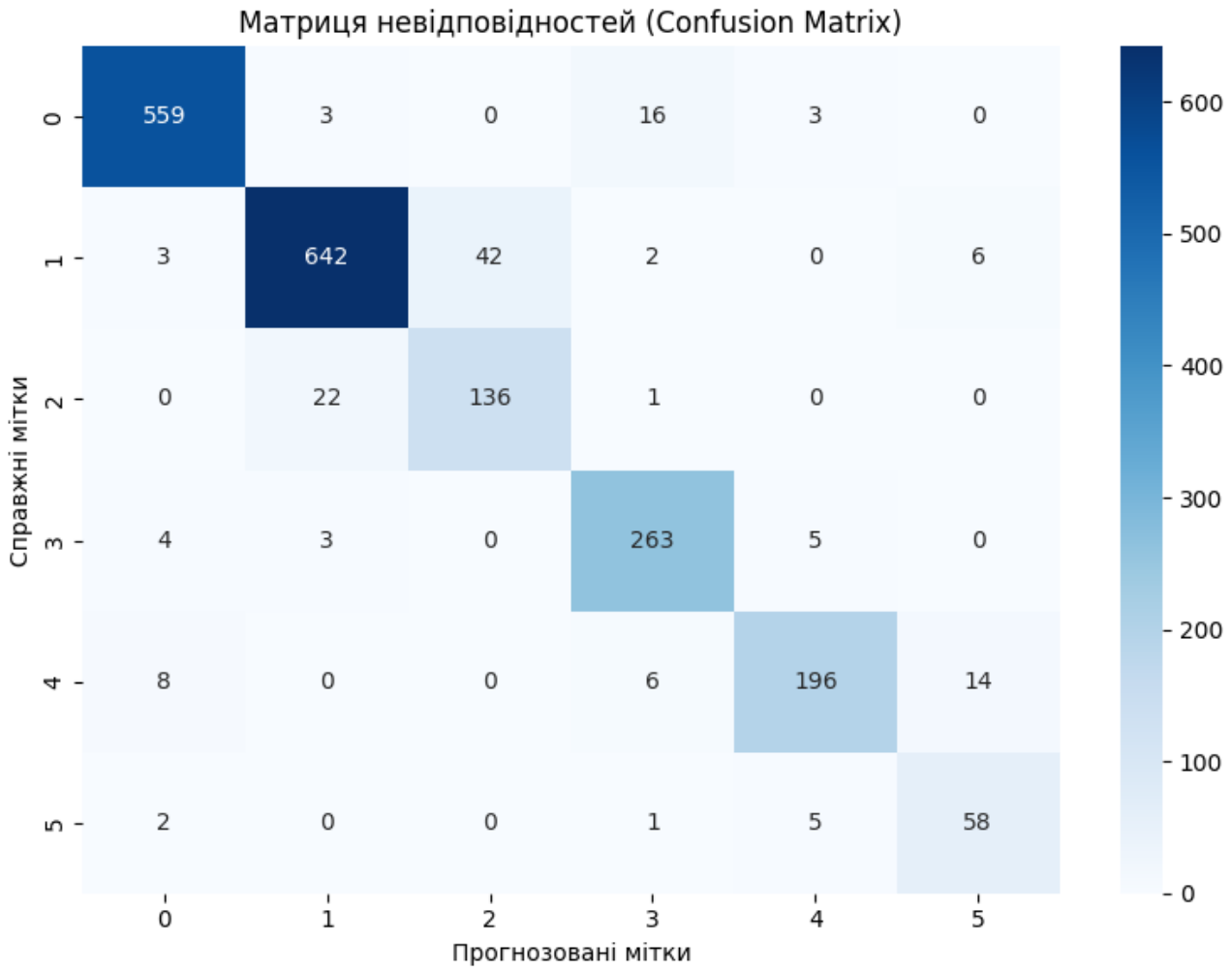


Рисунок 3.11 – Матриця невідповідностей

Отримані результати підтверджують здатність моделі ефективно вирішувати задачу класифікації емоційного забарвлення текстів, при цьому демонструючи достатній рівень узагальнення на нових, раніше не бачених прикладах.

Висновки

В даному розділі було реалізовано експериментальну частину дослідження, що охоплює повний цикл побудови моделі емоційного аналізу тексту з використанням сучасних методів глибокого навчання. Усі етапи реалізації були структуровані відповідно до методології CRISP-DM, що дозволило послідовно організувати процес — від підготовки даних до аналізу результатів.

Основою моделі стала рекурентна нейронна мережа типу LSTM, здатна ефективно працювати з послідовностями природної мови. Для покращення якості представлення тексту було інтегровано попередньо навчені векторні представлення GloVe, що забезпечило збагачення ембеддинг-простору семантичною інформацією.

У процесі тренування моделі було реалізовано:

- Dropout-регуляризацію для зниження ймовірності перенавчання,
- L2-регуляризацію на рівні ваг, що допомогло згладити модель і уникнути переускладнення,
- використання функції втрат Focal Loss, яка зменшує вплив домінування частих класів і підвищує чутливість до рідкісних емоційних категорій.

Ключовим нововведенням стало впровадження підходу навчання з розширеною пам’яттю. Модель зберігає внутрішні представлення (приховані стани) для кожного емоційного класу, формуючи центр пам’яті (центроїд), з яким порівнювались нові приклади під час навчання. Дистанція між поточним вектором і центром класу використовується як додаткова компонента функції втрат, що стимулює узгодженість представлень усередині кожного класу.

Результати тестування підтвердили ефективність розробленого підходу — отримано високі значення точності, F1-міри та AUROC. Аналіз матриці неточностей і графіків навчання дозволив виявити складні для класифікації випадки, а також переконатися в стабільності моделі в умовах дисбалансу даних.

Таким чином, запропонована архітектура, яка поєднує LSTM, GloVe, Focal Loss, регуляризаційні техніки та пам’ять класів, дозволила досягти високих результатів в задачі багатокласової емоційної класифікації тексту.

РОЗДІЛ 4. ВПРОВАДЖЕННЯ РОЗРОБЛЕНОЇ ТЕХНОЛОГІЇ

4.1. Технології застосування розробленої моделі

Розроблена модель на основі LSTM (Long Short-Term Memory) є ефективним інструментом для автоматичної класифікації текстових повідомлень за емоційним або тематичним змістом. Для забезпечення практичного використання даної моделі у реальних умовах, необхідно передбачити механізм її інтеграції у програмні продукти або сервіси. Розглянемо основні підходи до організації взаємодії користувача з моделлю та можливі сценарії її застосування.

1. Веб-сервіс (API)

Найбільш поширеним способом застосування моделі є розгортання її у вигляді веб-сервісу (наприклад, REST API), до якого можна надсилати HTTP-запити з текстовими даними. Серверна частина приймає запити, обробляє текст за допомогою токенізатора, викликає модель для передбачення класу та повертає результат у відповідь.

Для реалізації такого рішення можуть використовуватись:

— Flask або FastAPI – легкі Python-фреймворки для створення REST API.

— Gunicorn / Uvicorn – сервери для розгортання сервісу.

— Docker – для контейнеризації і зручного перенесення.

Такий підхід дозволяє інтегрувати модель у веб-додатки, мобільні застосунки або сторонні інформаційні системи.

2. Хмарні платформи

Іншим ефективним способом впровадження є використання хмарних платформ для зберігання, обробки та масштабування моделі. Серед найпопулярніших рішень:

— Google Cloud AI Platform

— Amazon SageMaker

— Microsoft Azure Machine Learning

Ці платформи дозволяють:

— автоматично масштабувати інстанси при зростанні навантаження

— інтегрувати логування, моніторинг та автооновлення моделі

— реалізовувати CI/CD-підхід для деплою нових версій

Хмарні сервіси є зручними для командної роботи та для випадків, коли важливо забезпечити високу доступність і безперервність обслуговування.

3. Вбудоване застосування

У випадках, коли необхідна робота моделі без постійного доступу до інтернету (наприклад, у мобільному застосунку або десктопному ПЗ), модель може бути експортована в оптимізований формат (наприклад, TensorFlow Lite, ONNX) та інтегрована безпосередньо у клієнтську програму.

Цей підхід зменшує час відгуку, підвищує конфіденційність даних і дозволяє уникати витрат на серверну інфраструктуру, проте вимагає додаткових оптимізацій моделі (зменшення розміру, кількості параметрів).

В контексті впровадження розробленої моделі на базі практики найбільш зручним та ефективним рішенням може стати розгортання моделі штучного інтелекту на хмарних платформах для зменшення навантаження на локальні серверні потужності та забезпечення зручного одночасного доступу до моделі відповідним відділам підприємства.

4.2. Способи хмарного розгортання розробленої моделі

Хмарне розгортання моделі машинного навчання дозволяє масштабувати рішення, забезпечити постійний доступ до нього через інтернет та інтегрувати модель у більші програмні продукти. Найбільш поширеними хмарними

платформами є Google Cloud Platform (GCP), Amazon Web Services (AWS) та Microsoft Azure.

Розпочнемо з детальної покрокової інструкціїї для розгортання моделі на Google Cloud Platform (GCP). Для цього підприємство може використатити Vertex AI.

1. Збереження моделі у форматі SavedModel або HDF5 (.h5)
2. Завантаження моделі в Google Cloud Storage: відповідальна особа має створити бакет (сховище) в Google Cloud Storage та завантажити модель у бакет використовуючи графічний інтерфейс або за допомогою команди gsutil cp -r lstm_model gs://your-bucket-name/
3. Створення моделі в Vertex AI: необхідно відкрити вкладку Models в Vertex AI, натиснути "Upload model", вказати шлях до моделі в GCS, обрати фреймворк (TensorFlow, версія), і натиснути "Deploy to endpoint".
4. Розгортання на ендпоінті: після створення моделі, натиснути “Deploy to endpoint” та оберіть ресурс (машину), на якій буде працювати модель.
5. Використання моделі: через REST API можна надсилати запити до ендпоінта. Приклад:

```
curl -X POST -H "Authorization: Bearer $(gcloud auth print-access-token)" \
-H "Content-Type: application/json" \
-d '{"instances": [{"input": [токенізований_текст]}]}' \
https://REGION-aiplatform.googleapis.com/v1/projects/PROJECT/locations/REGION/endpoints/ENDPOINT_ID:predict
```

Для розгортання на Amazon Web Services (AWS) слід використати Amazon SageMaker:

1. Підготовка моделі: експортувати модель у форматі SavedModel або .h5.

2. Завантажити у S3: створити бакет у S3 (Simple Storage Service) та завантажте модель, використовуючи графічний інтерфейс або за допомогою команди aws s3 cp lstm_model/ s3://your-bucket-name/lstm_model/ --recursive

3. Створити та ініціалізувати ноутбук: увійти у SageMaker, створити інстанс ноутбука, у ноутбуці імпортувати необхідні бібліотеки «from sagemaker.tensorflow import TensorFlowModel».

4. Імпортувати модель, використовучи приклад коду з рисунку 4.1.

```
model = TensorFlowModel(model_data='s3://your-bucket-name/lstm_model/model.tar.gz',
                        role='your-sagemaker-execution-role',
                        framework_version='2.11.0')
```

Рисунок 4.1 – Приклад коду для імпорту моделі в ноутбук на AWS

5. Розгорнути модель за допомогою коду «predictor = model.deploy(initial_instance_count=1, instance_type='ml.m5.large')»

6. Використання моделі: надсилати запити до ендпоінта через HTTP або прямо з Python «predictor.predict({'instances': [токенізований_текст]})»

Останній з найбільших і найпопулярніших хмарних сервісів Microsoft Azure. Для розгортання моделі необхідно виконати такі кроки, використовуючи Azure Machine Learning:

1. Підготовка середовища: встановити CLI або використати Azure ML Studio.

2. Реєстрація моделі**:** зберегти модель (.h5 або SavedModel) локально або завантажити її до Azure, використовуючи приклад коду з рисунку 4.2.

```
from azureml.core import Workspace, Model
ws = Workspace.from_config()
model = Model.register(workspace=ws, model_path="lstm_model/", model_name="lstm_model")
```

Рисунок 4.2 – Приклад коду для завантаження моделі до Azure

3. Створення середовища виконання: створити inference_config із залежностями.

```
from azureml.core.environment import Environment
from azureml.core.model import InferenceConfig

env = Environment.from_conda_specification(name="lstm-env", file_path="env.yml")
inference_config = InferenceConfig(entry_script="score.py", environment=env)
```

Рисунок 4.3 – Приклад створення inference_config із залежностями

4. Розгортання моделі: запакувати модель в інстанс.

```
from azureml.core.webservice import AciWebservice
deployment_config = AciWebservice.deploy_configuration(cpu_cores=1, memory_gb=1)
service = Model.deploy(workspace=ws,
                       name="lstm-service",
                       models=[model],
                       inference_config=inference_config,
                       deployment_config=deployment_config)
service.wait_for_deployment(show_output=True)
```

Рисунок 4.4 – Приклад створення інстансу

5. Використання моделі: можливе використання через REST-запити до отриманого URL сервісу або через SDK, як показано на рисунку 4.5.

```
import requests
response = requests.post(service.scoring_uri, json={"data": ['токенізований_текст']})
```

Рисунок 4.5 – Приклад використання через SDK

Усі три хмарні платформи пропонують ефективні засоби для розгортання моделей глибокого навчання. Google Cloud відзначається простою інтеграцією для TensorFlow-моделей, AWS надає найширші можливості кастомізації та автоматизації, а Azure має гарну інтеграцію з корпоративними інструментами Microsoft. Вибір платформи залежить від потреб проєкту, бюджету та досвіду команди. Порівняння усіх представлених сервсів представлено в таблиці 4.1.

Таблиця 4.1 - Порівняльна характеристика хмарних сервісів

Характеристика	**Google Cloud (Vertex AI)**	**AWS (SageMaker)**	**Azure (Machine Learning)**
Простота налаштування	Висока	Середня	Середня
Автоматичне масштабування	Так	Так	Так
Підтримка різних форматів моделей	TensorFlow, Scikit-learn, PyTorch	TensorFlow, PyTorch, XGBoost	TensorFlow, ONNX, PyTorch
Інтеграція з іншими сервісами	GCS, BigQuery, Colab	S3, EC2, Lambda	Azure Storage, Logic Apps
Вартість	Відносно доступна	Гнучка, може бути вища	Середня
Зручність REST API	Висока	Висока	Висока
Моніторинг та логування	Vertex AI Logging	CloudWatch	Azure Monitor

4.3. Технічо-економічне обґрунтування застосування

Запропонована модель глибокого навчання для класифікації емоцій в текстах демонструє високу точність та узагальнюючу здатність, що підтверджується результатами метрик оцінки. Впровадження цієї моделі у виробниче середовище може значно покращити якість автоматизованого аналізу текстових даних, що є актуальним для різноманітних галузей: від служби підтримки користувачів до медіааналітики, HR-сфер, маркетингу тощо.

З технічної точки зору, модель має такі переваги:

1. Висока точність: досягнута точність (accuracy ≈ 93%) дозволяє впевнено використовувати модель у реальних сценаріях.
2. Масштабованість: модель можна розгорнути на хмарних платформах, які підтримують автоматичне масштабування під навантаження (Google Vertex AI, AWS SageMaker, Azure ML).

3. Гнучкість інтеграції: модель легко інтегрується через REST API у веб-додатки, CRM-системи, мобільні застосунки тощо.
4. Мінімальні технічні вимоги: після оптимізації модель може працювати на невисокопродуктивних машинах або контейнерах, що знижує вартість інфраструктури.
5. Можливість подальшого навчання: модель підтримує додаткове навчання на нових даних (fine-tuning), що дозволяє адаптувати її до специфічних потреб конкретного бізнесу або мовного контексту.

З економічної точки зору, впровадження моделі дозволяє досягти таких вигід:

1. Зниження витрат на персонал: автоматизований аналіз емоцій у повідомленнях або коментарях зменшує потребу у великій кількості аналітиків або модераторів.
2. Підвищення якості обслуговування клієнтів: швидке виявлення негативних емоцій у зверненнях дозволяє оперативно реагувати та покращувати клієнтський досвід, що позитивно впливає на лояльність клієнтів.
3. Можливість повторного використання моделі: модель є універсальною — її можна використовувати в декількох підрозділах організації (служба підтримки, відділ маркетингу, HR тощо), що підвищує ефективність інвестицій.
4. Низька вартість розгортання: у випадку використання хмарних сервісів за моделлю «оплата за використання» організація уникає значних витрат на власне серверне обладнання.
5. Прогнозований ефект від впровадження: згідно з розрахунками, використання автоматизованої класифікації емоцій може зменшити час обробки звернень клієнтів у 2–3 рази, що призведе до скорочення операційних витрат.

З урахуванням технічних і економічних факторів, впровадження розробленої моделі є доцільним та ефективним. Її використання сприяє підвищенню продуктивності, якості обслуговування, оперативності прийняття рішень і зменшенню витрат. Таким чином, модель має великий потенціал для впровадження в комерційні та корпоративні системи, а також у науково-дослідні проєкти.

4.4. Розрахунок витрат на розгортання та підтримку моделі емоційного аналізу

Одним із важливих етапів оцінки ефективності розробленої системи є аналіз витрат, пов’язаних із її впровадженням та експлуатацією. Враховуючи, що емоційний аналіз буде виконуватися раз на тиждень, витрати на обчислювальні ресурси будуть значно меншими порівняно зі щоденною обробкою.

Нижче наведено орієнтовний розрахунок вартості розгортання моделі емоційного аналізу на прикладі хмарної інфраструктури та локального серверного рішення.

4.4.1. Витрати на хмарну інфраструктуру

Для виконання аналізу даних раз на тиждень достатньо короткочасного використання обчислювальних ресурсів. Припустимо, що на проведення повного аналізу витрачається близько 2 годин.

Основні витрати на місяць складатимуть:

— Оренда віртуального сервера з GPU (наприклад, AWS g4dn.xlarge) — близько 0,60 USD/год.

— 2 години × 4 рази = 8 годин роботи на місяць.

Відповідно, витрати на обчислення:

— 0,60 USD × 8 годин = 4,80 USD/місяць.

Додатково:

— Сховище даних (Amazon S3) для зберігання моделей і результатів (~50 ГБ): 50 × 0,023 USD = 1,15 USD/місяць.

— Трафік даних (~10 ГБ на місяць): 10 × 0,09 USD = 0,90 USD/місяць.

Загальні щомісячні витрати на хмарну інфраструктуру:

— 4,80 + 1,15 + 0,90 = 6,85 USD/місяць.

Тобто витрати на хмарні обчислення будуть складати лише близько 6,85 USD на місяць, або 82,2 USD на рік.

4.4.2. Витрати на локальну інфраструктуру

При розгортанні рішення на власному сервері витрати включають одноразові витрати на закупівлю обладнання:

— Сервер із GPU (наприклад, NVIDIA RTX 4060) — орієнтовно 600-1500 USD.

— Накопичувач SSD об'ємом 1 ТБ — близько 100 USD.

Оскільки навантаження на обладнання буде мінімальним (раз на тиждень), витрати на електроенергію також будуть незначними. Припустимо:

— Споживання 300 Вт × 2 години × 4 рази = 2,4 кВт·год/місяць.

— При середній вартості електроенергії 0,10 USD за 1 кВт·год витрати складуть 2,4 × 0,10 = 0,24 USD/місяць або 2.88 USD/рік.

Загальні витрати за перший рік для локального розгортання:

— Придбання обладнання: 1600 USD.

— Електроенергія: 2.88 USD.

Таблиця 4.2 - Порівняльний аналіз варіантів розгортання

Параметр	**Хмарна інфраструктура**	**Локальний сервер**
Початкові витрати	0 USD	1600 USD
Щомісячні витрати	6.85 USD	0.24 USD
Орієнтовні витрати за перший рік	82.2 USD	1602.88 USD
Гнучкість масштабування	Висока	Обмежена
Контроль над даними	Середній	Повний

Як видно з аналізу, за умови невисокої частоти використання (раз на тиждень) значно економічніше застосовувати хмарну інфраструктуру. Локальний сервер буде доцільним лише у випадках, коли контроль над даними є критичним або планується масштабування системи у майбутньому.

Висновки

В даному розділі було виконано техніко-економічне обґрунтування створення та впровадження системи емоційного аналізу текстів на базі методів машинного навчання. Проведено аналіз ресурсів, необхідних для реалізації проєкту, зокрема витрат часу на розробку, витрат на програмне та апаратне забезпечення, а також оцінено економічну доцільність застосування розробленої системи в умовах підприємства.

Було враховано сучасні тенденції автоматизації аналізу зворотного зв’язку від клієнтів та співробітників, що підтверджує актуальність запропонованого рішення для бізнес-середовища. Окрему увагу приділено визначенню періодичності використання моделі на практиці, а також потенційній економії часу та ресурсів за рахунок автоматичного визначення емоційної складової текстових повідомлень.

Розрахунки засвідчили, що впровадження такої системи є економічно обґрунтованим і може забезпечити значну додану вартість для організацій, які працюють із великими обсягами текстових даних. Застосування моделі дозволяє

знизити витрати на ручний аналіз, покращити якість обслуговування клієнтів та прискорити прийняття управлінських рішень на основі емоційної аналітики.

Таким чином, розроблена система емоційного аналізу не лише відповідає сучасним технологічним вимогам, але й є ефективною з погляду ресурсів, термінів розробки та практичної вигоди для потенційного замовника.

ЗАГАЛЬНІ ВИСНОВКИ

У межах даної кваліфікаційної роботи було проведено аналіз існуючих методів класифікації емоцій у текстах, зокрема класичних підходів на основі словників емоцій та машинного навчання, а також сучасних моделей глибокого навчання, таких як RNN, GRU та LSTM. У результаті порівняння було обґрунтовано вибір LSTM як базової архітектури, що здатна ефективно працювати з послідовними текстовими даними завдяки збереженню контексту у часі. Було розглянуто структуру LSTM, її особливості та переваги, які роблять цю модель придатною для задач емоційної класифікації.

На основі отриманих теоретичних знань було успішно розроблено та реалізовано модель на основі архітектури LSTM для автоматичної класифікації емоцій у текстах. Проєкт охопив повний цикл створення інтелектуальної системи згідно з методологією CRISP-DM — від збору та обробки даних до побудови, оптимізації, оцінки якості моделі та розгортання її в хмарному середовищі. З метою покращення ефективності класифікації було проведено балансування класів, використано попередньо навчені векторні представлення слів (GloVe), впроваджено механізм навчання з розширеною пам’яттю, додатко регуляризатори, а також реалізовано крос-валідацію, що дозволило об’єктивно оцінити якість моделі на різних підмножинах даних.

Результати експериментів засвідчили, що покращена модель LSTM значно перевершує базову версію за основними метриками якості. Зокрема, точність класифікації зросла з 91.20% до 93.10%, макро-усереднені показники precision, recall, F1 та F2 зросли на 2.28% та 3.32% відповідно, що свідчить про покращення здатності моделі виявляти менш репрезентовані емоційні категорії, а зростання показника AUROC на 7.2% вказує, що модель здатна значно краще розрізняти класи. Зросло і значення метрики MCC, а саме на 1.72%, що відображає зростання загальної збалансованості класифікації по всіх класах.

У рамках дослідження також було визначено шляхи практичного застосування розробленої моделі. Вона може бути інтегрована у веб- або мобільні додатки, корпоративні аналітичні системи або сервіси обробки зворотного зв’язку клієнтів. Крім того, було розглянуто декілька варіантів розгортання моделі в хмарному середовищі, зокрема за допомогою Google Cloud, Amazon Web Services та Microsoft Azure. Для кожної платформи була розроблена покрокова інструкція, що дозволяє швидко та зручно реалізувати інтеграцію без значних витрат часу та ресурсів.

Оцінка технічної та економічної доцільності впровадження показала, що розроблена технологія є не лише ефективною, але й економічно виправданою. Використання хмарної інфраструктури забезпечує гнучкість масштабування, надійність зберігання, швидкість доступу та мінімізацію витрат на підтримку власної серверної інфраструктури. Таким чином, запропоноване рішення має широкі перспективи для впровадження в реальні продукти, які орієнтовані на автоматизований аналіз емоційного забарвлення текстів.

СПИСОК ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Nkhata G., Anjum U., Zhan J. Sentiment Analysis of Movie Reviews Using BERT. arXiv preprint arXiv:2502.18841. – 2025.
2. Avila C. S. R. Tweet Influence on Market Trends: Analyzing the Impact of Social Media Sentiment on Biotech Stocks. arXiv preprint arXiv:2402.03353. – 2024.
3. Salcedo Gallo J. S., Solano J., García J. H., Zarruk-Valencia D., Correa-Bahnsen A. Proactive Detractor Detection Framework Based on Message-Wise Sentiment Analysis Over Customer Support Interactions. arXiv preprint arXiv:2211.03923. – 2022.
4. Dhamma Y. A., Barus S. P. Sentiment Analysis on Google Reviews Using Naïve Bayes, K-Nearest Neighbors, and Logistic Regression to Improve Novotel Services. Journal of Applied Informatics and Computing. – 2025.
5. Indriani, Wiranata A. D. Comparison of Accuracy Levels of SVM, Decision Tree and Random Forest Algorithms in Sentiment Analysis of User Responses of the GoPay Application. Jurnal Teknik Informatika (JUTIF). – 2024.
6. Sayeed, Md Shohel & Roji, Varsha & Anbananthen, Kalaiarasi. (2023). BERT: A Review of Applications in Sentiment Analysis. HighTech and Innovation Journal. 4. 453-462. 10.28991/HIJ-2023-04-02-015.
7. Guo, Xinli. (2024). Sentimental analysis based on RoBERTa for Amazon review: an empirical study on decision making.
8. Sentiment Analysis of Movie Reviews Using BERT. arXiv preprint arXiv:2502.18841. – 2025.
9. Dhamma, Yonathan & Barus, Simon. (2025). Sentiment Analysis on Google Reviews Using NaÃ¯ve Bayes, K-Nearest Neighbors, and Logistic Regression to Improve Novotel ServicesSentiment Analysis on Google Reviews Using NaÃ¯ve Bayes, K-Nearest Neighbors, and Logistic Regression to Improve Novotel Services. Journal of Applied Informatics and Computing. 9. 106-114. 10.30871/jaic.v9i1.8923.

10. Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.Kim Y. Convolutional Neural Networks for Sentence Classification. EMNLP. – 2014.
11. Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv. – 2018.
12. Pennington, Jeffrey & Socher, Richard & Manning, Christopher. (2014). Glove: Global Vectors for Word Representation. EMNLP. 14. 1532-1543. 10.3115/v1/D14-1162.
13. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, Illia Polosukhin. Attention Is All You Need. NeurIPS. – 2017.
14. Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. arXiv. – 2013.
15. Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollar; Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2980-2988
16. Jason Weston, Sumit Chopra, Antoine Bordes. Memory Networks. arXiv preprint. – 2014.
17. Alex Graves, Greg Wayne, Ivo Danihelka. Matching Networks for One Shot Learning. NeurIPS. – 2016.
18. Jake Snell, Kevin Swersky, Richard Zemel. Prototypical Networks for Few-shot Learning. NeurIPS. – 2017.
19. Chollet F. Deep Learning with Python. Manning Publications. – 2017.
20. Brownlee J. Deep Learning for Natural Language Processing. Machine Learning Mastery. – 2018.

21. Jurafsky D., Martin J.H. Speech and Language Processing (3rd ed.). Draft online. – 2021.
22. Han J., Kamber M., Pei J. Data Mining: Concepts and Techniques (3rd ed.). Elsevier. – 2011.
23. Shmueli G. [та ін.]. Data Mining for Business Analytics. Wiley. – 2016.
24. Fawcett, Tom & Provost, Foster. (2013). Data Science for Business.
25. Microsoft Azure. Team Data Science Process Documentation. Microsoft Docs. – 2017.
26. IBM. ASUM-DM Methodology. IBM Knowledge Center. – 2020.
27. SAS Institute. SEMMA and the SAS Enterprise Miner. Technical Documentation. – 2003.
28. Pedregosa, Fabian & Varoquaux, Gael & Gramfort, Alexandre & Michel, Vincent & Thirion, Bertrand & Grisel, Olivier & Blondel, Mathieu & Prettenhofer, Peter & Weiss, Ron & Dubourg, Vincent & Vanderplas, Jake & Passos, Alexandre & Cournapeau, David & Brucher, Matthieu & Perrot, Matthieu & Duchesnay, Edouard & Louppe, Gilles. (2012). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 12.
29. Tjoa E., Guan C. A Survey on Explainable Artificial Intelligence (XAI). arXiv. – 2020.
30. Marco Tulio Ribeiro, Sameer Singh, Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. KDD. – 2016.
31. Kowsari K, Jafari Meimandi K, Heidarysafa M, Mendu S, Barnes L, Brown D. Text Classification Algorithms: A Survey. *Information*. 2019; 10(4):150. https://doi.org/10.3390/info10040150
32. Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

33. Mohammad, Saif & Turney, Peter. (2013). Crowdsourcing a Word-Emotion Association Lexicon. Computational Intelligence. 29. 10.1111/j.1467-8640.2012.00460.x.
34. Aniruddha Ghosh and Tony Veale. 2016. Fracking Sarcasm using Neural Network. In Proceedings of the 7th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pages 161–169, San Diego, California. Association for Computational Linguistics.
35. Zhang, Lei & Wang, Shuai & Liu, Bing. (2018). Deep Learning for Sentiment Analysis : A Survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 8. 10.1002/widm.1253.
36. Hassan A., Mahmood A. Deep Learning Techniques for Sentiment Analysis: A Survey. International Journal of Computer Applications. – 2017.

ДОДАТОК А

Код функції втрат Focal Loss

```
class FocalLoss(Loss):
    def __init__(self, gamma=2., alpha=0.25, **kwargs):
        super(FocalLoss, self).__init__(**kwargs)
        self.gamma = gamma
        self.alpha = alpha

    def call(self, y_true, y_pred):
        # Перетворення y_true в one-hot
        y_true = tf.one_hot(tf.cast(y_true, tf.int32), depth=y_pred.shape[-1])

        # Розрахунок фокусуючої втрати
        cross_entropy = tf.keras.losses.categorical_crossentropy(y_true, y_pred)
        probs = tf.reduce_sum(y_true * y_pred, axis=-1)
        focal_weight = self.alpha * tf.pow(1. - probs, self.gamma)
        loss = focal_weight * cross_entropy
        return loss
```

ДОДАТОК Б

Код імплементації навчання з розширеною пам’яттю

```
# Memory Augmented Learning

class MemoryAugmentedModel(tf.keras.Model):

    def __init__(self, vocab_size, embedding_dim, embedding_matrix, max_length, num_classes, memory_limit=50):

        super(MemoryAugmentedModel, self).__init__()

        self.embedding = Embedding(input_dim=vocab_size,

                                   output_dim=embedding_dim,

                                   input_length=max_length,

                                   weights=[embedding_matrix],

                                   trainable=True)

        self.lstm = LSTM(128, dropout=0.3, recurrent_dropout=0.3,

                         kernel_regularizer=regularizers.l2(0.001),

                         return_sequences=False, return_state=True)

        self.dense1 = Dense(64, activation='relu', kernel_regularizer=regularizers.l2(0.001))

        self.output_layer = Dense(6, activation='softmax')


        self.memory = {i: [] for i in range(num_classes)}  # Пам'ять для кожного класу


        self.memory_limit = memory_limit  # Обмеження кількості збережених у пам'яті векторів для кожного класу
```

```
def call(self, inputs, training=False):
    x = self.embedding(inputs)
    output, hidden_state, cell_state = self.lstm(x, training=training)
    x = self.dense1(output)
    logits = self.output_layer(x)
    return logits, output


def update_memory(self, hidden_states, labels):
    for h, label in zip(hidden_states, labels):
        label = label.numpy()
        # Додаю новий вектор до пам'яті
        if len(self.memory[label]) < self.memory_limit:
            self.memory[label].append(h)
        else:
            # Якщо пам'ять повна, заміняю найменш важливий вектор
            # Для простоти заміняю найстаріший вектор
            self.memory[label].pop(0)  # Видаляю старий
            self.memory[label].append(h)  # Додаю новий


def get_memory_center(self, label):
    if self.memory[label]:
```

```
            return np.mean(self.memory[label], axis=0)
        else:
            return np.zeros_like(self.memory[0][0]) if self.memory[0] else
np.zeros(self.lstm.units)
```

ДОДАТОК В

Код для завантаження векторного представлення слів GloVe

```
# Завантаження GloVe
def load_glove_vectors(glove_file_path):
    embeddings_index = {}
    with open(glove_file_path, 'r', encoding='utf-8') as f:
        for line in f:
            values = line.split()
            word = values[0]
            vector = np.asarray(values[1:], dtype='float32')
            embeddings_index[word] = vector
    return embeddings_index

glove_file_path = 'glove.6B.100d.txt'
embeddings_index = load_glove_vectors(glove_file_path)

embedding_dim = 100
embedding_matrix = np.zeros((len(tokenizer.word_index) + 1, embedding_dim))
for word, i in tokenizer.word_index.items():
    if word in embeddings_index:
        embedding_matrix[i] = embeddings_index[word]
```