ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка методики вилучення емотивної складової інформації з тексту повідомлень в соцмережах на основі рекурентних нейронних мереж»

на здобуття освітнього ступеня магістра
зі спеціальності 121 Інженерія програмного забезпечення
(код, найменування спеціальності)
освітньо-професійної програми «Інженерія програмного забезпечення»
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Микита БОНДАРЕНКО
(підпис)

Виконав: здобувач вищої освіти групи ПДМ-62
Микита БОНДАРЕНКО

Керівник: Оксана ЗОЛОТУХІНА
к.т.н., доцент

Рецензент: _____
Ім'я, ПРІЗВИЩЕ

Київ 2024

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Магістр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____ Ірина ЗАМРІЙ
«_____» _____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Бондаренку Микиті Олеговичу

1. Тема кваліфікаційної роботи: «Розробка методики вилучення емотивної складової інформації з тексту повідомлень в соцмережах на основі рекурентних нейронних мереж»
керівник кваліфікаційної роботи Оксана ЗОЛОТУХІНА к.т.н., доцент,

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «19» жовтня 2023 р. №145.

2. Строк подання кваліфікаційної роботи «29» грудня 2023р.

3. Вихідні дані до кваліфікаційної роботи: науково-технічна література з обробки природної мови, методи глибокого навчання.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Теоретичні аспекти аналізу емоцій у текстах
2. Розробка та налаштування алгоритму
3. Аналіз результатів та їх практичне впровадження

5. Перелік графічного матеріалу: *презентація*

1. Аналіз існуючих методів
2. Етапи передобробки даних
3. Вбудовування слів
4. Алгоритм вилучення емоційної складової інформації з текстів
5. Модифікована модель

6. Дата видачі завдання «19» жовтня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	19.10-05.11.23	
2	Аналіз існуючих підходів до виявлення емотивної складової інформації з тексту повідомлень в соціальних мережах	06.11-12.11.23	
3	Розробка алгоритму класифікації емотивної складової повідомлень за допомогою двонаправлених рекурентних нейронних мереж	13.11-03.12.23	
4	Реалізація системи вилучення емотивної складової інформації тексту повідомлень в соціальних мережах	04.12-10.12.23	
5	Оформлення роботи: вступ, висновки, реферат	11.12-20.12.23	
6	Розробка демонстраційних матеріалів	21.12-29.12.23	

Здобувач вищої освіти _____ Микита БОНДАРЕНКО
(підпис)

Керівник
кваліфікаційної роботи _____ Оксана ЗОЛОТУХІНА
(підпис)

РЕФЕРАТ

Текстова частина магістерської роботи на здобуття освітнього ступеня магістра: 73 стор., 2 табл., 19 рис., 29 джерел.

Мета роботи – оптимізувати процес вилучення емотивної складової інформації з текстових повідомлень в соціальних мережах шляхом застосування рекурентних нейронних мереж.

Об'єкт дослідження – процес вилучення емотивної складової інформації з текстових повідомлень в соціальних мережах.

Предмет дослідження – методи обробки текстових повідомлень для вилучення з них емотивної складової інформації.

Короткий зміст роботи: Проведено дослідження ефективності рекурентних нейронних мереж для аналізу текстів повідомлень і класифікації емоційних станів. Встановлено, що використання глибинних навчальних алгоритмів дозволяє точно визначати емоційну складову текстів.

Розроблено алгоритм для вилучення емотивної інформації з текстів, орієнтований на роботу в соціальних мережах. Проведено моделювання розробленого алгоритму, показавши його ефективність у різних контекстах.

На підставі отриманих результатів пропонується модифікований алгоритм для аналізу емотивної складової у текстових повідомленнях із соціальних мереж. Це дозволить більш ефективно визначати емоційні стани користувачів і може бути використане для покращення комунікаційних стратегій та рекламних кампаній. Отримані результати відкривають нові можливості для автоматизованого аналізу великих обсягів текстової інформації в соціальних мережах, забезпечуючи точне визначення емоційних складових і можуть бути ефективно впроваджені в різноманітних сферах використання.

КЛЮЧОВІ СЛОВА: ГЛИБИННЕ НАВЧАННЯ, ЕМОТИВНІ СКЛАДОВІ, АНАЛІЗ ТЕКСТУ, РЕКУРЕНТНІ НЕЙРОННІ МЕРЕЖІ, КЛАСИФІКАЦІЯ ЕМОЦІЙ, АВТОМАТИЗОВАНИЙ АНАЛІЗ ПОВІДОМЛЕНЬ, ОБРОБКА ПРИРОДНОЇ МОВИ.

ABSTRACT

Text part of the master's qualification work: 75 pages, 2 tables, 23 pictures, 26 sources.

The purpose of the work: to optimize the process of attracting emotional component information from text messages in social networks by using recurrent neural networks.

Object of research – the study is the process of attracting the emotional component of information from text messages in social networks.

Subject of research – methods of processing text messages to extract emotional component information from them.

Summary of the work: This comprehensive research initiative aims to advance the field of emotional information extraction from message texts, leveraging the capabilities of recurrent neural networks (RNNs). The primary objective is to formulate and explore an effective methodology tailored for the extraction of emotional content within message texts, particularly those originating from social networks.

A robust preprocessing pipeline was implemented to enhance the quality and relevance of the input data, ensuring that the neural network could effectively capture the emotional nuances within message texts. The preprocessing steps involved text normalization, including tasks such as lowercasing, stemming, and removal of stop words to streamline the textual data. Emphasis was placed on handling special characters, emojis, and other non-standard linguistic elements commonly found in social media messages.

The research's central focus lies in the application of recurrent neural networks for the meticulous analysis of message texts, with the ultimate goal of accurately determining and understanding the emotional states expressed in these communications. By delving into the intricacies of emotional nuances within textual data, this work seeks to make a substantial contribution to the refinement and enhancement of methods for extracting emotional insights from social media messages.

To further improve the model's efficiency, a vectorization algorithm was employed to convert the processed textual data into numerical representations. This transformation

facilitated the input of meaningful features into the recurrent neural network architecture, enabling the model to better comprehend the intricacies of emotional expressions in the messages.

The modified model, comprising a Bi-LSTM structure coupled with the preprocessed data and advanced vectorization algorithms, demonstrated notable improvements in its ability to accurately analyze and classify emotional states. The combination of meticulous preprocessing and state-of-the-art vectorization techniques enhanced the model's capacity to discern subtle emotional cues, resulting in a more refined and accurate representation of emotional content within social media messages.

KEYWORDS: DEEP LEARNING, EMOTIONAL COMPONENTS, TEXT ANALYSIS, RECURRENT NEURAL NETWORKS, EMOTION CLASSIFICATION, AUTOMATED MESSAGE ANALYSIS, NATURAL LANGUAGE PROCESSING.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

NLP — Natural Language Processing — Обробка природної мови.

CNN — Convolutional Neural Network — Конволюційна нейронна мережа.

RNN — Recurrent Neural Network — Рекурентна нейронна мережа (РНМ).

TF-IDF — Term Frequency-Inverse Document Frequency — Частота термінів-інверсна частота документів.

URL — Uniform Resource Locator — Уніфікований локатор ресурсу.

PICKLE — Формат файлу для серіалізації об'єктів у Python.

LSTM — Long Short-Term Memory — Довга короткотривала пам’ять (тип РНМ).

SVM — Support Vector Machine — Машина опорних векторів.

BERT — Bidirectional Encoder Representations from Transformers — Двонаправлене представлення енкодерів з трансформаторів.

GPT — Generative Pre-trained Transformer — Генеративний попередньо навчений трансформер.

SVC — Support Vector Classifier — Класифікатор на основі методу опорних векторів.

SGD — Stochastic Gradient Descent — Стохастичний градієнтний спуск.

ВСТУП

Сучасний світ цифрових технологій і масової комунікації відкриває нові горизонти в аналізі та обробці інформації. Особливу увагу привертає аналіз текстів у соціальних мережах, де велика кількість даних може дати глибоке розуміння людських емоцій та поведінки. Актуальність цієї теми полягає у потребі розробки ефективних інструментів для вилучення емоційної складової із великих обсягів текстових даних. Застосування таких інструментів може мати велике значення у сферах маркетингу, психології, соціології та інших.

У маркетингу, аналіз емоцій у соціальних мережах допомагає брендам зрозуміти сприйняття їхніх продуктів та ефективніше планувати кампанії. У психології, це може виявити симптоми психічних розладів, таких як депресія. У соціології, аналіз думок та настроїв у соцмережах виявляє суспільні тенденції та реакції на актуальні події. У соціології, аналіз емоційних проявів на соціальних мережах може допомогти вивчити думки та настрої груп людей [1]. Це дає змогу встановити тенденції та зрозуміти реакцію суспільства на події, які відбуваються в реальному часі.

Загалом, аналіз емоційних реакцій у соціальних мережах відкриває безліч можливостей для розуміння та використання інформації. Розвиток ефективних інструментів для вилучення емоційної складової з великих обсягів текстових даних є актуальним завданням, яке відкриває нові перспективи у багатьох галузях знань.

Мета даної роботи оптимізувати процес вилучення емотивної складової інформації з текстових повідомлень в соціальних мережах шляхом застосування рекурентних нейронних мереж.

Для досягнення цієї мети передбачається:

1. Проаналізувати існуючі методи та особливості обробки текстових повідомлень у соціальних мережах. Провести огляд та розбір літератури пов’язаної зі штучними нейронними мережами;
2. Визначити методи передобробки для подальшого вилучення емотивної складової повідомлень з соцмереж;
3. Розробити алгоритм, що базується на рекурентних нейронних мережах

для вилучення емоцій з текстів повідомлень;

4. Експериментально перевірити ефективність запропонованої методики в порівнянні з традиційними методиками та глибокого навчання;

Об'єкт дослідження – процес вилучення емотивної складової інформації з текстових повідомлень в соціальних мережах.

Предмет дослідження – методи обробки текстових повідомлень для вилучення з них емотивної складової інформації.

Методи дослідження охоплюють аналіз даних, машинне навчання, роботу з рекурентними нейронними мережами, та методи обробки природної мови.

Джерела дослідження включають наукові статті та праці, що висвітлюють аспекти обробки текстових даних, аналіз емоцій, а також специфіку рекурентних нейронних мереж.

Наукова новизна роботи: розроблено методику для вилучення емотивної складової з текстів повідомлень у соцмережах. Особливістю цієї методики є здатність адаптуватися до специфіки емоційного контенту в різних соціальних мережах та покращення здатності моделі враховувати контекст специфічних платформ, що відкриває нові можливості для глибшого аналізу поведінкових патернів користувачів.

Практичне значення роботи відображається у можливості використання розробленої методики в маркетингових дослідженнях, аналізі соціальних мереж, психологічних дослідженнях, та інших галузях, де важливо розуміти емоційний стан користувачів.

Апробація результатів магістерської роботи, робота пройшла апробацію на Всеукраїнській науково-практичній конференції «ТАК» в ДВНЗ «Донецький національний технічний університет» (5-6 грудня 2023 року, м. Луцьк), Всеукраїнська конференція молодих вчених «Актуальні питання розвитку інформаційних технологій» в ДВНЗ «Приазовський державний технічний університет» (22 листопада 2023 року , м. Дніпро).

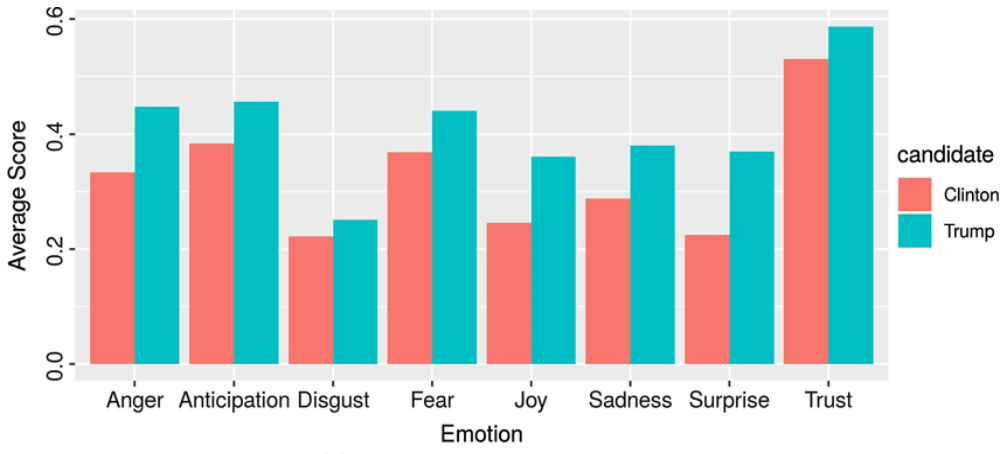
1 ТЕОРЕТИЧНІ АСПЕКТИ АНАЛІЗУ ЕМОЦІЙ У ТЕКСТАХ

1.1 Стан досліджень в області аналізу текстових даних соціальних мереж

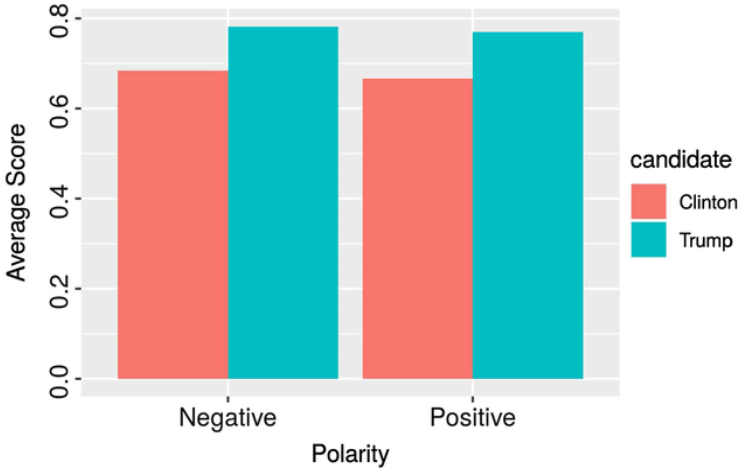
Аналіз емоцій, або аналіз настроїв, у цифровому контексті соціальних медіа-платформ виконує важливу роль у вивченні і розумінні людської поведінки, сприяючи виявленню, категоризації та кількісній оцінці афективних станів та суб'єктивної інформації, що виражаються у текстових даних. Процес аналізу емоцій починається з ідентифікації та розпізнавання емоційних сигналів в тексті, таких як позитивні, негативні чи нейтральні відтінки. Це може включати в себе виявлення конкретних емоцій, таких як радість, сум, злість, страх тощо, або оцінку загального емоційного настрою тексту. Для досягнення цього використовуються методи машинного навчання, природно-мовні технології та статистичні підходи.

Одним з основних застосувань аналізу емоцій у соціальних медіа-платформах є визначення настрою користувачів щодо певних продуктів, послуг, подій або тематичних речей [2]. Це дозволяє компаніям і брендам отримувати зворотний зв'язок та відгуки в реальному часі, а також оцінювати емоційну реакцію своєї аудиторії на різні ініціативи. Наприклад, аналізуючи емоційну реакцію користувачів на рекламні кампанії чи новий продукт, компанії можуть швидко реагувати та вносити зміни, що покращує їхню стратегію маркетингу. Крім того, аналіз емоцій може бути використаний для дослідження громадської думки і виявлення трендів у суспільстві. При вивченні соціальних мереж, аналіз емоцій допомагає визначити загальну тенденцію серед користувачів щодо певних подій або соціальних питань. Це може бути корисно для політичного аналізу, прогнозування громадських настроїв і моніторингу реакцій на новини. Прикладом реального застосування аналізу емоцій у соціальних мережах є вибори 2016 року в Сполучених Штатах Америки. Аналітики та дослідники активно використовували дані з Twitter та Facebook для вимірювання громадського настрою та популярності кандидатів [3]. Протягом виборчої кампанії, команди аналітиків застосовували

алгоритми аналізу емоцій для того, щоб проаналізувати великі обсяги постів і твітів, що стосувалися Дональда Трампа та Хілларі Клінтон. Зокрема, вони оцінювали такі параметри, як позитивні та негативні емоції, що пов'язані з кожним кандидатом (рис.1.1), а також рівень активності та взаємодії з їхніми повідомленнями. Аналіз виявив, що в певні періоди кампанії, наприклад під час дебатів або після важливих виборчих подій, тон коментарів у соціальних мережах різко змінювався, що вказувало на зміни в громадському сприйнятті кандидатів. Такий аналіз допомагав передбачити регіони з високим рівнем підтримки або невдоволення кандидатами, що могло бути корисним для стратегічного планування виборчих кампаній.



(a) Emotions of Extracted tweets



(b) Polarity of Extracted tweets

Рис. 1.1 Вимірювання емоції які викликали Трамп та Клінтон [3]

Сучасні технології аналізу емоцій значно розвинулися завдяки прогресу в

обробці природної мови (NLP) та машинному навчанню. Сучасні підходи переважно використовують складні алгоритми, такі як глибоке навчання і нейронні мережі, для розпізнавання та інтерпретації емоційного тону текстового контенту. Ці методи продемонстрували значний успіх не лише у виявленні основних настроїв, таких як позитивні, негативні чи нейтральні, але й у визначенні цілої низки емоцій, таких як щастя, гнів, здивування чи смуток. Розвиток технологій, які базуються на штучному інтелекті, дозволив аналізу емоцій перейти на новий рівень складності та точності [4]. Один з яскравих прикладів використання NLP та штучного інтелекту в аналізі емоцій демонструється у системах обслуговування клієнтів і чат-ботах. Сучасні чат-боти та віртуальні асистенти, які використовуються компаніями для спілкування з клієнтами, здатні не лише розуміти запити на основі текстових команд, але й аналізувати емоційний контекст повідомлень. Це дозволяє їм відповідати на запити користувачів більш ефективно та емоційно адекватно. Наприклад, у сфері обслуговування клієнтів, чат-бот може аналізувати текст запиту клієнта та визначати, чи відчуває клієнт розчарування, задоволення, або навіть злість. На основі цього аналізу, чат-бот може адаптувати свою відповідь, щоб бути більш емпатичним або надати більш детальну допомогу, якщо виявлено фрустрацію або плутанину. Це значно покращує якість обслуговування та допомагає збільшити задоволеність клієнтів.

Використання більш глибоких та комплексних моделей, таких як конволюційні нейронні мережі (CNN) та рекурентні нейронні мережі (RNN), забезпечило здатність аналізувати більш тонкі емоційні нюанси в тексті. Це включає здатність враховувати контекстуальні зміни та ідіоматичні вирази, що значно покращує точність інтерпретації емоційних станів. Наприклад CNN часто застосовують для класифікації текстів, у виявленні спаму в електронних листах. Тут CNN використовується для виявлення певних шаблонів у словах або фразах, які часто зустрічаються у спам-повідомленнях [5]. Що до RNN то такі мережі ефективно використовуються в системах машинного перекладу, де важливо зберегти контекстуальний зв’язок усього речення. RNN здатні враховувати

попередні слова у реченні для кращого контекстного розуміння. Їх також застосовуються у генерації тексту, наприклад, у створенні новинних статей або літературних творів. Вони можуть продовжувати заданий текст, зберігаючи смислову та стилістичну цілісність [6].

Аналіз емоцій є важливим компонентом аналізу текстових даних у соціальних мережах, що дозволяє зрозуміти настрої, вподобання та поведінку користувачів. Стан досліджень у цій галузі свідчить про значний прогрес у застосуванні різних підходів і методів. Однак залишаються проблеми, пов’язані з контекстним розумінням, багатомовним аналізом, адаптивністю до конкретної галузі та суб’єктивністю.

1.2 Методи вилучення емотивної інформації

Вилучення емоційної інформації полягає не лише у виявленні явного сентименту тексту, але й у розумінні складних шарів людської експресії, включаючи сарказм, радість, незадоволення і навіть ворожість. Методи, що використовуються для вирішення цього завдання, різноманітні і складні, поєднуючи в собі досягнення машинного навчання, обробки природної мови і комп’ютерної лінгвістики. Шлях до вилучення емоційної інформації починається з аналізу настроїв, який класифікує текст на основні настрої, такі як позитивні, негативні або нейтральні. Ця форма аналізу є фундаментальною для розуміння загального тону і настрою, переданого в тексті. Однак людські емоції та висловлювання набагато складніші та різноманітніші, ніж ці базові категорії. Тому для того, щоб охопити весь спектр людських емоцій, потрібні більш досконалі методи. Однією з таких складних категорій є сарказм. Виявлення сарказму в тексті передбачає не лише буквальне тлумачення слів, але й розуміння контексту, тону, а часто й культурних нюансів. Виявлення сарказму є унікальним викликом, оскільки вимагає від алгоритму розуміння значення, протилежного до того, що прямо вказано. Це вимагає складних технік, які можуть інтерпретувати контекст і виводити передбачуване значення, що стоїть за словами. Іншим важливим

аспектом є виявлення мови ворожнечі або агресивних висловлювань. В епоху соціальних мереж, коли онлайн-взаємодія іноді може набувати ворожого характеру, вкрай важливо розробити ефективні методи виявлення та позначення такого шкідливого контенту. Це передбачає розмежування між просто образливими висловлюваннями та мовою ненависті, що має важливе значення для модерації онлайн-платформ і підтримки здорового комунікаційного середовища.

Вилучення емотивної інформації, також відоме як аналіз сентименту, використовує різноманітні методи для розпізнавання та класифікації емоційних станів, виражених у тексті. Словники сентиментів містять списки слів разом із позначеними емоційними значеннями, такими як позитивний, негативний або нейтральний. Кожне слово або фраза у словнику має певний «сентиментний бал», який відображає його емоційну вагу (рис.1.2).

reviewid	content	Word	Sentiment
22745	accountable	accountable	-0.5
22725	ability	ability	0.5
22725	able	able	0.5
22725	able	able	0.5
22724	able	able	0.5
22724	actually	actually	1
22721	accomplish	accomplish	0.5
22720	actually	actually	1
22719	absurdly	absurdly	-1
22718	abrasive	abrasive	-1
22718	actual	actual	1
22718	actual	actual	1
22718	actual	actual	1
22718	actual	actual	1
22715	actually	actually	1
22714	actual	actual	1
22713	absorbed	absorbed	0.5

Рис. 1.2 Приклад словника сентиментів

Ці словники можуть бути створені експертами у галузі мови або згенеровані

автоматично за допомогою алгоритмів машинного навчання. Текст аналізується на наявність слів зі словника сентиментів. Кожне виявлене слово вносить свій вклад у загальний емоційний відтінок тексту. Загальний сентимент тексту визначається шляхом підсумовування сентиментних балів всіх виявлених слів. Наприклад, текст з більшою кількістю позитивно забарвлених слів отримає позитивний загальний сентимент [7]. На основі загальної суми сентиментних балів, текст класифікується як позитивний, негативний або нейтральний.

Хоча лексичний підхід може бути ефективним для прямого та явного вираження емоцій, він може не враховувати більш тонкі аспекти мови, такі як сарказм, контекст або полісемію (слова з декількома значеннями). Лексичний підхід часто використовується через свою простоту та прямолінійність, але він може бути обмежений у здатності адекватно інтерпретувати складні та неоднозначні вирази емоцій.

Машинне навчання надає потужні інструменти для аналізу сентиментів, дозволяючи розробляти моделі, які можуть класифікувати текстові дані з високою точністю. Для класифікації тексту часто використовуються такі алгоритми, як наївний Баєсів класифікатор, логістична регресія, та опорні векторні машини (SVM). Ці алгоритми навчаються на попередньо визначених даних, де кожен приклад тексту має позначку сентименту (наприклад, позитивний, негативний або нейтральний). Важливим етапом є підготовка та очищення даних, включаючи видалення шуму, токенізацію, та перетворення тексту в числові вектори, які можуть бути оброблені алгоритмами машинного навчання. Глибинне навчання використовує складніші моделі, такі як згорткові нейронні мережі (CNN) та рекурентні нейронні мережі (RNN), включаючи LSTM. Ці моделі можуть розпізнавати та інтерпретувати складні патерни в текстових даних. Особливо, LSTM ефективні у вирішенні проблем з короткочасною пам’яттю RNN, дозволяючи моделям зберігати інформацію протягом довших періодів часу, що є критично важливим для розуміння контексту в тексті. Методи машинного та глибинного навчання в аналізі сентиментів забезпечують високу точність і

гнучкість, дозволяючи адаптувати моделі до різних даних та контекстів. Вони можуть ефективно обробляти великі обсяги даних, виявляти тонкі емоційні нюанси та навіть адаптуватися до специфічних особливостей мови або жанру.

Гібридні методи об’єднують елементи лексичних підходів та алгоритмів машинного навчання, надаючи збалансований та всебічний інструмент для аналізу сентиментів. Ці методи використовуються для підвищення точності та надійності в аналізі емоційних відгуків у текстах. Використання словників сентиментів для оцінки емоційного забарвлення окремих слів або фраз. Це дозволяє швидко ідентифікувати загальний тон тексту на основі виразно емоційно забарвлених слів. Доповнення лексичного аналізу за допомогою алгоритмів машинного навчання, таких як SVM, наївний Баєсів класифікатор або нейронні мережі. Ці моделі можуть враховувати більш складні патерни в тексті, які важко визначити за допомогою тільки лексичного підходу. Аналіз структури тексту, такий як синтаксис та граматика, для кращого розуміння його значення. Врахування контексту, в якому вживаються слова, включаючи культурні та ситуативні фактори, які можуть впливати на інтерпретацію емоцій. Гібридні методи дозволяють ефективно поєднувати краще з обох світів, точність та швидкість лексичних методів з глибиною та контекстуальною здатністю машинного навчання. Це забезпечує більш точний та надійний аналіз сентиментів, здатний адаптуватися до різноманітних стилів та контекстів комунікації.

Переходячи від загальної картини викликів, які стоять перед сучасним аналізом тексту у соціальних мережах, необхідно зосередитись на розгляді ключових методів, які дозволяють глибше зрозуміти емоційну складову текстових даних. Це знання є вирішальним для формування власного алгоритму, здатного ефективно і точно інтерпретувати емоційні вияви в комунікаціях. Розуміння основних принципів, які лежать в основі цих методів, допоможе в розробці більш чутливих і адаптивних систем аналізу. В різних проектах пов’язаних з «Twitter Sentiment» [8], відображаються різні аспекти та підходи до аналізу емоцій у тексті. Дослідження цього методу надає можливість зрозуміти, як різні техніки можуть

бути застосовані для різних типів емоційних виразів. Емоції, які досліджуються в проекті зосереджується на визначенні основних настроїв, таких як позитивний, негативний та нейтральний. Це становить основу для розуміння загального емоційного тону в соціальних мережах. Аналіз методу дає змогу не лише виявити його сильні та слабкі сторони, але й розуміти, як різні підходи до обробки природної мови та машинного навчання можуть бути застосовані для розпізнавання різноманітних емоційних виразів. Sentiment Analysis є демонстрацією використання методів машинного навчання та NLP для аналізу емоцій у твітах. Проект починається з імпортування набору даних Sentiment140, який містить 1,600,000 твітів, витягнутих через Twitter API. Дані анотовані як негативні (0) та позитивні (4), і проект використовує лише поля sentiment (емоційний тон) та text (текст твіта).

Передобробка даних є критичним кроком, який включає:

1. Перетворення тексту на нижній регістр для забезпечення уніформності;
2. Заміна URL-адрес та емодзі для очищення тексту;
3. Видалення неалфавітних символів та повторюваних літер для зменшення шуму в даних;
4. Видалення коротких слів та стоп-слів для підвищення релевантності тексту;
5. Лематизація слів для зведення слів до їх базової форми.

Для кращого розуміння оброблених даних проводиться аналіз, включаючи візуалізацію за допомогою хмари слів (рис. 1.3). Це дозволяє ідентифікувати найчастіше вживані слова в позитивних та негативних твітах.

Далі, дані розділяються на тренувальні дані (95%) використовуються для тренування моделі, тестові дані (5%) використовуються для перевірки ефективності моделі. Тренувальні та тестові дані трансформуються в матрицю TF-IDF ознак за допомогою TF-IDF векторизатора, що допомагає визначити важливість слів у контексті даних. Проект включає розробку трьох моделей для аналізу емоцій:

1. Bernoulli Naive Bayes (BernoulliNB);
2. Linear Support Vector Classification (LinearSVC);
3. Logistic Regression (LR).

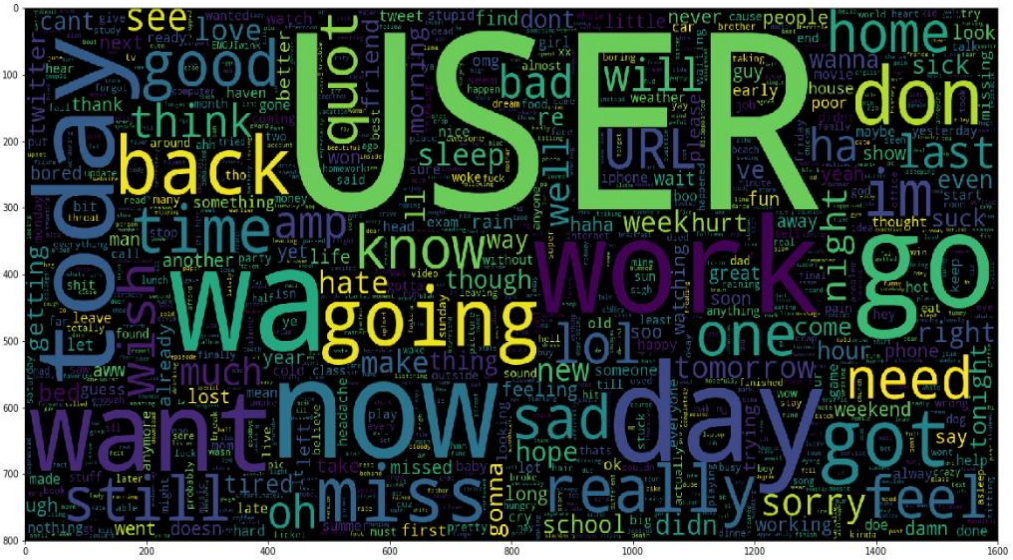


Рис. 1.3 Хмара негативних твітів [9]

Моделі оцінюються на основі точності, а також аналізуються за допомогою матриці помилок. Logistic Regression Model показала найвищу точність майже 82%, тоді як BernoulliNB Model досягла 80% точності, будучи при цьому найшвидшою для тренування та передбачення. Моделі та векторизатор зберігаються за допомогою PICKLE [9] для подальшого використання. Для використання моделі потрібно імпортувати векторизатор та модель LR через PICKLE. Передбачуваний текст спочатку потребує перед-обробки, а потім векторизації, після чого можна використовувати модель для аналізу емоцій.

Хоча проект [9] є хорошим вступом до аналізу емоцій у соціальних мережах, він може мати деякі недоліки. Модель зосереджується лише на базових емоціях (позитивних, негативних, нейтральних) і може не враховувати більш складні емоційні стани. Може мати труднощі з розпізнаванням іронії, сарказму або діалектних особливостей мови. Підхід, представлений у проекті, може бути не настільки ефективним для специфічних або незвичайних наборів даних, де

потрібен більш індивідуалізований підхід.

Розгорнутий аналіз настроїв в твітах, що використовує методи обробки природної мови та машинного навчання [10], є більш глибоким дослідженням у порівнянні з попереднім проектом. Використовується набір даних, який містить твіти з різними емоційними забарвленнями. Твіти класифіковані як позитивні, негативні чи нейтральні. Використовуються стандартні процедури передобробки, такі як очищення тексту від спеціальних символів, URL-адрес, згадувань користувачів тощо. Застосування лематизації та видалення стоп-слів для зменшення шуму та підвищення якості даних. Проект [10] використовує більш різноманітні та складні моделі, такі як логістична регресія, в порівнянні з більш базовими моделями в проекті для початківців. Також проект включає більш глибокий аналіз результатів, включаючи матриці помилок та інші метрики оцінки моделей.

Дослідження у сфері класифікації тексту проводиться за шістьма типами емоцій: радість, смуток, страх, гнів, любов, здивування (рис. 1.4).

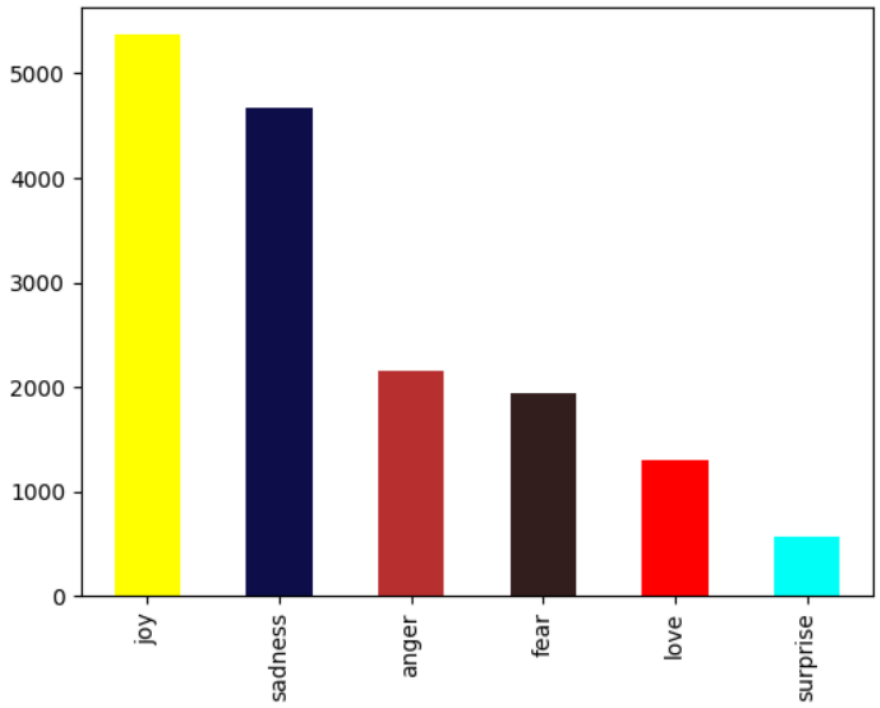


Рис. 1.4 Типи емоцій в проекті [10]

Проєкт використовує LSTM мережі, які дозволяють моделі аналізувати послідовності слів у тексті та розпізнавати емоційний зміст.

Один із ключових етапів у проекті це підготовка даних. Завантаження тренувального, перевірочного та тестового наборів даних є першим кроком. Після цього процес обробки даних включає наступні кроки:

1. Очищення даних: Перевірка на наявність відсутніх значень та їх обробка;
2. Токенізація: Процес перетворення тексту в послідовності токенів (слів), де кожному слову присвоюється унікальний ідентифікатор;
3. Стемінг: Скорочення слів до їх кореневої форми за допомогою PorterStemmer, що допомагає зменшити розмір словника;
4. Додавання падінгу: Уніфікація довжини всіх послідовностей до однакового розміру шляхом додавання нульових значень.

Модель складається з Embedding, Bidirectional LSTM і Dense шарів:

- Embedding Layer: Цей шар у моделі глибокого навчання використовується для перетворення словникових індексів слова на вектори реальних чисел. Кожне слово представляється як вектор у високорозмірному просторі, де слова з подібним значенням знаходяться ближче одне до одного;
- Bidirectional LSTM (Bi-LSTM): Це розширення стандартної LSTM мережі, яка додатково проходить інформацію в обох напрямках (з початку до кінця та навпаки). Це дозволяє моделі зберігати контекст з обох кінців послідовності, що покращує здатність моделі розуміти контекст та залежності у тексті;
- Dense Layers: Це повністю з’єднані нейронні шари, де кожен нейрон у шарі з'єднаний з усіма нейронами попереднього шару. Ці шари використовуються для класифікації або регресії на основі ознак, витягнутих попередніми шарами мережі.

Навчання ж моделі відбувається за допомогою Adam [12]. Це алгоритм оптимізації, який може динамічно налаштовувати швидкість навчання моделі. Adam поєднує переваги двох інших розширень градієнтного спуску: AdaGrad і

RMSProp. Після навчання модель проходить тестування і результати оцінюються за допомогою матриці помилок.

Аналіз попередньо описаних проектів з платформи Kaggle виявив, що хоча всі три проекти спрямовані на аналіз емотивної складової тексту, їх підходи відрізняються. Перший проект фокусується на базовому аналізі настроїв в Twitter за допомогою традиційних методів NLP, другий використовує більш складні методи для глибшого аналізу, в той час як третій застосовує LSTM мережі для розпізнавання широкого спектру емоцій. Цей аналіз надав інформацію щодо структури та можливих варіантів вдосконалення окремих функціональних частин алгоритмів у проектах.

1.3 Аналіз проблемних аспектів в методах аналізу емоцій

Аналіз емоцій в текстах соціальних мереж є складною задачею, яка вимагає глибокого розуміння не тільки мови, але й контексту, в якому вона використовується. Різноманітність соціальних медіа і нюанси людського спілкування створюють виклики для точного визначення емоцій. У вище описаних проектах не дивлячись на відносно хороші результати існує ряд ключових проблем які можна вирішити. Вирішення цих проблем може підвищити ефективність використаних інструментів та покращити загальні результати. Необхідно детальніше розібрати конкретні проблеми визначені у раніше описаних проектах.

Використання простих моделей у аналізі сентименту часто може бути обмеженим через відсутність розуміння контексту. Прості алгоритми, які покладаються на відповідність ключових слів або базові лінгвістичні правила, можуть не вловлювати складність людської мови, що призводить до неточної класифікації емоцій. Ця проблема особливо важлива у соціальних мережах, де використання іронії, сарказму та сленгу є розповсюдженим.

Для поліпшення точності класифікації емоцій можна використовувати більш просунуті методи машинного навчання, такі як машини опорних векторів (SVM).

SVM можуть ефективно розділяти дані в багатовимірному просторі, виявляючи складні закономірності, які не можуть бути виявлені простішими моделями [13]. Застосування моделей глибокого навчання, таких як рекурентні нейронні мережі (RNN), може допомогти в розумінні послідовностей та контексту в тексті. RNN здатні обробляти інформацію у вигляді послідовностей, що дозволяє їм враховувати попередній контекст у визначенні сентименту [14]. Це робить їх особливо ефективними для аналізу сентименту в соціальних медіа. Трансформаторні моделі, такі як BERT і GPT [15-16], представляють наступний крок у розвитку технологій обробки природної мови. Використовуючи попередньо навчені великі нейронні мережі, ці моделі можуть глибоко розуміти семантичні та контекстуальні нюанси мови, що дозволяє їм точніше визначати емоційний відтінок тексту. Такі моделі стають дедалі важливішими у застосуванні аналізу сентименту, здатні вловлювати тонкі відтінки мови, такі як іронія та сарказм, що раніше було складно виявити.

Проблема подібних проектів «Twitter Sentiment Analysis» з надмірним пристосуванням моделей, таких як LSTM, є важливим аспектом, що вимагає уваги. Перенавчання відбувається, коли модель стає надто "звиклою" до навчальних даних, втрачаючи здатність ефективно узагальнювати нові дані. Це може призводити до погіршення продуктивності моделі у реальних умовах використання. Рання зупинка є важливою технікою вирішення цієї проблеми. Зупиняючи тренування моделі, коли не спостерігається подальшого поліпшення на валідаційному наборі даних, можна запобігти перенавчанню. Це дозволяє моделі зберегти свою здатність до узагальнення, не втрачаючи точності на навчальному наборі.

Ансамблеві методи, такі як «Bagging» та «Boosting», також є ефективними у вирішенні проблеми перенавчання. Вони дозволяють комбінувати кілька моделей для підвищення загальної стабільності та точності. «Bagging» створює різні підмножини даних із тренувального набору та тренує окремі моделі на кожній з них, тоді як «Boosting» послідовно додає нові моделі, кожна з яких виправляє

помилки попередніх [17]. Ці підходи допомагають зменшити варіативність та упередження моделі, що призводить до більш точних прогнозів. Оновлення навчальних даних для відображення найновіших тенденцій та мовних використань у соціальних мережах є ще одним ключовим аспектом. Це дозволяє моделі залишатися актуальною та адаптуватися до постійно змінюваних мовних патернів, що особливо важливо в динамічному світі соціальних медіа.

В цілому, використання цих стратегій може значно підвищити ефективність аналізу, поліпшуючи їхню здатність точно класифікувати емоційні стани користувачів на основі текстових даних.

Перенавчання може стати значною перешкодою. Через складність LSTM-моделей, вони можуть стати надмірно специфічними до навчального набору даних, що призводить до зниження їх здатності до узагальнення на нових, різноманітних даних. Це може бути особливо проблематично у задачах розпізнавання емоцій, де важливо, щоб модель могла адекватно інтерпретувати широкий спектр емоційних виразів. Для боротьби з перенавчанням, методи регуляризації, такі як відсіювання (Dropout) і рання зупинка, є ефективними інструментами. Відсіювання допомагає запобігти занадто тісному "приляганню" моделі до тренувальних даних шляхом випадкового вимкнення деяких нейронів під час процесу навчання. Рання зупинка полягає у припиненні тренування, коли модель починає показувати ознаки перенавчання на перевірочному наборі даних.

Навчання з перенесенням та попереднє навчання на великих та різноманітних наборах даних, пов'язаних з емоціями, можуть бути використані для покращення здатності моделі уловлювати більш тонкі та різноманітні емоційні репрезентації. Ці методи дозволяють моделі отримати більш широке уявлення про різні емоційні стани та реакції, що підвищує її точність та універсальність.

Точне налаштування моделі для специфічного завдання та набору даних є критично важливим. Це включає оптимізацію гіперпараметрів, таких як швидкість навчання, кількість епох, розмір пакету, та архітектуру моделі. Правильне налаштування може значно підвищити продуктивність моделі, забезпечуючи

краще розуміння емоційних нюансів у даних. Використання цих передових підходів та технік дозволяє створити потужні та надійні інструменти для аналізу емоцій, що мають велике значення у різних областях застосування, від соціальних медіа до психологічних досліджень.

2 РОЗРОБКА ТА НАЛАШТУВАННЯ АЛГОРИТМУ

2.1 Методологічні підходи до дослідження

При проведенні будь-якого дослідження вибір інструментів та методів має вирішальне значення для ефективного досягнення цілей дослідження. Цей розділ має на меті окреслити та обґрунтувати вибір ключових інструментів та методів, які будуть використані в цьому дослідженні, зосередившись на порівнянні та виборі моделі. В рамках дослідження було обрано такі моделі:

LSTM [11] тип рекурентної нейронної мережі, який вразив своєю здатністю аналізувати та розуміти довгострокові залежності в послідовностях даних, включаючи текстову інформацію. Він є потужним інструментом для роботи з обробкою природної мови та виявився дуже корисним у багатьох задачах, пов'язаних з аналізом настроїв та емоцій. Однією з важливих особливостей LSTM є його здатність зберігати контекстну інформацію та ефективно працювати з довгостроковими залежностями у тексті. Це дозволяє моделі враховувати не лише найближчий контекст, але й інформацію, яка може бути важливою через декілька речень або абзаців. Ця здатність особливо корисна при аналізі емоційних нюансів у текстах, де часові залежності та контекст грають ключову роль.

Наївний класифікатор Байєса [19] імовірнісна модель, яка базується на теоремі Байєса та припущенні про наївну (умовну) незалежність між ознаками (словами) у даних при врахуванні мітки класу (емоції). Цей класифікатор отримав свою назву «наївний» через своє спрощене припущення про незалежність ознак, що, в реальності, може не завжди виконуватися, але незважаючи на це, він досить ефективний та широко використовується в аналізі настроїв та класифікації текстів.

Основним принципом роботи класифікатора Байєса є визначення ймовірностей належності тексту до певного класу (емоції). Він використовує теорему Байєса для обчислення цих ймовірностей на основі статистики, отриманої з навчальних даних. Згідно з цією теоремою, ймовірність належності тексту до класу обернено пропорційна ймовірності того, що цей текст містить дані ознаки

при умові, що він належить до даного класу. Основне припущення «наївного» класифікатора Байєса полягає в тому, що він вважає всі ознаки (слова) у тексті умовно незалежними один від одного, тобто ймовірність наявності кожного слова в тексті не залежить від наявності інших слів. Це спрощення допомагає в обчисленні ймовірностей, але, як вже зазначалося, не завжди відповідає реальності.

Незважаючи на це спрощення, наївний класифікатор Байєса часто демонструє гарні результати в багатьох проектах з аналізу настроїв та класифікації текстів, особливо коли набір ознак (слів) досить великий. Він є ефективним і швидким методом, що допомагає вирішити багато завдань в області обробки природної мови;

Логістична регресія [12] це статистична модель, яка використовується для прогнозування ймовірності бінарного результату на основі вхідних даних. У випадку аналізу емоцій, ця модель може бути дуже корисною для визначення ймовірності належності тексту до певного емоційного класу, наприклад, «позитивний», «нейтральний» або «негативний». Основна ідея логістичної регресії полягає в тому, що вона використовує лінійну комбінацію вхідних ознак і обчислює логарифм шансів (логіт) належності до певного класу. Цей логіт потім перетворюється в ймовірність за допомогою логістичної функції. Тобто, логістична регресія допомагає визначити, наскільки ймовірно, що даний текст належить до конкретного класу емоцій. Логістична регресія є простою та зрозумілою моделлю, і саме через це вона успішно використовується в багатьох дослідницьких проектах, включаючи аналіз емоцій у текстах. Вона має важливі переваги, такі як ефективність та швидкість роботи, і може бути використана як базова модель для аналізу настроїв та емоцій в текстах. Більш складні моделі можуть використовувати логістичну регресію як частину більшого аналітичного пайплайну для отримання більш точних результатів.

Linear SVC [13] потужний алгоритм для класифікації даних, і він є особливо ефективним у високорозмірних просторах, включаючи текстові дані. Головна ідея Linear SVC полягає в тому, що він намагається знайти таку гіперплощину, яка

максимально відокремлює класи даних, тобто максимізує відстань між гіперплощиною і найближчими точками об’єктів кожного класу (це називається «зазором»). Знаходження цієї оптимальної гіперплощини дозволяє Linear SVC добре справлятися з класифікацією, навіть у високорозмірних просторах, які часто виникають у текстовому аналізі. Linear SVC відрізняється від ядрових методів SVC тим, що він працює з лінійною гіперплощиною, що розділяє дані, в той час як ядрові методи дозволяють використовувати нелінійні функції ядра для розділення даних в більш складних формах. Linear SVC може бути особливо корисним у випадках, коли кількість ознак (слів у тексті) значно перевищує кількість вибірок, і він зазвичай працює швидше, ніж ядрові методи, завдяки своїй лінійній природі.

Необхідно проаналізувати кожен з варіантів, щоб зрозуміти який найкраще підходить для реалізації задач роботи. Розглянемо структуру моделі LSTM (рис. 2.1).

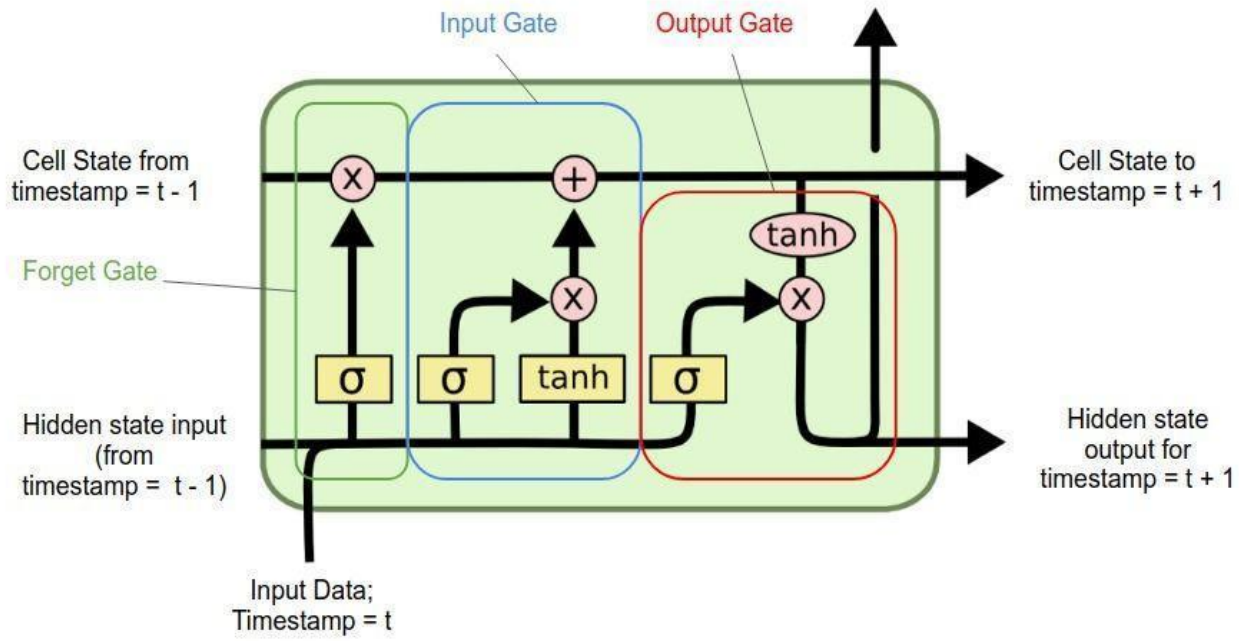


Рис. 2.1 Модель LSTM [28]

LSTM має такі компоненти, як стан комірки та різні вентилі. Стан комірки проходить прямо по діаграмі з деякими лінійними взаємодіями, зберігаючи

інформацію в часі. Вентилі контролюють потік інформації в стан комірки і з нього, на який впливає прихований стан і поточний вхід. Вентиль забування вирішує, яку інформацію видалити зі стану комірки, вентиль входу вирішує, яку нову інформацію додати, а вентиль виходу вирішує, яким має бути наступний прихований стан. Для регулювання потоку інформації в цих воротах використовуються сигмоїдальні (σ) і тангенціальні функції [18]. Запам’ятовувальний шлюз (Forget Gate) у LSTM моделях відіграє ключову роль у визначенні, яку інформацію необхідно видалити з клітинного стану, його можна визначити виразом (2.1):

$$f_t = \sigma(W_f \ [h_{t-1}, x_t] + b_f \tag{2.1}$$

де:

- f_t – це вихідний сигнал запам’ятовувального шлюза, який визначає, яку частину інформації потрібно забути або відкинути з попереднього стану клітини;
- σ – сигмоїдна активаційна функція, яка перетворює вхідні дані в діапазоні від 0 до 1, що дозволяє вирішувати, яку інформацію зберігати або відкинути;
- W_f – вагові коефіцієнти для запам'ятовувального шлюза, які визначають важливість кожного вхідного та попереднього прихованого стану;
- h_{t-1} – попередній прихований стан клітини, інформація з минулого кроку часу;
- x_t поточний вхідний вектор, дані поточного кроку часу;
- b_f – вектор зміщення для запам’ятовувального шлюза, що додається до зважених вхідних даних перед їх перетворенням сигмоїдною функцією.

Forget Gate використовує сигмоїдну функцію σ для вирішення, які дані слід забути або відкинути, на основі поточних вхідних даних x_t та попереднього прихованого стану h_{t-1}. Ваги W_f та зміщення b_f налаштовуються під час навчання

моделі для оптимізації процесу забування.

Вхідні ворота (Input Gate) в моделі відповідають за оновлення інформації в клітині пам’яті, працюють за виразом (2.2):

$$\begin{aligned} i_t &= \sigma(W_i\ [h_{t-1}, x_t] + b_i \\ \tilde{C}_t &= tanh(W_C\ [h_{t-1}, x_t] + b_C \end{aligned} \tag{2.2}$$

де:

- i_t — це вихідний шлюз, який визначає, яку нову інформацію слід оновити в стані клітини;
- $\tilde{C}_t$ — кандидат на стан клітини, який пропонує можливе оновлення до стану клітини;
- σ — сигмоїдна активаційна функція, яка перетворює вхідні дані в діапазоні від 0 до 1, що дозволяє визначити, яку частину інформації треба оновити.;
- $tanh$ — гіперболічна тангенс активаційна функція, яка нормалізує значення між -1 та 1, забезпечуючи регуляцію ваги оновлень.

Змінні W і b представляють ваги та зсуви для відповідних шлюзів та слугують для керування потоком інформації через мережу під час тренування. Змінна h_{t-1} представляє попередній прихований стан, а x_t поточний вхідний дані.

Функція оновлення стану клітини (Cell State) дозволяє моделі зберігати інформацію протягом тривалого часу. Вона використовує дві частини: забування (за допомогою шлюзу забуття f_t) та додавання нової інформації (через вхідний шлюз i_t та кандидат на оновлення клітини $\tilde{C}_t$ (2.3)):

$$C_t = f_t\ \ C_{t-1} + i_t\ \ \tilde{C}_t \tag{2.3}$$

де:

- C_t — оновлений стан клітини на поточному кроці.

Вихідний шлюз LSTM моделі (Output Gate) відповідає за визначення наступного прихованого стану, який виводиться з комірки, визначається за таким виразом (2.4):

$$o_t = \sigma\ (W_o\ \ [h_{t-1}, x_t] + b_o \tag{2.4}$$

де:

- o_t — вихідний вектор шлюзу на кроці часу t;
- σ — сигмоїдальна функція активації, яка обмежує вихідні значення в інтервалі від 0 до 1;
- W_o — матриця ваг вихідного шлюзу;
- h_{t-1} — попередній прихований стан;
- x_t — віхдні дані на кроці часу t;
- b_o — Вектор упередження (bias) вихідного шлюзу.

Функція визначає, яка частина коміркового стану повинна бути виведена.

Наївний Баєсівський класифікатор використовує теорему Байєса для прогнозування ймовірності класу *A* на основі певної характеристики *B*[19] (2.5):

$$P\ (B) = \frac{P\ (B|A)\ \ P(A)}{P(A)} \tag{2.5}$$

де:

- *P(A|B)* — це умовна ймовірність класу *A*, за умови, що спостерігається характеристика *B*;
- *P(B|A)* — це ймовірність спостереження характеристики *B*, коли істиною є клас *A*;
- *P(A)* — апріорна ймовірність класу *A*, тобто ймовірність класу *A* до врахування будь-якої конкретної характеристики;
- *P(B)* — апріорна ймовірність характеристики *B*, незалежно від класу.

Модель припускає, що всі характеристики незалежні одна від одної, що спрощує розрахунки, але може знизити точність на даних, де ця незалежність не

виконується.

Модель логістичної регресії (рис. 2.2), яка є статистичним методом, що використовується для прогнозування ймовірності належності до певного класу, який зазвичай є бінарним (тобто, два можливих результати).

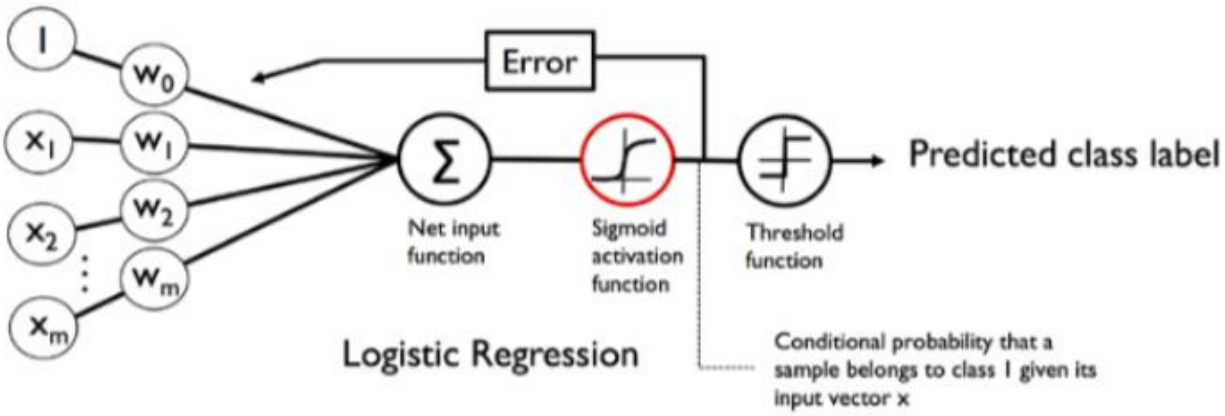


Рис. 2.2 Модель логістичної регресії [29]

Алгоритм роботи цієї моделі можна описати нижченаведеними кроками.

1. Входи (X1, X2, ..., Xm): Входи моделі логістичної регресії є набором характеристик або ознак, які використовуються для визначення результату. Кожен інпут має свою вагу, що вказує на його значення у прогнозі. Наприклад, у задачі класифікації спаму інпути можуть включати частоту певних ключових слів у тексті електронного листа. Кожен інпут множиться на відповідну вагу (W1, W2, ..., Wm). Ваги визначаються під час процесу навчання моделі і представляють значення кожної ознаки у процесі прийняття рішення.

2. Ваги (W0, W1, W2, ..., Wm): Кожна вхідна характеристика множиться на відповідну вагу. Ваги - це параметри моделі, які навчаються з даних, і вони показують, наскільки важливою є кожна характеристика для результату класифікації. W0, також відомий як вільний член або зсув (bias), є ключовим у регуляції порогу активації сигмоїдної функції. Цей параметр дозволяє моделі зміщувати вихід сигмоїди для кращого відповідання даним.

3. Суматор (Σ): Суматор обчислює лінійну комбінацію вхідних даних та їх

ваг. Фактично, він сумує всі вагомі входи разом з вільним членом, щоб сформувати загальне вхідне значення для функції активації. Це сумарне значення подається у сигмоїдну функцію для перетворення в ймовірність.

4. Сигмоїдна активаційна функція: игмоїдна функція перетворює вхід, який може мати будь-яке значення, в ймовірність між 0 та 1. Це дозволяє моделі вирішувати, чи належить спостереження до одного класу або іншого. Завдяки своїй S-подібній формі, сигмоїдна функція є ідеальною для бінарної класифікації.

5. Порогова функція: Порогова функція встановлює поріг, за яким модель вирішує, до якого класу належить спостереження. Наприклад, поріг може бути встановлений на 0.5, де значення вище цього порогу вказують на один клас, а нижче на інший. Вибір правильного порогу є важливим, оскільки він впливає на чутливість та специфічність моделі.

6. Передбачувана мітка класу: На основі виходу з сигмоїдної функції та встановленого порогу, модель робить прогноз, до якого класу належить спостереження.Кінцевий результат може бути представлений як бінарна класифікація (наприклад, спам або не спам).

7. Помилка: Помилка визначається як різниця між передбачуваною міткою класу та справжньою міткою. Ця помилка використовується для коригування ваг моделі під час процесу навчання через алгоритми оптимізації, такі як градієнтний спуск. Зменшення помилки є основною метою навчання, що веде до покращення точності та надійності моделі.

Оптимізована функція Linear SVC має на меті знайти такі параметри моделі, які мінімізують відстань між роздільною гіперплощиною та найближчими точками даних з обох класів (ці точки називаються підтримуючими векторами), вираховується таким чином [20] (2.6):

$$\frac{1}{2} w^T w + C \sum_{i=1} \quad (0, y_i(w^T \phi(x_i) + b)) \tag{2.6}$$

де:

- w – вектор ваг, який визначає орієнтацію роздільної гіперплощини в

просторі характеристик;

- b – скалярний зсув (bias), який визначає положення гіперплощини відносно походження координат;
- $\phi(x_i)$ – функція відображення вхідних даних x_i в простір характеристик. Для лінійного SVC ϕ зазвичай є ідентичною функцією, оскільки модель припускає лінійну роздільність даних;
- y_i – мітка класу для кожного вхідного вектора x_i, де y_i може приймати значення +1 або -1 залежно від того, до якого класу належить точка;
- C – регулятор, параметр який контролює компроміс між максимізацією розмежувальної маржі та мінімізацією помилки класифікації.

Основний процес, який описує формула, полягає в мінімізації функції втрат, яка складається з двох частин:

- $\frac{1}{2}w^T w$ – ця частина штрафує модель за великі значення ваг, сприяючи знаходженню роздільної гіперплощини з великою маржою (розмежуванням) між класами;
- $C\sum_{i=1} \quad (0, y_i(w^T\phi(x_i) + b))$ – ця частина відома як «hinge loss» і штрафує модель за класифікаційні помилки. Це сума помилок на всіх тренувальних даних, де помилка враховується тільки, якщо класифікована точка знаходиться на неправильній стороні гіперплощини з маржею меншою за 1.

Задача оптимізації полягає в знаходженні таких ваг w та зсуву b, які мінімізують цю функцію втрат з урахуванням вищезазначених умов.

Моделі LSTM, наївний класифікатор Байєса, логістична регресія та Linear SVC мають свої унікальні особливості та використовуються для різних завдань у машинному навчанні. LSTM використовується для збереження контексту в послідовностях даних, що є важливим для завдань, які потребують розуміння довгострокових залежностей. Наївний класифікатор Байєса є ефективним для класифікації документів, заснованої на статистичному підході. Логістична регресія

широко застосовується для прогнозування ймовірностей належності до класів. Linear SVC ефективно вирішує завдання класифікації з максимізацією розділової маржі між класами. Кожен з цих методів може бути оптимізований для покращення точності та ефективності у конкретних застосуваннях.

Для тренування моделей було обрано стохастичний градієнтний спуск (SGD) який є одним з найпоширеніших методів оптимізації в машинному навчанні та глибокому навчанні. Він використовується для тренування моделей шляхом мінімізації функції втрат шляхом оновлення ваг моделі в напрямку, що зменшує цю втрату. Основна ідея полягає в тому, щоб знаходити градієнт функції втрат на основі одного або кількох випадково вибраних навчальних прикладів (міні-пакетів) замість вирахування градієнта для всього набору даних [21].

Основна формула SGD має вигляд (2.7):

$$_{\theta_{t+1}} = \theta_t - \alpha \nabla J(\theta_t, x_i, y_i) \tag{2.7}$$

де:

- θ_t – вектор параметрів моделі на ітерації t;
- α – швидкість навчання (learning rate), яка визначає крок оновлення параметрів;
- $\nabla J(\theta_t, x_i, y_i)$ – градієнт функції втрат J відносно параметрів θ_t на основі навчального прикладу (x_i, y_i).

Робочий процес SGD виглядає так [22]:

1) вибирається міні-пакет навчальних прикладів (зазвичай випадковим чином);

2) рахується градієнт функції втрат для цього міні-пакету відносно поточних параметрів моделі;

3) параметри моделі оновлюються в напрямку зменшення втрат, множачи градієнт на швидкість навчання;

4) цей процес повторюється для багатьох ітерацій або поки не досягнуто критерію зупинки.

SGD є ефективним методом оптимізації для тренування моделей машинного навчання та глибокого навчання через його здатність швидко пристосовуватися до локальних оптимальних розв’язків та здатність працювати з великими об’ємами даних. Однак важливо правильно вибрати швидкість навчання, оскільки надто велика або занадто мала швидкість може спричинити проблеми з тренуванням моделі.

2.2 Програмна реалізація методики

Наступним етапом у методології є вибір мови програмування для реалізації роботи. Було обрано мову програмування Python, яка є дуже популярною серед дослідників та розробників у галузі машинного навчання і обробки природних мов, включаючи аналіз тексту та класифікацію емоційного забарвлення. Нижче наведено кілька причин, чому Python є доцільним вибором для таких завдань.

1. Легкість використання: Python має простий і зрозумілий синтаксис, що робить його дуже приємним для розробки. Він також має велику кількість бібліотек та модулів, які полегшують створення, навчання та оцінку моделей машинного навчання.

2. Багатофункціональність: Python має широку базу бібліотек, таких як Keras, TensorFlow, PyTorch та scikit-learn, які надають потужні інструменти для роботи з нейронними мережами та аналізу тексту.

3. Розширюваність: Python є розширюваною мовою програмування, що означає, що можна використовувати код на C/C++ для оптимізації продуктивності. Багато популярних бібліотек машинного навчання, таких як TensorFlow та PyTorch, засновані на Python і надають можливість прискорення обчислень на графічних процесорах.

4. Велике співтовариство: Python має активне та розгалужене співтовариство розробників, що веде до швидкої розробки та вдосконалення бібліотек машинного навчання. Це означає, що завжди є можливість знайти відповідь на питання або отримати допомогу в разі потреби.

Отже, Python є найкращим вибором для класифікації емоційного забарвлення тексту, завдяки своїй легкості використання, потужним бібліотекам та розширюваності. Реалізація моделей глибокого навчання для обробки тексту та класифікації емоційного забарвлення, використовуються різні ключові бібліотеки та їх компоненти. Одними з основних є:

- keras.models.Sequential: ця бібліотека використовується для створення послідовної моделі нейронної мережі в Keras, вона дозволяє додавати шари до моделі послідовно, по одному за одним;
- keras.layers: ця бібліотека містить різні шари, які можна використовувати при побудові нейронних мереж;
- keras.metrics: ця бібліотека містить різні метрики якості, такі як Precision (точність) та Recall (повнота), які можна використовувати для оцінки моделі під час тренування та тестування;
- keras.optimizers: ця бібліотека надає різні оптимізатори, які використовуються для навчання моделі;
- keras.callbacks: ця бібліотека дозволяє налаштовувати зворотні виклики під час тренування моделі;
- keras.losses: ця бібліотека містить різні функції втрат, які використовуються для оптимізації моделі під час тренування, наприклад, binary_crossentropy та categorical_crossentropy є поширеними функціями втрат для класифікаційних завдань;
- sklearn.metrics.confusion_matrix: ця бібліотека надає інструменти для обчислення матриці помилок та візуалізації результатів класифікації.

Як можна побачити з переліку, в основному фігурує бібліотека Keras, вона є важливою бібліотекою для завдань обробки тексту та класифікації емоційного забарвлення з кількох причин. Вона надає зручний та інтуїтивний інтерфейс для побудови та тренування нейронних мереж, дозволяючи легко додавати та налаштовувати шари для побудови складних архітектур. Крім того, Keras має широкий спектр шарів та компонентів, які можна використовувати для обробки

тексту та класифікації емоційного забарвлення, включаючи Embedding-шари для векторизації слів та LSTM-шари для розуміння послідовностей [23]. Інтеграція Keras з TensorFlow [24] дозволяє використовувати всі переваги цього потужного фреймворку глибокого навчання.

2.3 Конструкція алгоритму на базі РНМ

Рекурентні нейронні мережі (РНМ) є потужними моделями глибокого навчання, які дозволяють ефективно моделювати послідовні дані, такі як текст, звук чи часові ряди. Вони мають властивість запам’ятовування попередніх станів та використання цієї інформації для передачі залежностей в часі. Основна ідея РНМ полягає в тому, що вона має зв’язки між своїми нейронами, що створюють зациклені зворотні зв’язки. Це дозволяє інформації про попередні стани передаватись через час та впливати на поточні вихідні значення. Такий механізм дозволяє РНМ утримувати інформацію з попередніх кроків та враховувати її при прийнятті рішень на поточному кроці. Одним з найпоширеніших типів РНМ є Long Short-Term Memory (LSTM). LSTM має додаткову структуру, яка дозволяє контролювати та регулювати потік інформації через нейрони. Це допомагає уникнути проблеми зникнення градієнту та дозволяє більш ефективно моделювати довгострокові залежності в послідовних даних. Алгоритм роботи з використанням LSTM продемонстровано на рис. 2.3.

З рисунку видно, що алгоритм вилучення емоційної складової інформації з текстів у соціальних мережах включає ряд кроків, які дозволяють здійснити глибокий аналіз текстових даних та розкрити їх емоційну природу. Спочатку розпочинається попередня обробка даних, яка має на меті підготувати текст для подальшого аналізу. На цьому етапі проводиться видалення так званих «стоп-слів», які не несуть значущої інформації та можуть спотворювати результати. Після цього текст піддається токенізації, де кожне слово розбивається на окремі токени, а також визначенню основи слів, що дозволяє зменшити варіативність слів і спростити аналіз. Наступним важливим кроком є використання методів векторизації слів,

таких як GloVe чи Word2Vec.

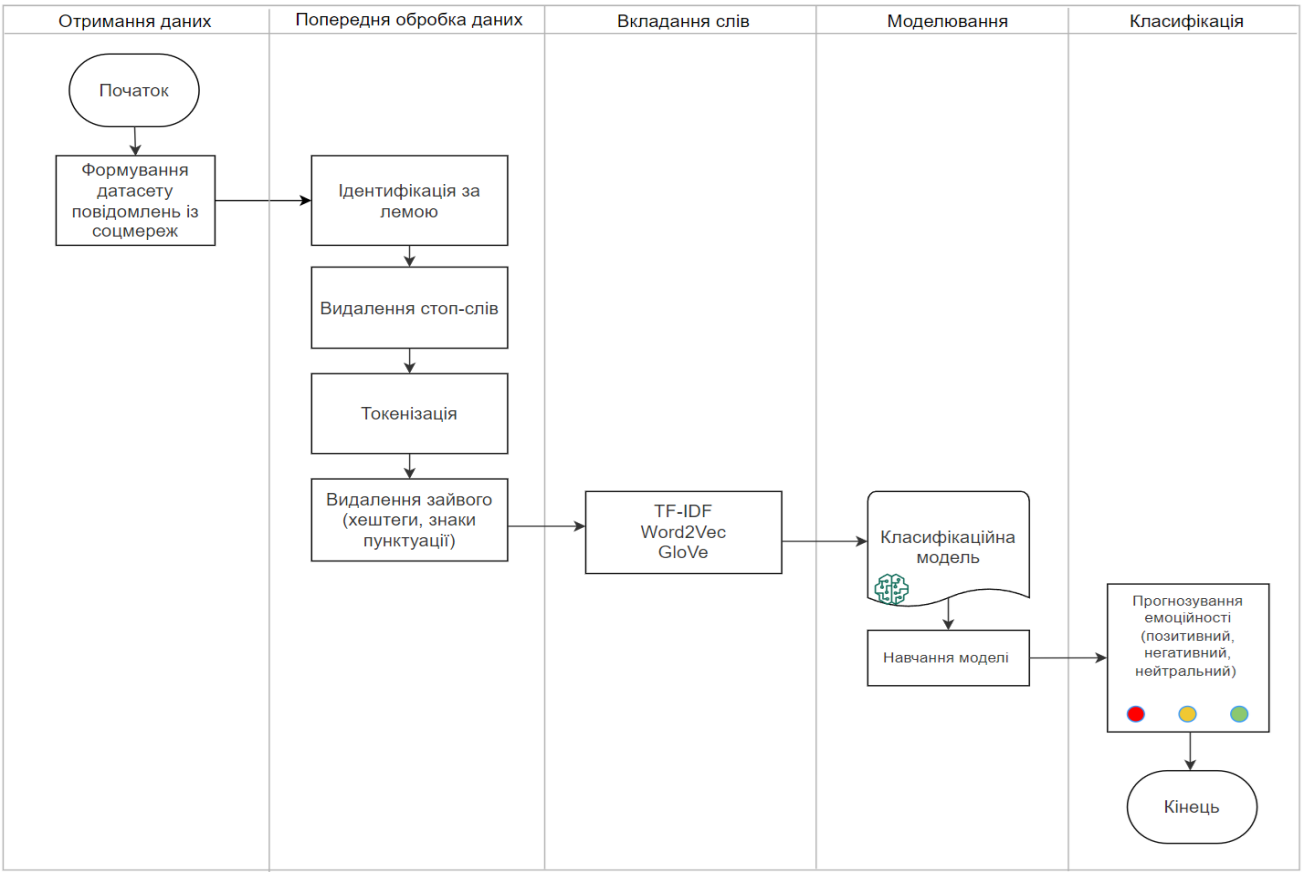


Рис. 2.3 Алгоритм вилучення емоційної складової інформації з текстів

GloVe виконує статистичний аналіз співвідношення кожного слова до інших слів у великому корпусі тексту, допомагаючи визначити семантичні зв’язки між словами [25]. Word2Vec, натомість, використовує нейронні мережі для створення векторних представлень слів, і має дві архітектури: CBOW і Skip-gram, які враховують контекст та слово відповідно [26]. Ці методи векторизації дозволяють створити багатовимірні просторові представлення слів, де семантично схожі слова розташовані близько одне до одного.

Основна ідея GloVe полягає в створенні матриці спільного входження слова-слова з великого корпусу і подальшому розкладі цієї матриці для отримання векторів слів. Далі наведено основні формули та кроки, що входять в алгоритм GloVe.

Для кожної пари слів (i, j) в корпусі обчислюється, як часто слово j

зустрічається в контексті слова i. Результатом є симетрична матриця спільного входження X, де X(i, j) представляє кількість разів, коли слово j зустрічається в контексті слова i.

На наступному кроці обчислюється розподіл ймовірностей P(i, j) як ймовірність спільного входження слів i та j. Потім виконується нормалізація матрицю спільного входження для отримання ймовірностей, шляхом поділу кожного елементу на загальну кількість пар слів (2.8):

$$\tilde{P}_{(i,j)} = \frac{X(i,j)}{\sum_k X(i,k)}. \quad (2.8)$$

Після цього визначається цільова функція, яку модель намагається мінімізувати. Вектори слів навчаються так, щоб їхній скалярний добуток відображав логарифм ймовірності спільного входження (2.9):

$$J = \sum_{i,j} f(X(i,j))(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X(i,j))^2 \quad (2.9)$$

де:

- f - це член зважування, який зменшує внесок часто зустрічаючихся пар слів;
- w_i $\tilde{w}_j$ – вектори слів для слів i, j;
- $b_i, \tilde{b}_j$ – терміни зсуву.

Після тренування вектори слів w_i та $\tilde{w}_j$ представляють семантичні значення слів.

В даному випадку застосовується LSTM мережа, яка є рекурентною нейронною мережею і використовується для аналізу послідовності словесних векторів. Останнім етапом є класифікація емоційного забарвлення тексту та прогнозування кінцевої емоції. В результаті аналізу векторних представлень слів, LSTM мережа може визначити, чи містить текст негативну, позитивну чи нейтральну емоційну складову. Таким чином, цей алгоритм дозволяє детально

дослідити емоційний вміст текстів у соціальних мережах, що є корисним для розуміння емоційної реакції користувачів та аналізу їхнього ставлення до різних тем та подій.

Модифікації, що були внесені для оптимізації процесу вилучення емоційної інформації з текстових повідомлень у соціальних мережах, включають наступні деталі.

1. Стековані двонаправлені LSTM: Ця модифікація додає глибину до архітектури моделі. Використання стекованих LSTM дозволяє моделі аналізувати текст більш докладно, розглядаючи його у багатьох напрямках. Два напрямки LSTM (вперед та назад) допомагають моделі краще зберігати та використовувати контекст інформації в обох напрямках тексту. Це особливо корисно для виявлення емоційних нюансів, оскільки деякі з них можуть бути виражені у тексті у більш складних або контекстуальних формах.

2. Додавання механізму уваги (attention): Впровадження механізму уваги дозволяє моделі приділяти більше уваги конкретним частинам тексту, які мають більший вплив на визначення емоцій. Наприклад, коли в тексті є певні ключові слова або фрази, що вказують на певний настрій або емоцію, модель може активувати увагу до цих фрагментів, що поліпшує точність визначення емоційного забарвлення.

3. Додаткові етапи обробки тексту: Додавання етапів обробки тексту для виділення важливих компонент, таких як хештеги та емотікони, допомагає моделі краще розуміти контекст інформації. Наприклад, емотікони часто виражають конкретні емоції, тому їх врахування може бути корисним для класифікації. Хештеги можуть надавати важливу інформацію про тему повідомлення, що також впливає на емоційний контекст.

Ці модифікації допомагають зробити модель більш потужною та здатною виявляти емоційні відтінки у тексті, покращуючи її точність і здатність адаптуватися до складних емоційних виразів у соціальних мережах. Архітектура нової моделі, представлена на рисунку 2.4, включає в себе декілька ключових

елементів, які значно підвищують її ефективність у розпізнаванні емоцій. Завдяки використанню передових технік обробки природної мови та глибинного навчання, модель здатна аналізувати не тільки поверхневий зміст тексту, але й глибший контекст та семантичні відносини між словами. Це дозволяє точніше визначати суб'єктивні та нюансовані емоційні стани, які часто зустрічаються у соціальних мережах.

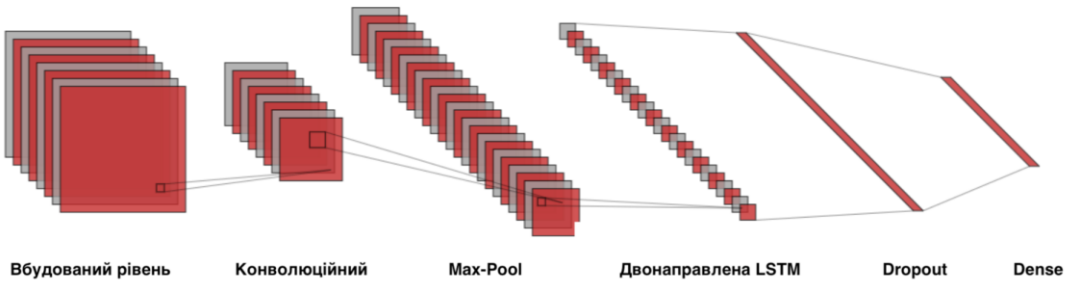


Рис. 2.4 Модифікована модель

Ключові елементи цієї моделі наведені нижче.

1. Embedding: Шар вбудовування, є фундаментальним у моделях обробки природної мови. Він перетворює вхідні індекси слів у щільні вектори, які представляють кожне слово в багатовимірному просторі. Вектори в шарі вбудовування мають здатність відображати семантичні та синтаксичні відносини між словами, дозволяючи моделі вловлювати складні мовні шаблони. Шар вбудовування є ефективним способом обробки тексту, оскільки він значно скорочує розмірність даних порівняно з однорідним кодуванням і забезпечує збереження контекстуальної інформації слів.

2. Conv1D (1D Convolutional Layer): Conv1D застосовується в задачах обробки послідовностей для виявлення локальних шаблонів у вхідних даних. Шар використовує набір фільтрів, які проходять по векторам вбудовування, аналізуючи кожен сегмент тексту для виявлення особливостей, які можуть бути корисними для класифікації або інших завдань NLP. Використання Conv1D дозволяє моделі

ефективно обробляти інформацію в масивах даних, ідентифікувати важливі закономірності у тексті та виконувати швидке та ефективне обчислення.

3. MaxPooling1D: Шар максимального об’єднання, зменшує розміри вхідних даних, вибираючи максимальне значення в кожному вікні фільтрів. Цей процес дозволяє моделі зосередитися на найважливіших особливостях, виявлених конволюційними фільтрами, ігноруючи менш значущу інформацію. Шар максимального об’єднання сприяє зменшенню обчислювальних вимог і запобігає перенавчанню, забезпечуючи кращу генералізацію моделі на нових даних.

4. Bidirectional LSTM: Двонаправлений шар довгострокової короткочасної пам’яті (LSTM), розширює звичайний LSTM, дозволяючи моделі ефективно обробляти дані як в прямому, так і в зворотному напрямку, забезпечуючи більш повне розуміння контексту тексту. Це особливо корисно в задачах, де контекст залежить від порядку слів або їх взаємодії в рамках всієї послідовності. Шари Bidirectional LSTM допомагають моделі захоплювати довгострокові залежності в текстових даних, покращуючи здатність розпізнавати складні мовні структури та відтінки.

5. Dropout: Dropout є технікою регуляризації, яка "вимикає" випадково вибрані нейрони під час тренування. Це запобігає перенавчанню мережі, допомагаючи їй краще узагальнювати вивчені закономірності на нових даних.Використання шару Dropout може значно покращити здатність моделі до адаптації та уникнення залежності від шуму або нерелевантних особливостей у тренувальних даних.

6. Dense: Повнозв'язний шар (Dense) об’єднує особливості, отримані з попередніх шарів, і використовується для виходу кінцевого прогнозу або класифікації. Цей шар може інтерпретувати виявлені шаблони та виробляти висновки на їх основі. Dense шари є критичним компонентом багатьох моделей глибинного навчання, дозволяючи моделям виконувати складні задачі класифікації та регресії, інтегруючи високорівневі особливості та патерни, виявлені попередніми шарами мережі.

Завдяки своїй гнучкості та адаптивності, архітектура моделі може бути налаштована для вирішення специфічних потреб різних застосувань. Наприклад, у задачах, де необхідно виявляти тонкі емоційні нюанси, модель може бути оптимізована для більш глибокого аналізу сентименту, використовуючи контекстуальні відносини та семантичні закономірності. Крім того, модель може бути ефективною у завданнях розпізнавання тем або категоризації тексту, де важливо розуміти загальну тематику та зміст великих обсягів тексту.

2.4 Перевірка ефективності алгоритму

Оцінка ефективності алгоритму є важливим кроком у визначенні його потенціалу та застосовності. В контексті даного дослідження ми розглядаємо ефективність алгоритму, який використовується для вилучення емоційної інформації з текстів у соціальних мережах. Розроблена модель та алгоритм пропонує автоматичний підхід до вилучення емоційної інформації з текстів у соціальних мережах. Він використовує різноманітні техніки обробки тексту та глибокого навчання, що дозволяють аналізувати великий обсяг текстових даних та класифікувати їх за емоційним забарвленням. Для демонстрації ефективності розробленого алгоритму оптимізації можна порівняти його з стандартним алгоритмом LSTM, за наведеними нижче аспектами.

1. Складність моделі: цей аспект вказує на те, як обидва алгоритми обробляють дані та чи враховують вони контекст з інших напрямків. Розроблений алгоритм оптимізації може враховувати додатковий контекст, що може поліпшити його ефективність у деяких завданнях. Вища складність оптимізованого алгоритму вказує на його здатність більш ефективно враховувати контекст і різноманітність даних, забезпечуючи більш точне виявлення закономірностей та відносин між елементами. Особливо це стає важливим у задачах, де необхідно уловлювати складні мовні структури або залежності.

2. Продуктивність: вказує на те, наскільки швидко та обчислювально

складно працюють обидва алгоритми. Оптимізований алгоритм може бути менш обчислювально складним та швидшим у порівнянні зі стандартним LSTM.

3. Навчальні дані: цей аспект стосується того, які характеристики даних можуть сприяти ефективності одного алгоритму над іншим. Якість та характеристики навчальних даних мають значний вплив на ефективність алгоритму. Оптимізований алгоритм може бути більш ефективним, якщо навчальні дані містять важливий контекст на обох напрямках.

4. Застосування: для яких завдань кожен алгоритм може бути більш придатним. Розуміння контексту є ключовим для багатьох задач обробки природної мови, від класифікації тексту до машинного перекладу. Оптимізований алгоритм може бути більш ефективним для завдань, які вимагають розуміння контексту з обох сторін.

5. Час і ресурси для навчання: цей аспект вказує на те, скільки пам’яті та часу потрібно для навчання обох алгоритмів. Оптимізований алгоритм може бути менш вимогливим до ресурсів.

6. Точність і метрики: наскільки точно та ефективно працюють обидва алгоритми у вимірюваних метриках. Оптимізований алгоритм може досягати кращих результатів у вимірюваних метриках, якщо він покращує розуміння контексту.

Враховуючи дані аспекти, була створена порівняльна таблиця алгоритмів LSTM (див. табл. 2.1), що допомагає визначити їх відповідність та ефективність для конкретного завдання.

За результатами порівняння можна зробити декілька висновків стосовно двонапрямлених LSTM. Двонаправлені LSTM мають перевагу в задачах, де важливо розуміння контексту з обох напрямків вхідних даних. Це дозволяє їм більш точно інтерпретувати послідовності, де контекст з початку та кінця даних є ключовим для розуміння загального значення. Вони часто досягають вищої точності в більш складних задачах, зокрема в таких, де потрібно аналізувати багатий контекст або нюанси. Це робить їх ідеальними для розпізнавання емоцій,

обробки природної мови та інших задач, де важливі контекстуальні взаємозв'язки. Двонаправлені LSTM особливо корисні в лінгвістичних застосуваннях, де зміст слова часто залежить від контексту, що передує йому та слідує за ним. Вони можуть краще інтерпретувати фрази та тексти зі складною структурою. Хоча двонаправлені LSTM вимагають більше обчислювальних ресурсів і часу для навчання порівняно зі звичайними LSTM, це інвестиція часто виправдовується підвищенням продуктивності в складних задачах.

Таблиця 2.1

Порівняльна таблиця алгоритмів

Аспект	**LSTM**	**Двонаправлені LSTM**
Складність моделі	Обробляють дані послідовно, не враховуючи контекст з інших напрямків.	Обробляють дані з обох напрямків, захоплюючи контекст з обох напрямків.
Продуктивність	Зазвичай менш обчислювально складні та швидше навчаються.	Зазвичай більш обчислювально складні та потребують більше часу для навчання.
Навчальні дані	Ефективні для послідовностей без значущого контексту на початку та в кінці.	Можуть бути ефективнішими, якщо важливий контекст на обох напрямках.
Застосування	Добре підходять для аналізу настрою в текстах, де контекст лінійний.	Більш придатні для більш нюансованого розпізнавання емоцій та складних текстів.
Час і ресурси для навчання	Зазвичай менше пам'яті та часу для навчання.	Вимагають більше пам'яті та часу для навчання.
Точність і метрики	Можуть досягати високої точності, але обмежені контекстом.	Можуть досягати високої точності, особливо в задачах з багатим контекстом.

Двонаправлені LSTM є більш потужним інструментом у ситуаціях, де необхідне глибоке розуміння контексту з обох напрямків вхідних даних. Хоча вони вимагають більше ресурсів для навчання, їх здатність досягати високої точності в складних задачах з багатим контекстом є значною перевагою.

Для кращого порівняння та візуалізації результатів можна порівняти моделі за ключовими аспектами тренувальними, перевірочними графіками точності та втрати. Для початку розглянемо тренувальну і перевірочну точність для звичайного алгоритму LSTM (рис. 2.5).

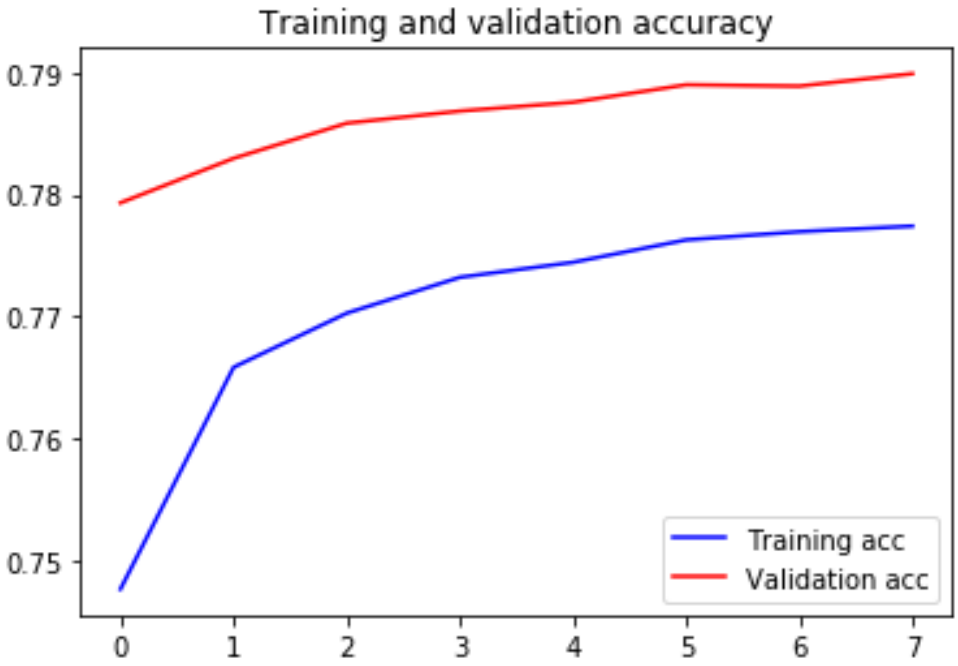


Рис. 2.5. Тренувальна і перевірочна точність для LSTM

Тренувальна точність (Training acc) показує, наскільки точно модель передбачає тренувальні дані. На графіку видно, що точність на тренувальному наборі даних збільшується з кожною епохою, що є ознакою того, що модель вчиться і адаптується до тренувальних даних. Перевірочна точність (Validation acc): відображає точність моделі на наборі даних, який не використовувався під час тренування. Цей показник є важливим, оскільки він демонструє здатність моделі генералізувати на нових даних. На графіку видно, що перевірочна точність стабільно зростає, хоча і повільніше, ніж тренувальна точність.

Наступним розглянемо графік втрат (рис. 2.6).

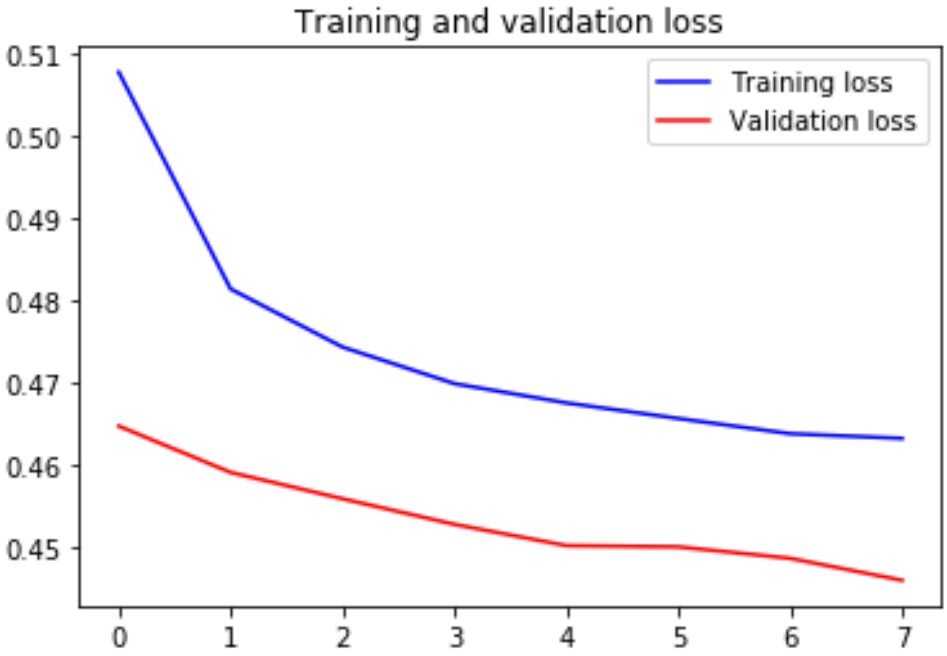


Рис. 2.6. Тренувальна і перевірочна втрата для LSTM

Тренувальна втрата (Training loss) показник того, наскільки далеко прогнози моделі від фактичних значень для тренувального набору даних. Зниження втрати з часом свідчить про те, що модель покращує свої прогнози. Перевірочна втрата (Validation loss): Представляє рівень втрат для даних перевірки. Ідеально, коли тренувальна та перевірочна втрата знижуються паралельно, що свідчить про те, що модель ефективно навчається без перенавчання. За даними графіків можна сформувати певні висновки щодо LSTM:

1. Навчання: модель покращує свою продуктивність з кожною епохою, що свідчить про ефективність тренувального процесу;
2. Перенавчання: немає очевидних ознак перенавчання (overfitting), оскільки перевірочна точність зростає поряд з тренувальною точністю, а перевірочна втрата продовжує знижуватися. Це вказує на те, що модель добре узагальнює.
3. Загальна ефективність: Враховуючи, що втрата зменшується та точність зростає, можна зробити висновок, модель працює добре на цих даних.

Розглянемо аналогічні графіки для двонаправленого алгоритму LSTM,

тренувальна та перевірочна точність (рис. 2.7).

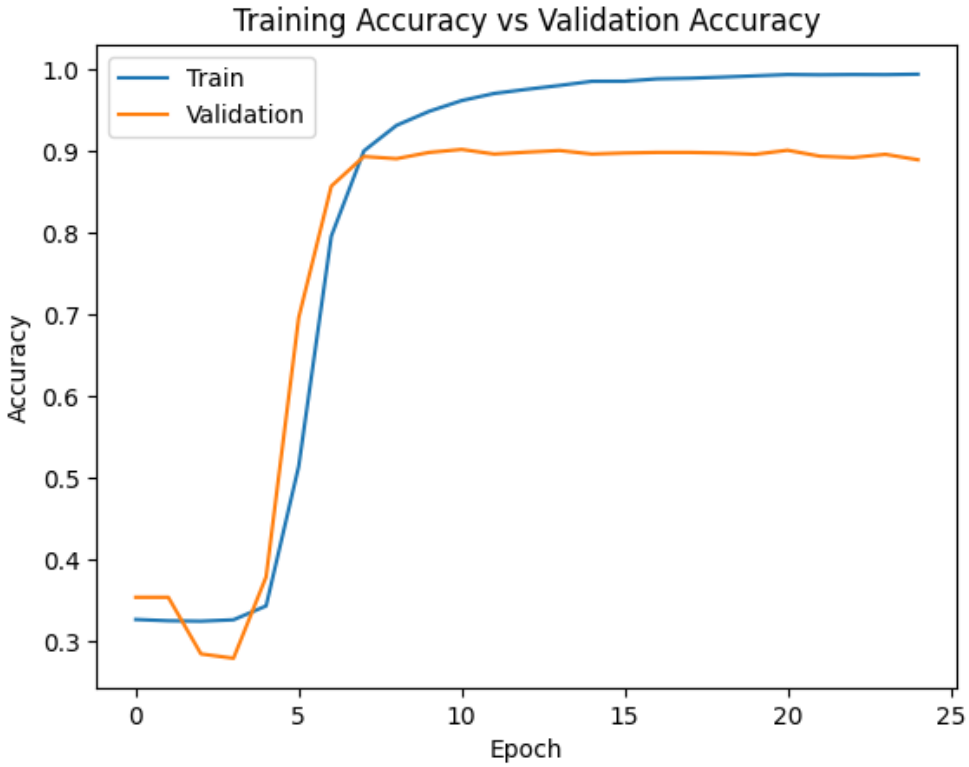


Рис. 2.7. Тренувальна і перевірочна точність для двонаправленої LSTM

Тренувальна точність (Train) модель швидко досягає високої точності, що свідчить про її здатність ефективно вчитися та адаптуватися до тренувальних даних. Така швидка конвергенція є ознакою потужності двонаправленої LSTM архітектури. Перевірочна точність (Validation) показник точності на перевірочному наборі даних стабілізується на високому рівні, що свідчить про гарне узагальнення моделі. Стабільність перевірочної точності на високому рівні вказує на те, що модель не лише добре навчається, але й ефективно застосовує навчене до невідомих даних.

Графік для тренувальних та перевірочних втрат для двонаправленої LSTM зображено на рисунку 2.8.

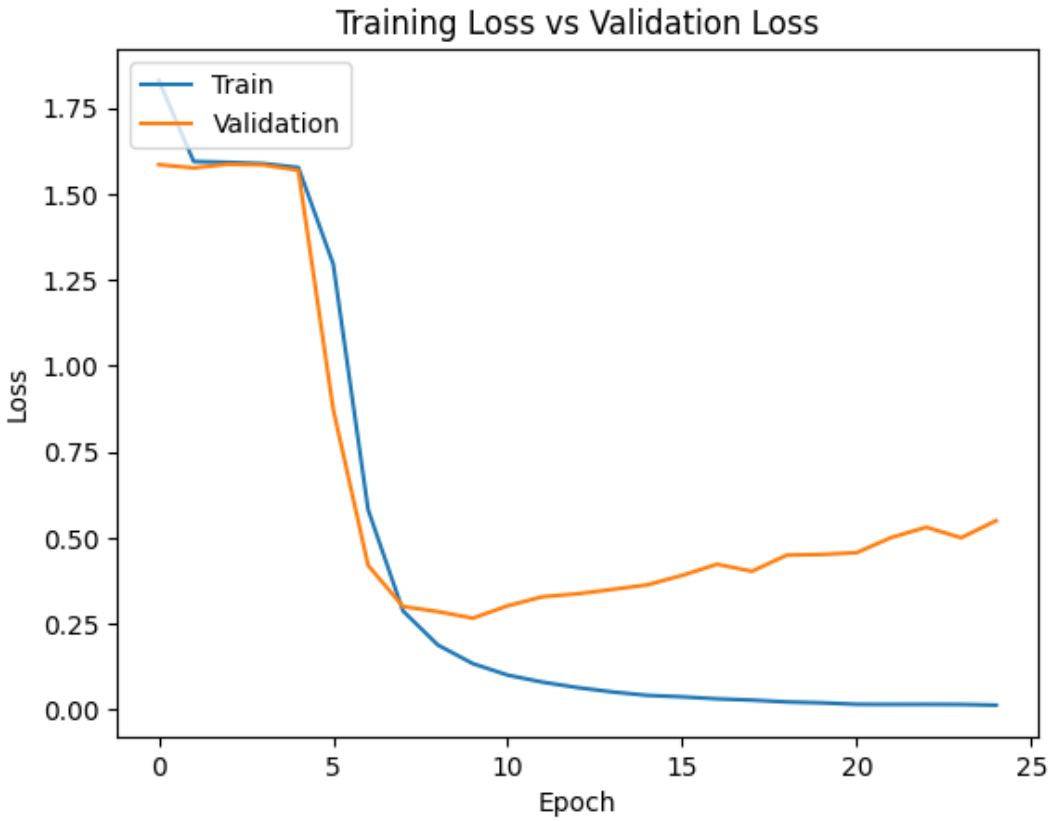


Рис. 2.8 Тренувальна і перевірочна втрата для двонаправлених LSTM

Тренувальна втрата (Train) спостерігається швидке зменшення тренувальної втрати, що підтверджує ефективність моделі у вивченні тренувального набору даних. Перевірочна втрата (Validation): незважаючи на деяке зростання після початкового спаду, перевірочна втрата залишається на порівняно низькому рівні. Це відносне зростання не обов'язково є ознакою перенавчання, особливо якщо воно не супроводжується падінням перевірочної точності. Підсумовуючи дані графіків для двонаправлених LSTM можна зробити наступні висновки:

1. Вища точність: двонаправлені LSTM забезпечують вищу точність на тренувальних і перевірочних даних, що свідчить про їх здатність більш ефективно захоплювати контекстуальну інформацію з обох напрямків послідовностей;
2. Краще узагальнення: збереження перевірочної точності на високому рівні показує, що модель краще узагальнює та менш схильна до перенавчання порівняно зі звичайними LSTM;

3. Ефективність навчання: швидка конвергенція моделі свідчить про те, що двонаправлені LSTM можуть бути більш ефективними в певних типах задач, де необхідне розуміння контексту в обох напрямках послідовностей.

Ці характеристики вказують на те, що в даному випадку двонаправлені LSTM перевершують звичайні LSTM, особливо в умовах, де контекстуальна інформація є критично важливою для точності моделі.

3 АНАЛІЗ РЕЗУЛЬТАТІВ ТА ЇХ ПРАКТИЧНЕ ВПРОВАДЖЕННЯ

3.1 Оцінка результатів дослідження

Оцінка результатів дослідження є важливим етапом у визначенні вагомості та значущості отриманих висновків. Необхідно оцінити отримані результати дослідження алгоритму, який використовується для вилучення емоційної інформації з текстів у соціальних мережах. В алгоритмі використовувалось 4 моделі, з отриманих результатів було створено таблицю порівняння (див. табл. 3.1).

Таблиця 3.1

Порівняння значень використаних моделей

Метрика	**LSTM**	**Naive Bayes**	**LinearSVC**	**Логістична регресія**
Accuracy	**0.9152**	0.80	0.82	0.83
Precision	**0.9175**	0.81	0.82	0.83
Recall	**0.9127**	0.79	0.81	0.82
F1 Score	**0.9151**	0.80	0.82	0.83

З таблиці видно, що серед чотирьох порівнюваних моделей LSTM показує найкращі результати за всіма метриками: Accuracy (точність класифікації), Precision (точність), Recall (повнота) та F1 Score (гармонійне середнє Precision і Recall). Особливо високі показники у LSTM за метриками Precision та F1 Score, що є дуже важливими в багатьох застосуваннях, де важлива точність визначення позитивного класу. На другому місці за ефективністю йде логістична регресія, яка показує стабільно вищі результати в порівнянні з Naive Bayes та LinearSVC, але все ще нижчі за LSTM. LinearSVC показує помірковані результати, трохи кращі, ніж Naive Bayes, але значно нижчі за LSTM та логістичну регресію. Naive Bayes має найнижчі показники за всіма метриками, що може бути зумовлено його статистичними припущеннями про незалежність ознак, які можуть не відповідати

дійсності в конкретному випадку застосування.

Загалом, висновок може бути наступний, LSTM є найефективнішою моделлю для цього конкретного набору даних і завдання, забезпечуючи найвищу точність і баланс між всіма метриками оцінки ефективності. Це може вказувати на те, що задача, для якої були натреновані моделі, має сильну часову або послідовнісну залежність, яку LSTM здатна ефективно моделювати. Логістична регресія також є гідним вибором, особливо якщо потрібна простіша модель з меншими вимогами до обчислювальних ресурсів. Naive Bayes і LinearSVC можуть бути розглянуті для завдань, де обмеження на обчислювальні ресурси є надзвичайно жорсткими, але слід очікувати нижчу ефективність.

Вивчення сентименту в соціальних мережах є ключовим аспектом розуміння публічної думки та взаємодії між користувачами. Текстові дані, зокрема твіти, можуть надавати цінну інформацію про емоційні стани та реакції людей на різноманітні події. Аналіз довжини текстів твітів з позитивним сентиментом дозволяє оцінити не лише частоту та інтенсивність позитивних висловлювань, але й зрозуміти, як короткість чи довжина повідомлень корелює з емоційним забарвленням контенту. На наведеній візуалізації (рис. 3.1) представлено описову статистику та гістограму, які ілюструють розподіл довжини текстів твітів, що містять позитивний сентимент.

Візуалізація включає аналіз кількості твітів та вивчення центральної тенденції та розсіювання даних. З таблиці (див. табл. 3.1) можна зрозуміти, що аналізовано 1543 твіти. Середня довжина тексту становить приблизно 18 символів, зі стандартним відхиленням близько 11 символів, що свідчить про помірну варіативність довжини твітів. Мінімальна довжина тексту – 1 символ, а максимальна – 54 символи. Перші 25% твітів мають довжину до 10 символів, медіана (50%) – 16 символів, що означає, що половина всіх твітів мають довжину 16 символів або менше. Третій квартиль (75%) показує, що 75% твітів мають довжину 23 символи або менше. Гістограма вказує на найбільшу частоту (count) твітів з довжиною від 10 до 20 символів, що є найпоширенішою довжиною для

позитивних твітів у наборі даних. Розподіл має правий хвіст, де дуже мало твітів мають дуже велику кількість символів (більше 30). Це може свідчити про те, що користувачі схильні виражати позитивні емоції в більш коротких твітах. Ці дані можуть бути корисними для аналізу поведінки користувачів у соціальних мережах, планування маркетингових кампаній або для розробки алгоритмів обробки природної мови, зокрема, для аналізу сентименту.

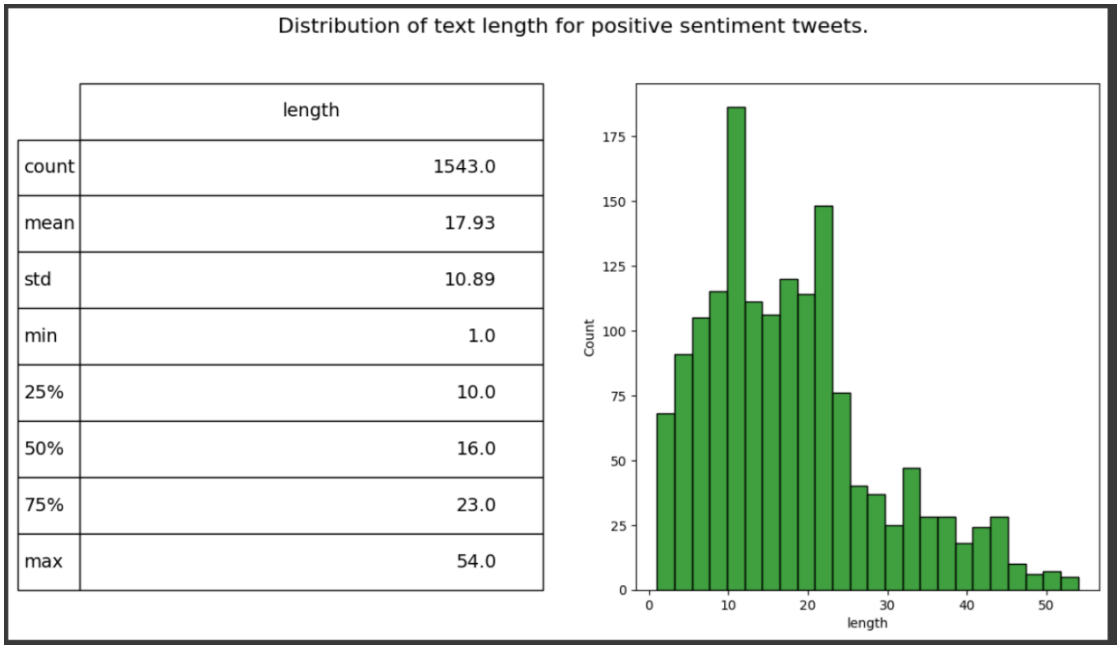


Рис. 3.1 Візуалізації та статистика позитивних твітів

Візуалізація негативних твітів представлена на рисунку 3.2. Кількість аналізованих твітів (count) становить 5955. Середня довжина тексту (mean) приблизно 22 символи. Стандартне відхилення (std) – близько 11, що свідчить про досить значну варіативність у довжині твітів. Мінімальна довжина тексту (min) – 1 символ, максимальна (max) – 61 символ. Перший квартиль (25%) – 14 символів, медіана (50%) – 21 символ, і третій квартиль (75%) – 26 символів. Це означає, що 25% твітів мають довжину не більше 14 символів, половина твітів мають довжину не більше 21 символу, і 75% твітів мають довжину не більше 26 символів. Гістограма показує, що найбільша кількість твітів має довжину між 10 та 30 символами, із піком частоти між 20 та 25 символами. Розподіл також має правий

хвіст, де кількість твітів з довжиною більше 30 символів швидко зменшується.

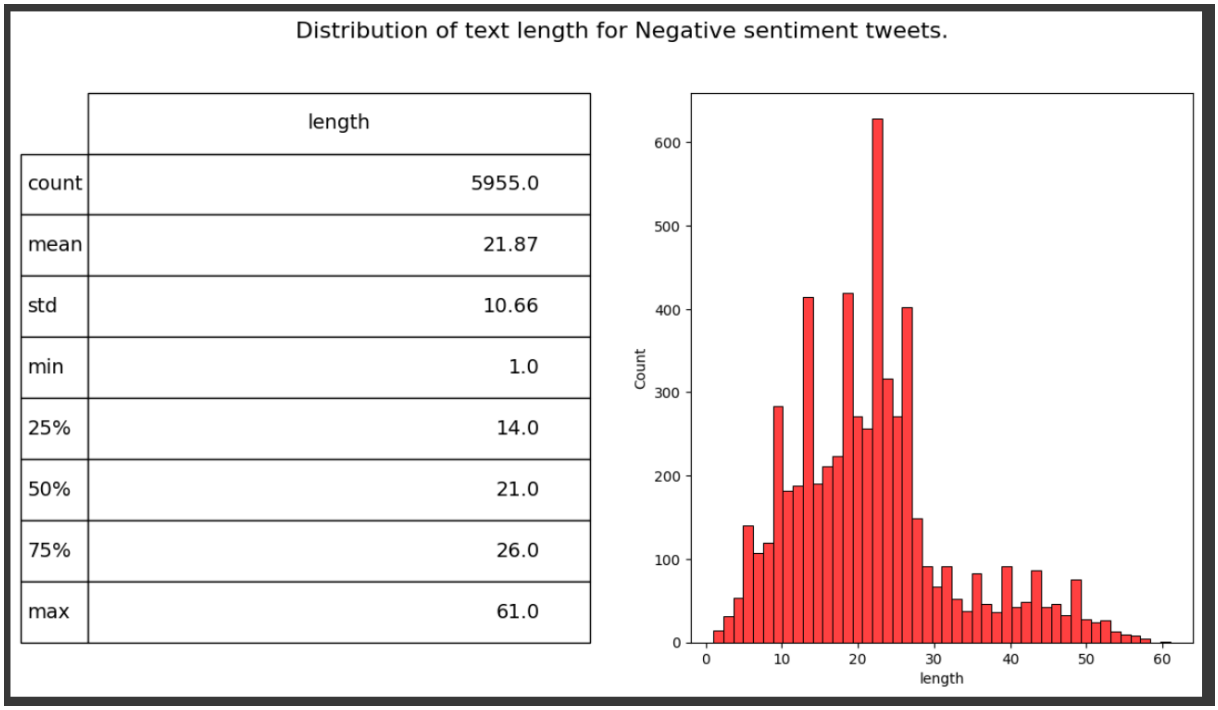


Рис. 3.2 Візуалізації та статистика негативних твітів

Порівняно з розподілом для позитивних твітів, можна помітити, що середня довжина негативних твітів трохи більша. Це може вказувати на тенденцію користувачів використовувати більшу кількість символів, коли вони висловлюють негативні емоції або думки. Однак, для більш точного аналізу цих висновків потрібно брати до уваги контекст та інші змінні, які можуть впливати на довжину твітів.

Матриця невідповідності, відома також як матриця помилок або конфузійна матриця, є фундаментальним інструментом в машинному навчанні та статистиці для оцінювання ефективності алгоритмів класифікації. Цей інструмент дозволяє не тільки визначити кількість правильних та неправильних передбачень, які зробив класифікатор, але й надає інформацію щодо конкретних типів помилок. Важливість матриці невідповідності полягає в її здатності детально розкласти результати передбачення на чотири категорії:

- істинно позитивні (TP);

- істинно негативні (TN);
- помилково позитивні (FP);
- помилково негативні (FN).

Завдяки цьому, можливо розрахувати ключові метрики, такі як точність (precision), повнота (recall), F1-міру та точність класифікації (accuracy). Кожна з цих метрик відіграє роль в міркуваннях щодо балансу між типами помилок та важливості правильних передбачень для конкретного додатку. Матрицю невідповідності, отриману в результаті використання алгоритму наведено на рисунку 3.3.

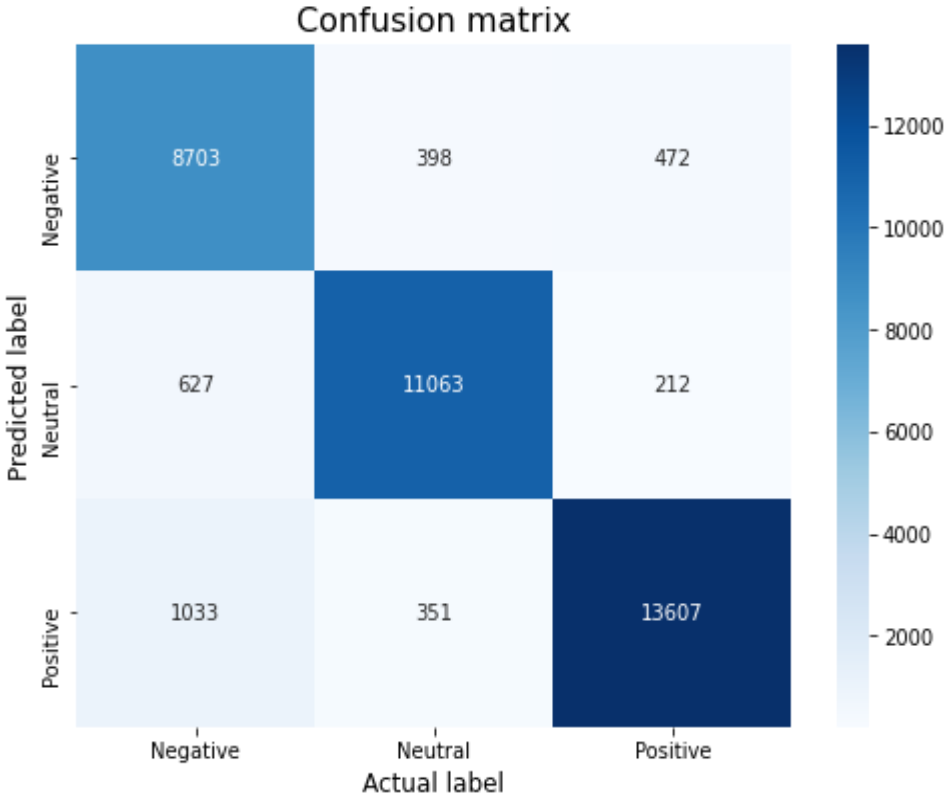


Рис. 3.3 Матриця невідповідності

З рисунку 3.3 видно, що матриця невідповідності представляє результати класифікації трьох класів (позитивний, негативний, нейтральний), що були отримані за допомогою моделі аналізу емоцій на основі нейронної мережі. Елементи в матриці мають наступні значення:

1. Істинно Позитивні (TP) для Позитивного класу: 8703 твіти було вірно класифіковано як позитивні;
2. Помилково Негативні (FN) для Позитивного класу: 398 твітів було невірно класифіковано як не позитивні (негативні або нейтральні);
3. Помилково Позитивні (FP) для Негативного класу: 472 твіти було невірно класифіковано як негативні;
4. Істинно Негативні (TN) для Негативного класу: 13607 твітів було вірно класифіковано як не позитивні;
5. Істинно Негативні (TN) для Нейтрального класу: 11063 твіти було вірно класифіковано як нейтральні;
6. Помилково Позитивні (FP) для Нейтрального класу: 627 твітів було невірно класифіковано як нейтральні;
7. Помилково Негативні (FN) для Нейтрального класу: 1033 твіти було невірно класифіковано як не нейтральні (позитивні або негативні);
8. Значення 351 може відображати кількість помилково класифікованих твітів між двома з трьох класів;
9. Значення 212 також може відображати кількість помилково класифікованих твітів між двома з трьох класів.

На основі представленої матриці невідповідності можна зробити висновок, що нейронна мережа демонструє хорошу здатність до класифікації твітів за емоційним забарвленням. Висока кількість істинно позитивних (8703) та істинно негативних (13607) результатів свідчить про ефективність моделі у визначенні позитивних та негативних твітів відповідно. Крім того, значна кількість істинно нейтральних результатів (11063) вказує на те, що модель також добре справляється з виявленням нейтральних твітів.

3.2 Принцип використання розробленої методики

Представлена методика використовує передові технології та підходи для

точного виявлення емоційних станів в текстових даних, зокрема у соціальних мережах, як-от твіти. Суть методики полягає у застосуванні комплексного програмного забезпечення, яке базується на моделі нейронної мережі для аналізу емоцій. Архітектура цієї мережі включає в себе шар вбудування, що трансформує слова в щільні вектори, одновимірну згортку, що вловлює локальні залежності в тексті, та максимальне пулінгування для вибору найважливіших ознак. Двонаправлений LSTM дозволяє моделі ефективно враховувати контекст інформації як в прямому, так і в зворотному напрямку. Випадкове вимкнення (dropout) використовується для запобігання перенавчанню моделі, а щільний вихідний шар із застосуванням активації softmax забезпечує можливість класифікації текстів на кілька класів. Тренування моделі відбувається з використанням стохастичного градієнтного спуску (SGD), методу оптимізації, який допомагає моделі ефективно оновлювати ваги на основі кожного тренувального прикладу. Перед обробкою тексту до навчання моделі, дані проходять ряд етапів попередньої обробки: перетворення тексту в нижній регістр, вилучення неалфавітних символів, токенізація для розбиття тексту на слова або фрази, вилучення стоп-слів для відкидання загальновживаних слів, що не несуть важливої інформації, та стемінг для зведення слів до їх кореневої форми.

В основі програмного забезпечення лежить Keras, високорівневий API нейронних мереж, який будується на основі TensorFlow, що забезпечує інтуїтивне конструювання та швидке прототипування моделей. Для завдань попередньої обробки даних також використовуються бібліотеки з scikit-learn.

Для програмної реалізації методики необхідні наступні інструменти:

1) Sequential це клас в Keras, який використовується для ініціалізації лінійного стеку шарів для моделі - це один із способів створення моделі, де шари додаються послідовно;

2) Embedding шар вбудування, який перетворює позитивні цілі числа (індекси) у щільні вектори фіксованого розміру; Conv1D одновимірний згортковий шар, який зазвичай використовується в обробці часових послідовностей або

текстових даних; MaxPooling1D шар максимального пулінгу для одновимірних сигналів, який допомагає зменшити розмір проміжних представлень і виловлювати важливі ознаки; Bidirectional обгортка для RNN шарів, таких як LSTM, що дозволяє мережі вчитися з контексту в обох напрямках (прямому і зворотному). LSTM двонаправлений довгостроковий шар пам’яті, який використовується для навчання залежностей у часових послідовностях; Dense щільний шар (або повнозв’язний шар), який є стандартним шаром нейронної мережі, де кожен вхід з’єднується з кожним виходом; Dropout: шар, який випадково «вимикає» деякі виходи нейронів під час тренування, що допомагає запобігти перенавчанню;

3) Precision метрика, яка вимірює точність прогнозів моделі (частка істинно позитивних результатів в усіх позитивних прогнозах). Recall: метрика, яка вимірює повноту прогнозів моделі (частка істинно позитивних результатів в усіх реально позитивних випадках);

4) SGD оптимізатор, який використовується для оновлення вагів на основі градієнтів з метою мінімізації функції втрат. RMSprop: оптимізатор, що регулює швидкість навчання, роздільно для кожного параметра, використовуючи рухоме середнє квадрату градієнтів.

5) Datasets модуль Keras, який містить зразки даних, які можна використовувати для експериментів і тренування моделей.

6) LearningRateScheduler Callback, який дозволяє змінювати швидкість навчання в процесі тренування. History Callback, який записує результати тренування (такі як втрати та метрики) у вигляді історії для аналізу.

7) losses модуль, який містить функції втрат, такі як крос-ентропія, які використовуються для оцінки розбіжності між прогнозами та реальними даними.

8) confusion_matrix функція з бібліотеки scikit-learn, яка використовується для обчислення матриці невідповідності, що дозволяє оцінити ефективність класифікації моделі.

Ці імпорти разом формують основу для створення, конфігурування, тренування та оцінки нейронної мережі, що використовується.

Наступним кроком є розділення даних на набори для тренування, валідації та тестування. Лістинги коду знаходяться у додатку Б.

Представлені незалежні (признаки) та залежні (цільові) змінні датасету відповідно. train_test_split розділяє X і y на тренувальний набір (X_train, y_train) та тестувальний набір (X_test, y_test). test_size=0.2 вказує, що 20% даних будуть використані для тестувального набору, тоді як решта 80% даних піде на тренувальний набір. random_state=1 встановлює насіння для генератора випадкових чисел, щоб забезпечити відтворюваність результатів при кожному запуску коду. Після першого розбиття, тренувальний набір даних (X_train, y_train) додатково розділяється на новий тренувальний набір і валідаційний набір (X_val, y_val). test_size=0.25 вказує, що з решти 80% тренувальних даних 25% буде використано як валідаційний набір. Це означає, що 60% початкового датасету залишається для тренування моделі, 20% для валідації та 20% для тестування. random_state=1 знову забезпечує відтворюваність результатів. Результатом виконання цього коду є три пари наборів даних:

1) тренувальний набір (X_train, y_train), який використовується для навчання моделі;

2) перевірочний набір (X_val, y_val), який використовується для тонкого налаштування параметрів моделі та регулярної оцінки її ефективності під час тренування;

3) тестувальний набір (X_test, y_test), який використовується для оцінки кінцевої ефективності моделі після завершення тренування і перевірки.

Встановлюються гіперпараметри, які є конфігураційними налаштуваннями, використовуваними для управління процесом тренування нейронної мережі. Використовуються такі гіперпараметри, як:

- vocab_size = 5000: Це визначає розмір словникового запасу, що використовується в моделі. У контексті шару вбудування, це означає, що тільки 5000 найчастіших слів будуть включені до моделі. Всі інші слова будуть розглядатися як один і той же, невідомий токен;

- embedding_size = 32: Це розмір вектора вбудування для кожного слова. Замість використання one-hot encoding, що може бути дуже розрідженим і великим, кожне слово представляється щільним вектором з 32 числами (ознаками);
- epochs=20: Це кількість проходів навчальних даних через модель під час тренування. Тобто, весь тренувальний датасет буде пройдений через модель 20 разів;
- learning_rate = 0.1: Швидкість навчання використовується в оптимізаторі під час процесу навчання для оновлення вагів моделі. Високий коефіцієнт навчання може прискорити процес навчання, але також може призвести до нестабільності або перестрибування оптимальних значень;
- decay_rate = learning_rate / epochs: Це коефіцієнт зниження швидкості навчання, який використовується для поступового зменшення швидкості навчання з кожною епохою. Це робиться для того, щоб модель почала з великої швидкості навчання для швидкого руху та поступово зменшувала цю швидкість, щоб уникнути перестрибування оптимальних значень вагів;
- momentum = 0.8: Коефіцієнт імпульсу, який використовується в оптимізаторі для врахування попереднього оновлення ваг при обчисленні поточного оновлення. Це може допомогти прискорити навчання та згладити коливання.

Коли всі ці гіперпараметри встановлені, вони використовуються для конфігурування моделі та її тренувального процесу, щоб забезпечити оптимальне навчання та уникнути перенавчання або недостатнього навчання.

Наступним кроком є створення архітектури нейронної мережі в Keras, описуючи послідовність шарів та як вони будуть стековані один за одним. Створюється екземпляр оптимізатора SGD з певною швидкістю навчання та імпульсом. nesterov=False вказує, що в даному випадку не використовується ускорення Nesterov. Далі ініціюється нова послідовна модель, яка дозволяє

додавати шари в лінійній послідовності. Додається шар вбудування, який перетворює вхідні цілі числа (індекси слів) у щільні вектори вказаного розміру (embedding_size). input_length=max_len визначає розмір вхідної послідовності. Додається одновимірний згортковий шар (Conv1D) з 32 фільтрами та ядром розміром 3. padding='same' означає, що вихід матиме той же розмір, що і вхід, завдяки додаванню падінгу. Функція активації relu використовується для введення нелінійності. Додається шар максимального пулінгу (MaxPooling1D), який зменшує розмір вихідних даних з кожного фільтра вдвічі (pool_size=2), вибираючи найбільше значення з кожного вікна пулінгу.

Додається двонаправлений шар Bidirectional LSTM з 32 одиницями. return_sequences=True означає, що шар буде повертати вихідні дані за кожний часовий крок, що дозволяє шарам послідовностей отримувати послідовні дані. Додається шар випадкового вимкнення (Dropout) з імовірністю 0.2, який «вимикає» випадково вибрані нейрони під час тренування, запобігаючи перенавчанню. Фінальний щільний шар (Dense) з трьома одиницями та softmax активацією використовується для класифікації вхідних даних у три категорії. Softmax забезпечує виведення ймовірностей класифікації для кожного класу. Кожен шар моделі побудований з урахуванням специфіки задачі класифікації тексту та архітектури нейронної мережі, яка є найбільш ефективною для даної задачі.

Створюється список sentiment_classes, який містить назви класів, що використовуються моделлю для класифікації тексту. Задається максимальна довжина послідовності, яку приймає модель. Це значення використовується для падінгу (доповнення) послідовностей до однакової довжини. Вхідний текст перетворюється в послідовність цілих чисел за допомогою об'єкта токенізатора, який має бути передбачений у коді. Кожне унікальне слово в тексті перетворюється на унікальне ціле число. Послідовності доповнюються ззаду (padding='post') до max_len за допомогою функції pad_sequences, щоб вони мали однакову довжину, необхідну для моделі. Виконується передбачення моделлю для доповненої послідовності. Функція argmax використовується для отримання індексу

найбільшої ймовірності в передбаченнях, який відповідає класу сентименту. Виводиться результуючий клас сентименту, вибраний з sentiment_classes відповідно до індексу, отриманого з передбачення моделі.

Функція predict_class може бути використана для швидкого і простого передбачення емоційного забарвлення текстів без необхідності виконувати вручну всі кроки підготовки даних, оскільки всі вони автоматизовані всередині функції.

Створюється матриця невідповідності. Функція plot_confusion_matrix(model, X_test, y_test) приймає три аргументи:

- model: модель, яка буде використовуватися для виконання передбачень;
- X_test: незалежні змінні тестового дата сету;
- y_test: істинні мітки класів тестового дата сету.

Всередині функції визначається список sentiment_classes, який містить назви класифікаційних груп для сентименту: негатичні, нейтральні, позитивні. Використовуючи метод predict моделі, функція отримує передбачення y_pred для тестових даних X_test. За допомогою функції confusion_matrix з бібліотеки scikit-learn обчислюється матриця невідповідності cm. Для цього спочатку використовується np.argmax для перетворення імовірностей, повернутих моделлю, та істинних міток у вектори індексів максимальних значень (тобто перетворення one-hot encoded міток назад у їх класові індекси). Використовуючи бібліотеку matplotlib та seaborn, створюється візуалізація матриці невідповідності. sns.heatmap використовується для створення теплової карти:

- cm: матриця невідповідності, яка буде візуалізована;
- cmap=plt.cm.Blues: колірна схема для теплової карти;
- annot=True: вказує, що числа повинні бути анотовані всередині квадратів матриці;
- fmt='d': формат чисел як цілих;
- xticklabels та yticklabels: мітки для осей X та Y відповідно з використанням назв класів сентименту.

Далі задаються назва графіка та мітки для осей. Функція

plot_confusion_matrix викликається з поточною моделлю та тестовими даними для створення та відображення матриці невідповідності. Цей код використовується для аналізу того, як модель класифікує різні класи, та ідентифікації можливих слабких місць у її передбаченнях, наприклад, які класи часто плутаються між собою.

Загалом описана програмна реалізація представляє собою комплексний підхід до побудови та застосування нейронної мережі для класифікації емоційного забарвлення тексту, як-от відгуки або коментарі в соціальних мережах. Він охоплює всі кроки від підготовки даних до валідації та тестування моделі. Ключові етапи включають підготовку даних, де вхідні дані розбиваються на тренувальні, перевірочні та тестові набори. Це гарантує, що модель може бути навчена та перевірена на різноманітних даних для забезпечення її узагальнюючої здатності. Модель включає декілька шарів для обробки тексту, включаючи шар вбудування для перетворення тексту у вектори, згорткові шари для вилучення особливостей, шари максимального пулінгу для зменшення розміру даних та двонаправлений LSTM для збереження контексту в послідовностях. Шар випадкового вимкнення додається для запобігання перенавчанню моделі.

Для тренування моделі використовується оптимізатор SGD зі зниженням швидкості навчання та моментом, щоб забезпечити стабільне та ефективне навчання. Фінальний шар моделі використовує функцію активації softmax для класифікації вхідних даних на позитивні, негативні та нейтральні категорії. Код також включає функції для візуалізації результатів за допомогою матриці невідповідності, яка показує ефективність моделі в класифікації, а також для виконання передбачень на нових даних. Завантаження збереженої моделі та робота з нею для прогнозування сентименту в тексті дозволяє легко використовувати навчену модель без необхідності повторного її тренування.

В цілому, цей код представляє собою повний потік роботи для застосування машинного навчання до задачі NLP.

3.3 Практичні рекомендації

Застосування глибинного навчання в області обробки природної мови (NLP) охоплює не тільки розуміння теоретичних аспектів, але й вимагає практичних навичок у роботі з даними, моделями та інструментарієм. Практичні аспекти починаються з підготовки даних, вибору архітектури та гіперпараметрів, а також охоплюють процеси тренування та перевірки моделі. Особлива увага приділяється методам моніторингу та оцінювання продуктивності моделі, які є важливими для забезпечення її надійності та точності. Також розглядаються методи масштабування та оптимізації моделі для ефективної роботи з великими наборами даних, а також стратегії для вирішення типових проблем, що можуть виникнути під час впровадження моделі у виробництво. Включені практичні поради щодо розгортання моделей, їх тонкого налаштування та вдосконалення, що робить цей підрозділ цінним ресурсом для досягнення високої ефективності NLP-моделей.

Оптимізація гіперпараметрів є ключовою частиною процесу побудови ефективних моделей машинного навчання. Цей етап включає вибір і налаштування параметрів моделі, які не навчаються безпосередньо з даних, але мають важливий вплив на процес навчання та продуктивність моделі. Важливо звернути увагу на такі параметри, як розмір вбудування (embedding size), який визначає розмірність векторів для представлення слів, кількість епох тренування, що впливає на кількість разів, яку модель буде бачити тренувальні дані, швидкість навчання, яка керує темпом оновлення ваг моделі, та розмір пакету (batch size), який впливає на кількість даних, які модель обробляє за один крок. Для знаходження оптимальних гіперпараметрів можна використовувати методи перебору параметрів, такі як grid search, де перевіряються всі комбінації значень гіперпараметрів, або random search, який дозволяє випадково вибирати комбінації параметрів, що може бути ефективнішим у великих просторах пошуку. Успішна оптимізація гіперпараметрів може значно покращити здатність моделі до узагальнення та зменшити ризик перенавчання. Це, у свою чергу, призведе до більш точних та надійних результатів

у практичних застосуваннях моделі.

Ефективна підготовка даних є фундаментальною для успішного застосування моделей машинного навчання, особливо в області обробки природної мови. Процес підготовки даних охоплює декілька важливих етапів, які забезпечують високу якість вхідних даних для моделі.

1. Очищення Тексту та Токенізація: Початковий етап підготовки включає очищення тексту від небажаних символів та шуму, таких як знаки пунктуації та спеціальні символи. Токенізація, процес розбиття тексту на слова або фрази, дозволяє подальше їх оброблення.

2. Видалення стоп-слів, стемінг та лематизація: Видалення стоп-слів (слів, які не несуть значущої інформації) допомагає зменшити розмір навчальних даних. Стемінг та лематизація, процеси зведення слів до їх основних форм, підвищують консистентність даних та сприяють кращому розумінню контексту.

3. Балансування набору даних: Балансування даних, процес забезпечення рівномірного розподілу класів у наборі даних, є важливим для уникнення упередженості моделі. Незбалансовані набори даних можуть призвести до упередженості моделі на користь переважаючих класів, що може негативно вплинути на здатність моделі узагальнювати.

Ретельна підготовка даних є критично важливою для забезпечення того, що модель машинного навчання буде тренуватися на якісних, репрезентативних даних, що є необхідною умовою для досягнення високої точності та надійності у реальних застосуваннях.

Процес тренування та перевірки моделей машинного навчання відіграє вирішальну роль у визначенні їхньої ефективності та надійності. Важливо ретельно підійти до розподілу даних та використання технік перевірки для досягнення оптимальних результатів. Основною стратегією є поділ даних на тренувальні, перевірочні та тестові набори. Тренувальний набір використовується для навчання моделі, перевірочний для налаштування гіперпараметрів та оцінки моделі під час тренування, а тестовий набір для остаточної оцінки ефективності моделі після

тренування. Правильний розподіл забезпечує, що модель буде оцінена на непересічних даних, що дозволяє зробити надійні висновки про її продуктивність. Техніка cross-validation, особливо k-fold cross-validation, є ефективним способом для оцінки моделі. При цьому тренувальний набір даних ділиться на k окремих частин (folds) (рис. 3.4), і модель тренується та оцінюється k разів, кожного разу використовуючи різну частину даних як валідаційний набір [27].

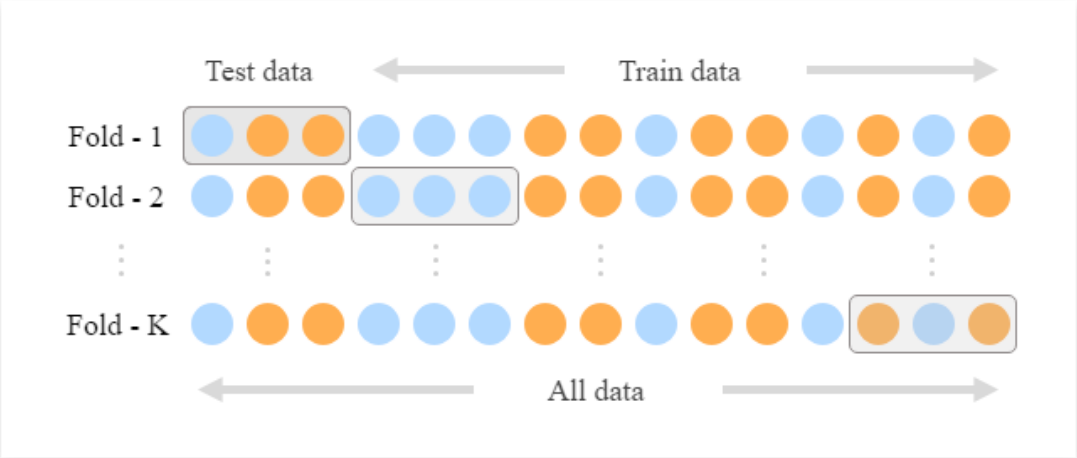


Рис. 3.4 K-Fold Cross-Validation [27]

Це дозволяє оцінити стійкість моделі до різних наборів даних та зменшує ризик перенавчання. Ретельне виконання цих процесів є ключовим для розробки моделей, які не тільки показують високу продуктивність на тренувальних даних, але й добре узагальнюють результати на нових, раніше невідомих даних. Систематичне тренування та перевірка забезпечують важливу перевірку та забезпечення якості протягом усього процесу розробки моделі.

Правильний вибір архітектури нейронної мережі є фундаментальним для успішної реалізації будь-якого проекту машинного навчання. Ефективна архітектура моделі не тільки впливає на точність та швидкість навчання, але й визначає, наскільки добре модель зможе узагальнювати свої знання на нових даних. Розробка архітектури нейронної мережі включає вибір типів шарів та їх конфігурацію. Важливо розуміти функції різних шарів, таких як згорткові шари для вилучення ознак, пулінг-шари для зменшення розміру вхідних даних, щільні шари

для класифікації та шари випадкового вимкнення для запобігання перенавчанню.

Залежно від завдання, може бути вибрано різні типи нейронних мереж. Наприклад, згорткові нейронні мережі (CNN) ефективні для обробки зображень, рекурентні нейронні мережі (RNN) і довгострокові короткочасні мережі пам’яті (LSTM) використовуються для обробки послідовностей даних, таких як текст або часові ряди. Двонаправлені LSTM (Bidirectional LSTM) можуть бути особливо корисними для задач NLP, оскільки вони обробляють дані в обох напрямках, забезпечуючи більший контекст. Ретельний вибір архітектури та типу мережі забезпечує фундамент для побудови потужних та ефективних моделей, здатних досягати високих результатів у вирішенні різноманітних завдань машинного навчання. Адекватне структурування архітектури враховує особливості даних та задачі, забезпечуючи оптимальне використання обчислювальних ресурсів та досягнення кращих результатів.

Моніторинг та оцінка продуктивності є важливими етапами в процесі розробки моделей машинного навчання. Вони забезпечують необхідну зворотну інформацію для оцінки та налаштування моделей, дозволяючи розробникам зрозуміти та вдосконалити їхні моделі. Моніторинг процесу тренування моделі може бути виконаний за допомогою візуалізації кривих навчання та втрат. Це включає відстеження змін у точності та втратах моделі протягом різних епох тренування. Візуалізація цих кривих допомагає ідентифікувати проблеми, такі як перенавчання або недостатнього навчання, та робить процес налаштування моделі більш інтуїтивно зрозумілим. Матриця невідповідності та інші метрики продуктивності, такі як точність, повнота та F1-міра, є ключовими для оцінки якості моделей класифікації. Матриця невідповідності детально відображає відносини між фактичними та передбаченими класами, дозволяючи розробникам зрозуміти типи та причини помилок моделі.

Використання цих метрик дозволяє робити обґрунтовані рішення щодо подальшого вдосконалення моделі. Систематичний моніторинг та оцінка продуктивності є важливими для забезпечення, що розроблені моделі не тільки

досягають високої точності на тренувальних даних, але й ефективно узагальнюють свої передбачення на нових, невиданих даних. Це також допомагає забезпечити, що модель буде надійною та ефективною у реальних застосуваннях.

Ефективне використання та впровадження моделей машинного навчання після їх тренування є ключовим етапом, що забезпечує перехід від теоретичних досліджень до практичного застосування. Завершення процесу тренування відкриває шлях для реалізації різних післятренувальних дій. Одним із основних кроків є використання навченої моделі для вирішення реальних завдань та отримання передбачень на нових даних. Це може включати інтеграцію моделі в додатки, веб-сервіси або інші системи, де її можна використовувати для автоматичного аналізу даних, розпізнавання образів, прогнозування трендів тощо. Важливою частиною післятренувального процесу є також збереження та розгортання моделі. Збереження моделі дозволяє використовувати її в майбутньому без необхідності повторного тренування. Розгортання може включати реалізацію моделі в хмарному середовищі, використання контейнеризації для забезпечення легкого та ефективного розповсюдження або використання спеціалізованих серверів для обробки запитів у реальному часі. Виконання цих післятренувальних дій забезпечує, що модель не лише досягає технічного успіху у лабораторних умовах, але й ефективно функціонує у реальних застосуваннях, вносячи значний вклад у вирішення практичних завдань.

Підходи до розширення та масштабування моделей машинного навчання відіграють важливу роль у забезпеченні їх ефективності та адаптивності до зростаючих обсягів даних та складності задач. Розширення моделі для роботи з більшими наборами даних або складнішими завданнями може включати оптимізацію архітектури та збільшення масштабів її шарів. Це може означати введення додаткових згорткових або рекурентних шарів, збільшення кількості нейронів у існуючих шарах, або внесення змін у структуру мережі для кращого засвоєння складних патернів у даних. Важливо також звертати увагу на ефективність обчислень, щоб модель могла ефективно обробляти збільшені обсяги

даних. Для прискорення процесу тренування моделі, особливо коли мова йде про великі набори даних або складні архітектури, використання хмарних обчислень та графічних процесорів (GPU) є ефективним рішенням. Хмарні платформи, такі як AWS, Google Cloud або Microsoft Azure, пропонують потужні обчислювальні ресурси та гнучкість у масштабуванні, дозволяючи значно зменшити час тренування. GPU особливо ефективні у паралельній обробці даних, що робить їх ідеальними для тренування глибоких нейронних мереж. Розширення та масштабування моделей є ключовим для підтримки їх ефективності в умовах швидкого росту обсягів даних та складності задач, з якими стикаються багато сучасних застосувань машинного навчання. Ці процеси забезпечують, що моделі залишаються конкурентоспроможними та відповідають сучасним вимогам обробки великих даних та їх аналізу.

ВИСНОВКИ

Основні результати роботи полягають у наступному:

1. Проведено огляд та аналіз джерел, пов’язаних зі штучними нейронними мережами, а також було розглянуто складові рекурентних нейронних мереж, методи навчання, тестування моделей, які можна використовувати в задачах визначення сентименту. Проаналізовані існуючі методики обробки та аналізу природної мови та визначені особливості обробки повідомлень з соцмереж.
2. Для підготовки даних, отриманих з текстів повідомлень в соцмережах була використана спеціальна багатошарова комбінація методів передобробки даних, таких як: лематизація, обробка специфічних символів, емодзі, видалення стоп-слів, видалення хештегів та посилань, збереження пов’язаних структур. Було використано семантичні векторизатори слів для перетворення тексту у векторне представлення, такі як GloVe, TF-IDF.
3. Розроблено методику вилучення емотивної складової інформації з тексту повідомлень в соцмережах, що базується на спеціальній комбінації методів передобробки даних, векторизації даних та використанні двонаправлених рекурентних нейронних мереж.
4. В ході дослідження ефективності методики були порівнянні як традиційні підходи (метод опорних векторів, логістична регресія, алгоритм Байєса), так й підходи на основі глибокого навчання (рекурентні нейронні мережі, довга короткочасна пам’ять). Як результат, комбінована модель на основі двонаправлених LSTM та алгоритму векторизації слів GloVe продемонструвала найкращі результати у завданні аналізу емоційної складової текстів з соцмереж, продемонструвавши кінцевий показник точності 0,892.

ПЕРЕЛІК ПОСИЛАНЬ

1. Попова. О.В. Емотивність як прагматичний фактор комунікації. Січень 2010 року.[Електронний ресурс]. Режим доступу: https://www.researchgate.net/publication/277851357_Emotivnist_ak_pragmaticnij_faktor_komunikacii (дата звернення 20.11.2023)

2. Семенова О. В., Горлівський інститут іноземних мов. Донбаського державного педагогічного університету. ЕМОТИВНИЙ КОМПОНЕНТ ЯК ПРАГМАТИЧНИЙ СКЛАДНИК РЕКЛАМНОГО ТЕКСТУ.

3. Oh Chong, Kumar Savan. Association for Information Systems AIS Electronic Library (AISeL) How Trump won: The Role of Social Media Sentiment in Political Elections. [Електронний ресурс]. Режим доступу: https://core.ac.uk/download/pdf/301372818.pdf (дата звернення 20.11.2023)

4. Sancheng Peng a, Lihong Cao b, Yongmei Zhou c, Zhouhao Ouyang d, Aimin Yang e, Xinguang Li a, Weijia Jia f, Shui Yu g. Digital Communications and Networks. A survey on deep learning for textual emotion analysis in social networks.[Електронний ресурс]. Режим доступу: sciencedirect.com/science/article/pii/S2352864821000833 (дата звернення 20.11.2023)

5. SpringerLink [Електронний ресурс]. A review on sentiment analysis and emotion detection from text. Режим доступу: link.springer.com/article/10.1007/s13278-021-00776-6 (дата звернення 20.11.2023)

6. Quora [Електронний ресурс]. How does a recurrent neural network learn the correct order of words?. Режим доступу: https://www.quora.com/How-does-a-recurrent-neural-network-learn-the-correct-order-of-words (дата звернення 20.11.2023)

7. Medium [Електронний ресурс]. Laurenflynn. Comparing Sentiment Analysis Dictionaries in R. Режим доступу: https://medium.com/@laurenflynn1211/comparing-sentiment-analysis-dictionaries-in-r-c695fca64326 (дата звернення 20.11.2023)

8. Kaggle [Електронний ресурс]. NIKIT PERIWAL. Twitter Sentiment Analysis for Beginners. Режим доступу: https://www.kaggle.com/code/stoicstatic/twitter-sentiment-analysis-for-beginners/notebook (дата звернення 19.11.2023)

9. Docs Python [Електронний ресурс]. pickle — Python object serialization. Режим доступу: https://docs.python.org/3/library/pickle.html (дата звернення 19.11.2023).

10. Kaggle [Електронний ресурс]. PAOLO RIPAMONTI. Twitter Sentiment Analysis. Режим доступу: https://www.kaggle.com/code/paoloripamonti/twitter-sentiment-analysis/notebook (дата звернення 15.11.2023)

11. Kaggle [Електронний ресурс]. ADEM HAMIC. Emotion Recognition Using LSTM. Режим доступу: https://www.kaggle.com/code/ademhph/emotion-recognition-using-lstm#4-%7C-Create-model-(LSTM) (дата звернення 15.11.2023)

12. Міжнародна науково-технічна Internet-конференція. «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами». 22 листопада 2018 рік.

13. Шеремет І.О. Садовой О.В. Метод опорних векторів (SVM). Режим доступу: https://www.dstu.dp.ua/Portal/Data/74/72/3st13-17.pdf (дата звернення 08.11.2023)

14. IBM [Електронний ресурс]. What are recurrent neural networks? . Режим доступу: https://www.ibm.com/topics/recurrent-neural-networks (дата звернення 08.11.2023)

15. Medium [Електронний ресурс]. BERT Explained: State of the art language model for NLP . Режим доступу: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270 (дата звернення 11.11.2023)

16. Medium [Електронний ресурс]. Chat GPT and GPT 3 Detailed Architecture Study-Deep NLP Horse. Режим доступу: https://medium.com/nerd-for-tech/gpt3-and-chat-gpt-detailed-architecture-study-deep-nlp-horse-db3af9de8a5d (дата звернення 19.11.2023)

17. Medium [Електронний ресурс]. Ensemble methods: bagging, boosting and stacking. Режим доступу: https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205 (дата звернення 11.11.2023)

18. Wang Shaoxiu,Yonghua Zhu ,Wenjing Gao,Meng Cao, Mengyao Li Emotion-

Semantic-Enhanced Bidirectional LSTM with Multi-Head Attention Mechanism for Microblog Sentiment Analysis. Shanghai University. Режим доступу: https://www.mdpi.com/2078-2489/11/5/280 (дата звернення 11.11.2023)

19. Analytics Vidhya [Електронний ресурс]. Naive Bayes Classifier Explained: Applications and Practice Problems of Naive Bayes Classifier. Режим доступу: analyticsvidhya.com/blog/2017/09/naive-bayes-explained/ (дата звернення 17.11.2023)

20. Kaggle [Електронний ресурс]. The Math behind Linear SVC Classifier. Режим доступу: https://www.kaggle.com/code/xingewang/the-math-behind-linear-svc-classifier (дата звернення 02.12.2023)

21. RealPython [Електронний ресурс]. Stochastic gradient descent algorithm with python and numpy. Режим доступу: https://realpython.com/gradient-descent-algorithm-python/ (дата звернення 02.12.2023)

22. geeksforgeeks [Електронний ресурс]. Difference between batch gradient descent and stochastic gradient descent. Режим доступу: https://www.geeksforgeeks.org/difference-between-batch-gradient-descent-and-stochastic-gradient-descent/ (дата звернення 11.11.2023)

23. Keras [Електронний ресурс]. Introducing Keras 3.0. Режим доступу: https://keras.io/keras_3/ (дата звернення 11.11.2023)

24. TensorFlow [Електронний ресурс]. CREATE PRODUCTION-GRADE MACHINE LEARNING MODELS WITH TENSORFLOW. Режим доступу: https://www.tensorflow.org/ (дата звернення 12.11.2023)

25. Microsoft [Електронний ресурс]. CONVERT WORD TO VECTOR COMPONENT. Режим доступу: https://learn.microsoft.com/uk-ua/azure/machine-learning/component-reference/convert-word-to-vector?view=azureml-api-2 (дата звернення 12.11.2023)

26 Medium [Електронний ресурс]. Oleg Dubetsky. Обробка природної мови (NLP) у Python. Режим доступу: medium.com/@oleg-dubetcky (дата звернення 12.11.2023)

27. Philschmid [Електронний ресурс]. K-fold as cross-validation with a bert text-

classification. Режим доступу: https://www.philschmid.de/k-fold-as-cross-validation-with-a-bert-text-classification-example (дата звернення 12.11.2023).

28. Christopher Olah. Understanding LSTM Networks. [Електронний ресурс]. Режим доступу: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

29. Sebastian Raschka. LogisticRegression: A binary classifier, 2014. [Електронний ресурс]. Режим доступу: https://rasbt.github.io/mlxtend/user_guide/classifier/LogisticRegression

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

(Презентація)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Магістерська робота

«Розробка методики вилучення емотивної складової інформації з тексту повідомлень в соцмережах на основі рекурентних нейронних мереж»

Виконав: студент групи ПДМ-62 Бондаренко Микита Олегович

Керівник: к.т.н., доц., доцент кафедри ІПЗ Золотухіна Оксана Анатоліївна

Київ - 2024

МЕТА, ОБ'ЄКТА ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: оптимізувати процес вилучення емотивної інформації з текстових повідомлень в соціальних мережах шляхом застосування рекурентних нейронних мереж.

Об'єкт дослідження: процес вилучення емотивної складової інформації з текстових повідомлень в соціальних мережах.

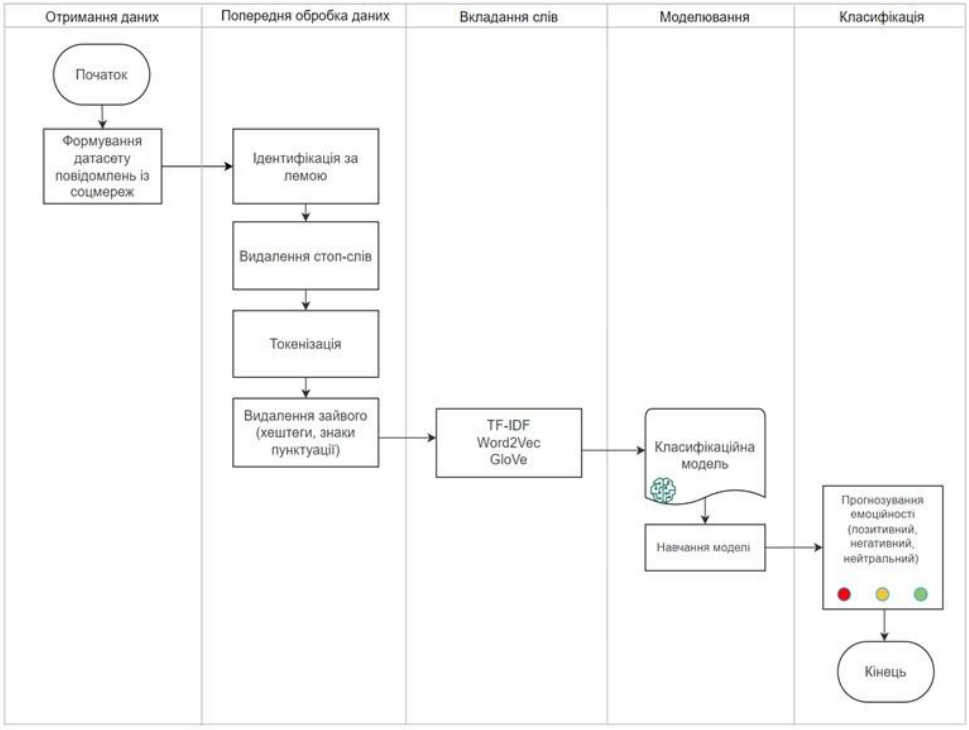
Предмет дослідження: методи обробки текстових повідомлень для вилучення з них емотивної складової інформації.

АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ КЛАСИФІКАЦІЇ ЕМОТИВНОЇ СКЛАДОВОЇ ПОВІДОМЛЕНЬ

	Опис	Переваги	Недоліки
Логістична регресія	Дискримінаційна модель, має низьке зміщення та вищу дисперсію	Простота та ефективність для бінарної класифікації	Не здатний узагальнювати складні взаємодії. Не враховує загальний контекст
Наївний Баєсів класифікатор	Генеративна модель, лінійний класифікатор, який використовує теорему Байєса	Швидкий, простий, добре працює для простих завдань	Не враховує контекст та взаємодії між словами
Рекурентні нейронні мережі (RNN)	Фіксує інформацію з попередніх часових кроків,	Добре підходять для завдань, де минула інформація суттєво впливає на майбутнє	Страждають від зникаючих, що ускладнює охоплення довгострокових залежностей.
LSTM (Довга короткочасна пам'ять)	Вводять механізм "шлюзів", який дозволяє мережі дізнаватися, коли зберігати або забувати інформацію з попередніх часових кроків.	Ефективні для довгих залежностей в тексті	Дорогі з точки зору обчислювальних ресурсів і вимагають великої кількості даних для ефективного навчання.

3

АЛГОРИТМ ВИЛУЧЕННЯ ЕМОТИВНОЇ СКЛАДОВОЇ ІНФОРМАЦІЇ З ТЕКСТУ ПОВІДОМЛЕНЬ В СОЦМЕРЕЖАХ



4

ЕТАПИ ПЕРЕДОБРОБКИ ДАНИХ

1. Видалення хештегів
2. Видалення знаків пунктуації
3. Видалення стоп-слів
4. Фільтрація цифр
5. Лематизація
6. Обробка специфічних символів
7. Обробка емодзі
8. Заміна URL
9. Токенізація
10. Кодування

5

МАТЕМАТИЧНА МОДЕЛЬ LSTM

Forget Gate:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

Input Gate:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

Оновлення Cell State:

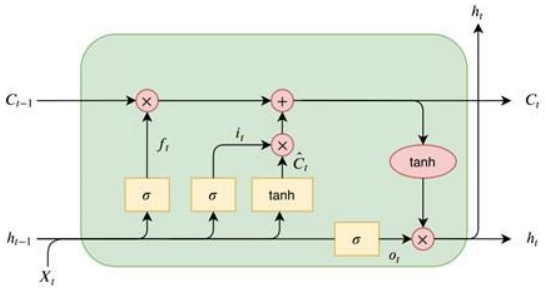
$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

Output Gate:

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

Оновлення Hidden State:

$$h_t = o_t \cdot \tanh(C_t)$$



Структура елементу рекурентної нейронної мережі LSTM.

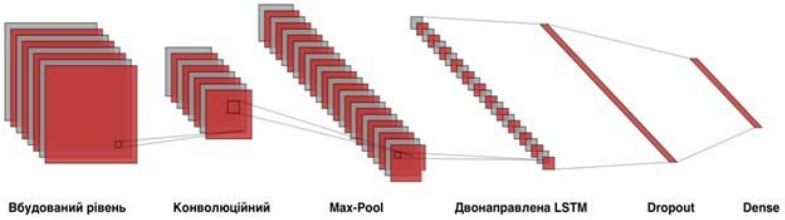
- $\mathbf{f_t}$ - вихід з лінійного шлюзу "забуття",
- $\mathbf{i_t}$ - вихід з вхідного шлюзу,
- $\tilde{\mathbf{C}}_\mathbf{t}$ - кандидат на оновлення пам'яті,
- $\mathbf{C_t}$ - оновлений стан пам'яті (cell state),
- $\mathbf{o_t}$ - вихід з вихідного шлюзу,
- x_t - вхідна послідовність у момент часу t
- $\mathbf{h_t}$ - оновлений прихований стан,
- $\mathbf{W_f}, \mathbf{W_i}, \mathbf{W_C}, \mathbf{W_o}$ - вагові матриці,
- $\mathbf{b_f}, \mathbf{b_i}, \mathbf{b_C}, \mathbf{b_o}$ - вектори зсуву,
- σ - функція сигмоїди,
- $\tanh$ - гіперболічний тангенс.

$$\sigma = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}$$

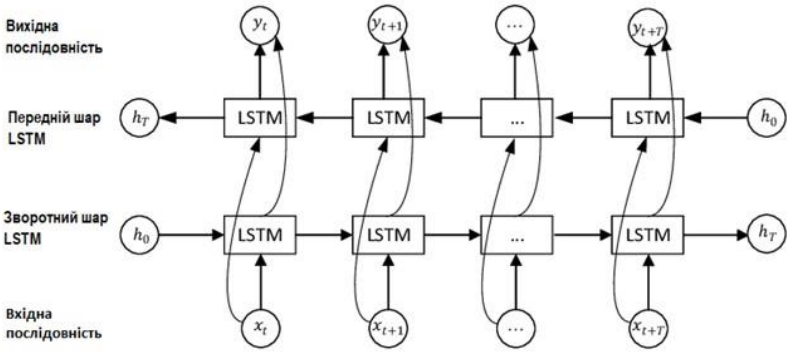
Функція сигмоїди (σ)

6

МОДИФІКОВАНА МОДЕЛЬ ГЛИБОКОГО НАВЧАННЯ



Архітектура модифікованої моделі глибоко навчання



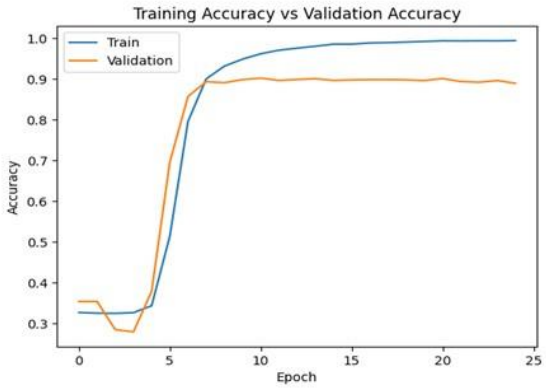
Діаграма структури шару двонаправленої LSTM

7

НАВЧАННЯ МОДЕЛІ

Загальні параметри навчання:

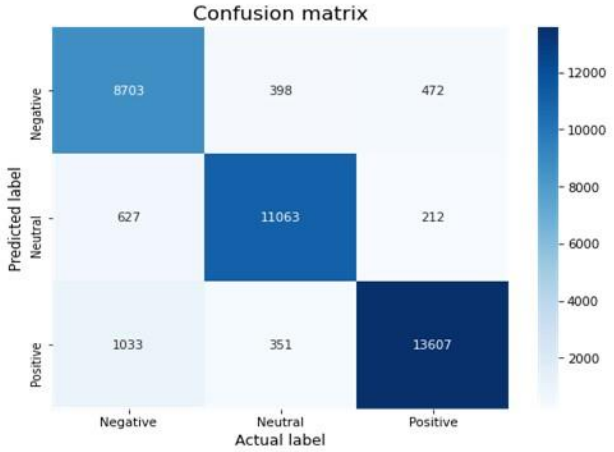
- Кількість епох: *20*
- Оптимізатор: алгоритм Адама
- Функція втрат: категоріальна крос-ентропія
- Розмір словника для вбудовування слів: *5000*
- Частка одиниць, які потрібно відкинути для лінійного перетворення вхідних даних (dropout): *0.2*



Графік валідаційної точності від епохи

8

МАТРИЦЯ НЕВІДПОВІДНОСТЕЙ



9

РЕЗУЛЬТАТИ

Метод	F1	Precision	Recall
Логістична регресія	0.830	0.828	0.822
Naive Bayes	0.803	0.812	0.798
LSTM + TD-IDF (*M1*)	0.853	0.860	0.854
LSTM + GloVe (*M2*)	0.869	0.881	0.863
Bi-LSTM + TD-IDF (*M3*)	0.871	0.868	0.879
Bi-LSTM + GloVe (*M4*)	**0.889**	**0.892**	**0.890**

10

ВИСНОВКИ

1. Проведено огляд та аналіз джерел, пов'язаних зі штучними нейронними мережами, а також було розглянуто складові рекурентних нейронних мереж, методи навчання, тестування моделей, які можна використовувати в задачах визначення сентименту. Проаналізовані існуючі методики обробки та аналізу природної мови та визначені особливості обробки повідомлень з соцмереж.
2. Для підготовки даних, отриманих з текстів повідомлень в соцмережах була використана спеціальна багатошарова комбінація методів передобробки даних, таких як: лематизація, обробка специфічних символів, емодзі, видалення стоп-слів, видалення хештегів та посилань, збереження пов'язаних структур. Було використано семантичні векторизатори слів для перетворення тексту у векторне представлення, такі як GloVe, TF-IDF.
3. Розроблено методику вилучення емотивної складової інформації з тексту повідомлень в соцмережах, що базується на спеціальній комбінації методів передобробки даних, векторизації даних та використанні двонаправлених рекурентних нейронних мереж.
4. В ході дослідження ефективності методики були порівнянні як традиційні підходи (метод опорних векторів, логістична регресія, алгоритм Байєса), так й підходи на основі глибокого навчання (рекурентні нейронні мережі, довга короткочасна пам'ять). Як результат, комбінована модель на основі двонаправлених LSTM та алгоритму векторизації слів GloVe продемонструвала найкращі результати у завданні аналізу емоційної складової текстів з соцмереж, продемонструвавши кінцевий показник точності 0,892.

11

ПУБЛІКАЦІЇ ТА АПРОБАЦІЯ РОБОТИ

Тези доповідей:

1. Бондаренко М.О. "Оптимізація процесу вилучення емотивної складової інформації з тексту повідомлень в мережах на основі двонаправлених рекурентних нейронних мереж". // Всеукраїнська науково-практична конференція «ТАК». – Луцьк: ДВНЗ «Донецький національний технічний університет», 2023. Подано до друку.

2. Бондаренко М.О. "Методи обробки природної мови (NLP) для аналізу емотивної складової" // Всеукраїнська конференція молодих вчених «Актуальні питання розвитку інформаційних технологій». – Дніпро: ДВНЗ «Приазовський державний технічний університет», 2023. Подано до друку.

12

ДЯКУЮ ЗА УВАГУ!

ДОДАТОК А

ФРАГМЕНТИ ЛІСТИНГУ КОДУ

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size=0.25, random_state=1)

from keras.models import Sequential
from keras.layers import Embedding, Conv1D, MaxPooling1D, Bidirectional, LSTM, Dense, Dropout
from keras.metrics import Precision, Recall
from keras.optimizers import SGD
from keras.optimizers import RMSprop
from keras import datasets
from sklearn.metrics import confusion_matrix
from keras.callbacks import LearningRateScheduler
from keras.callbacks import History

from keras import losses

sentiment_classes = ['Negative', 'Neutral', 'Positive']

vocab_size = 5000
embedding_size = 32
epochs=20
learning_rate = 0.1
decay_rate = learning_rate / epochs
momentum = 0.8

sgd = SGD(learning_rate=learning_rate, momentum=momentum, nesterov=False)
model= Sequential()
model.add(Embedding(vocab_size, embedding_size, input_length=max_len))
model.add(Conv1D(filters=32, kernel_size=3, padding='same', activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Bidirectional(LSTM(32, return_sequences=True)))
model.add(Dropout(0.4))
model.add(Dense(3, activation='softmax'))

def predict_class(text):
    max_len=50
    xt = tokenizer.texts_to_sequences(text)
    xt = pad_sequences(xt, padding='post', maxlen=max_len)
    yt = model.predict(xt).argmax(axis=1)

def plot_confusion_matrix(model, X_test, y_test):    # use model to do the prediction
    y_pred = model.predict(X_test)
    cm = confusion_matrix(np.argmax(np.array(y_test),axis=1), np.argmax(y_pred, axis=1))
    sns.heatmap(cm, cmap=plt.cm.Blues, annot=True, fmt='d',
                xticklabels=sentiment_classes,
                yticklabels=sentiment_classes)
plot_confusion_matrix(model, X_test, y_test)
```