Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Розробка системи виявлення кібербулінгу на основі методів обробки природної мови
(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-21-1
Тетяна Слюсаренко
(власне ім'я, прізвище)
Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект
(повна назва освітньої програми)
Керівник ас. Марія Погурська
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Олег ЗОЛОТУХІН
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

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

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Слюсаренко Тетяні Андріївні
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка системи виявлення кібербулінгу на основі методів обробки природної мови

затверджена наказом університету від 19 травня 20 25 р. № 378Ст
2. Термін подання студентом роботи до екзаменаційної комісії 17 червня 20 25 р.
3. Вихідні дані до роботи Науково-технічні публікації, дані Інтернет-джерел та відомих наукових проєктів щодо тематики дослідження, документація Python та бібліотек nltk, Pytorch і scikit-learn , набір даних Cyberbullying Classification для тренування та тестування системи

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі
2) Методи обробки природньої мови
3) Моделі для виявлення кібербулінгу та їх метрики
4) Програмна реалізація

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	19.05.2025	виконано
2	Аналіз предметної галузі	21.05.2025	виконано
3	Огляд методів обробки природньої мови	25.05.2025	виконано
4	Огляд моделей для виявлення кібербулінгу та	25.05.2025	виконано
	їх метрик		
5	Програмна реалізація	25.05.2025	виконано
6	Написання пояснювальної записки	26.05.2025	виконано
7	Перевірка на академічний плагіат	29.05.2025	виконано
8	Нормоконтроль	30.05.2025	виконано
9	Підготовка презентації та доповіді	07.06.2025	виконано
10	Попередній захист	10.06.2025	виконано
11	Рецензування	13.06.2025	виконано
12	Захист перед ЕК	17.06.2025	

Дата видачі завдання 19 травня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ ас. Марія Погурська
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 99 с., 27 рис., 5 дод., 28 джерел.

КІБЕРБУЛІНГ, КЛАСИФІКАЦІЯ ТЕКСТУ, МАШИННЕ НАВЧАННЯ, НАЇВНИЙ БАЙЄС, ОБРОБКА ПРИРОДНОЇ МОВИ, СОЦІАЛЬНІ МЕРЕЖІ, ШТУЧНИЙ ІНТЕЛЕКТ, LSTM, MLP, TF-IDF, WORD EMBEDDINGS.

Об’єкт дослідження – процес виявлення та класифікації проявів кібербулінгу в текстових повідомленнях, опублікованих у цифровому середовищі.

Предмет дослідження – методи та моделі обробки природної мови для автоматичного виявлення кібербулінгу в текстовому контенті соціальних мереж.

Мета роботи – розробка та дослідження ефективності моделей машинного навчання для автоматичного виявлення кібербулінгу в текстах.

Методи дослідження – аналіз наукової літератури з питань виявлення кібербулінгу за допомогою NLP та машинного навчання; побудова та навчання моделей машинного навчання та глибокого навчання; використання методів векторизації тексту; оцінка ефективності моделей за допомогою метрик точності, повноти, F1-міри.

У роботі здійснено як теоретичне, так і практичне дослідження проблеми кібербулінгу. Проаналізовано його природу, форми, наслідки та сучасні підходи до автоматичного виявлення агресії в онлайн-середовищі. Проведено огляд наукових праць, законодавчих ініціатив і технічних рішень, що використовуються у цій сфері. У практичній частині розроблено та порівняно три моделі різної складності для виявлення кібербулінгу на основі методів обробки природної мови. Найкраща з реалізованих моделей досягла точності 92,8%, що свідчить про перспективність обраних підходів.

ABSTRACT

Bachelor’s thesis contains: 99 pp., 27 fig., 5 ann., 28 references.

ARTIFICIAL INTELLIGENCE, CYBERBULLYING, LSTM, MACHINE LEARNING, MLP, NAIVE BAYES, NATURAL LANGUAGE PROCESSING, SOCIAL MEDIA, TEXT CLASSIFICATION, TF-IDF, WORD EMBEDDINGS.

The object of research is the process of detecting and classifying cyberbullying in text messages published in the digital environment.

The subject of the study is methods and models of natural language processing for the automatic detection of cyberbullying in textual content of social networks.

The purpose is to develop and study the effectiveness of machine learning models for the automatic detection of cyberbullying in texts.

Research methods are analysis of scientific literature on cyberbullying detection using NLP and machine learning; building and training machine learning and deep learning models; using text vectorisation methods; evaluating the effectiveness of models using accuracy, completeness, and F1-measure metrics.

The paper conducts both a theoretical and practical study of the problem of cyberbullying. Its nature, forms, consequences and modern approaches to automatic detection of aggression in the online environment are analysed. A review of scientific papers, legislative initiatives and technical solutions used in this area are carried out. In the practical part, three models of varying complexity for detecting cyberbullying based on natural language processing methods are developed and compared. The best of the implemented models achieved an accuracy of 92.8%, which indicates the prospects of the selected approaches.

ЗМІСТ

Вступ 8
1 Аналіз предметної галузі 9
1.1 Сучасний стан проблеми кібербулінгу 9
1.1.1 Причини та наслідки кібербулінгу 14
1.1.2 Вплив технологій на поширення кібербулінгу 17
1.1.3 Міжнародний досвід боротьби з кібербулінгом 19
1.2 Огляд існуючих рішень 21
1.3 Постановка задачі 25
2 Методи обробки природньої мови 27
2.1. Основи обробки природної мови 27
2.1.1 Основні задачі обробки тексту 28
2.1.2 Особливості обробки неформального тексту 30
2.2. Попередня обробка текстових даних 31
2.2.1 Токенізація 32
2.2.2 Стемінг та лематизація 34
2.2.3 Видалення стоп-слів 35
2.2.4 Нижній регістр, пунктуація, очищення 36
2.3. Векторизація тексту 37
2.3.1 Bag of Words 37
2.3.2 TF-IDF 38
2.3.3 Word Embeddings (Word2Vec, GloVe, FastText) 40
3 Моделі для виявлення кібербулінгу та їх метрики 42
3.1 Класичні моделі машинного навчання 42
3.1.1 Логістична регресія 42
3.1.2 Наївний байєсівський класифікатор 44
3.1.3 Дерева рішень 45
3.2 Нейронні мережі для обробки тексту 47
3.2.1 Базові feed-forward мережі 48

3.2.2 Рекурентні нейронні мережі 49
3.2.3 Довготривала короткочасна пам’ять 51
3.3 Моделі на основі трансформерів 52
3.4 Метрики якості моделей 54
3.4.1 Accuracy, recall, precision, F1-міра 54
3.4.2 Матриця неточностей 57
3.4.3 Особливості оцінювання багатокласової класифікації 57
3.5 Вибір моделей для задачі розпізнавання кібербулінгу 58
4 Програмна реалізація 60
4.1 Інструменти та бібліотеки 60
4.2 Опис набору даних 61
4.3 Попередня обробка даних 62
4.4 Векторизація тексту 64
4.4.1 TF-IDF 64
4.4.2 Word2Vec 65
4.5 Побудова моделей 67
4.5.1 Наївний Байєс 67
4.5.2 Багатошарова лінійна мережа 67
4.5.3 LSTM мережа з механізмом уваги 72
4.6 Оцінка ефективності моделей 74
Висновки 83
Перелік джерел посилання 85
Додаток А Функція для очистки тексту 89
Додаток Б Створення та навчання багатошарової лінійної мережі 91
Додаток В Архітектура LSTM мережі з механізмом уваги 94
Додаток Г Створення та навчання LSTM мережі з механізмом уваги 96
Додаток Д Відомість кваліфікаційної роботи 99

ВСТУП

У сучасному цифровому суспільстві кібербулінг став однією з найпоширеніших форм агресії, що негативно впливає на психічне здоров'я та соціальне благополуччя підлітків і молоді. Згідно з дослідженням BrightPath Behavioral Health, у 2023 році 26,5% підлітків у США повідомили про досвід кібербулінгу, що свідчить про зростання цієї проблеми порівняно з попередніми роками [1].

Зростання популярності соціальних мереж, месенджерів та інших онлайн-платформ створює нові можливості для спілкування, але водночас відкриває простір для агресивної поведінки, яка часто залишається непоміченою або безкарною. Анонімність в інтернеті та швидкість поширення інформації ускладнюють виявлення та протидію кібербулінгу традиційними методами.

У відповідь на ці виклики, наукова спільнота активно досліджує можливості застосування методів обробки природної мови (NLP) та машинного навчання для автоматичного виявлення та класифікації проявів кібербулінгу. Такі підходи дозволяють аналізувати великі обсяги текстових даних з метою ідентифікації образливих, принизливих або загрозливих висловлювань. Наприклад, дослідження, опубліковане в журналі ScienceDirect, демонструє ефективність поєднання NLP та машинного навчання у виявленні кібербулінгу в соціальних мережах [2].

Розробка систем, здатних автоматично виявляти кібербулінг, має широкі перспективи застосування. Такі системи можуть бути інтегровані в платформи соціальних мереж, освітні онлайн-середовища, форуми та інші цифрові платформи для своєчасного виявлення та запобігання агресивній поведінці. Це сприятиме створенню безпечнішого онлайн-простору та зменшенню негативного впливу кібербулінгу на користувачів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

Інтернет став невід’ємною частиною нашого життя. Соціальні мережі, месенджери, форуми, ігрові чати – це середовище, де люди спілкуються, обмінюються думками, підтримують один одного. Але, як і в реальному житті, в цифровому просторі теж існує місце для агресії. Одним із найпоширеніших її проявів є кібербулінг – сучасна форма цькування, яка набуває все більшої актуальності, особливо серед молоді.

1.1 Сучасний стан проблеми кібербулінгу

Кібербулінг – це навмисне агресивне переслідування чи приниження людини з використанням цифрових технологій. Найчастіше це відбувається через соціальні мережі, месенджери, електронну пошту, коментарі під постами, анонімні чати тощо. На відміну від звичайного булінгу, кібербулінг може тривати 24/7, не прив’язаний до конкретного місця, наприклад, школи чи роботи, і часто здійснюється анонімно. Людина може прокинутися зранку, відкрити телефон і побачити десятки образливих повідомлень – навіть якщо вчора все було тихо.

Кібербулінг проявляється в багатьох формах, і важливо розуміти, як саме він може виглядати, бо іноді його складно розпізнати одразу. Часто це не відкриті погрози, а більш завуальовані дії, які завдають не меншої шкоди.

Одна з найпоширеніших форм – образливі коментарі або повідомлення. Це може бути пряме приниження, знецінення, насмішки або навіть загрози. Людину можуть ображати через зовнішність, національність, гендер, сексуальну орієнтацію або особливості поведінки. Особливо боляче, коли такі повідомлення з’являються публічно – під постами або в чатах з великою кількістю людей.

Дуже поширеним є також розповсюдження чуток або особистої інформації. Це може бути «злив» особистих фото без згоди, поширення

скрінів приватних переписок або навіть вигадування компромату. Такі дії завдають серйозного удару по репутації людини, особливо в замкнутих онлайн-спільнотах, де «сарафанне радіо» працює блискавично.

Ще одна форма – доксинг. Це публікація особистих даних без дозволу людини, часто з метою залякування чи підбурення інших до атак. Хоча цей термін більше відомий у хакерських або політичних колах, насправді він все частіше трапляється і серед підлітків.

Кіберсталкінг – це тривале онлайн-переслідування. Людина може отримувати десятки повідомлень, навіть якщо не відповідає. Її «моніторять», лайкають старі пости, слідкують за діями, коментують будь-який її контент. І хоча це виглядає не завжди агресивно, таке переслідування викликає страх і напругу.

Форма, яку іноді недооцінюють, – ізоляція або виключення з онлайн-спільноти. Людину навмисно не додають до групи в месенджері, не позначають на фото, ігнорують у чаті, хоча вона є «присутньою». Це може бути тонке, але болюче виключення з кола спілкування, яке також є формою цькування.

Окремо варто згадати кіберфлудинг – ситуацію, коли жертві надсилають велику кількість небажаних повідомлень або засипають негативними реакціями. Це може бути частиною організованої атаки, іноді навіть ботами, або діями невеликої групи.

Інколи булінг має форму імітації – коли хтось створює фейковий профіль, що копіює жертву, і від її імені публікує провокаційний контент. Це одна з найнебезпечніших форм, бо людина може навіть не знати, що її репутація страждає через підроблену активність.

Також важливо розуміти, що кібербулінг не завжди відбувається у формі прямого спілкування. Це можуть бути образливі меми, відео, фотожаби, в яких натякають або висміюють конкретну людину. І хоча ніхто прямо не називає імен, усім у групі або класі може бути зрозуміло, про кого йдеться.

Усі ці форми можуть поєднуватися між собою – наприклад, булінг починається з жарту, переходить у поширення фото, а потім – у кампанію з цькування в коментарях. І навіть якщо кожна дія окремо здається «незначною», разом вони утворюють серйозну психологічну атаку.

На рисунку 1.1 зображені найбільш поширені форми кібербулінгу серед підлітків у США станом на 2022 рік, згідно з даними Pew Research Center [1].

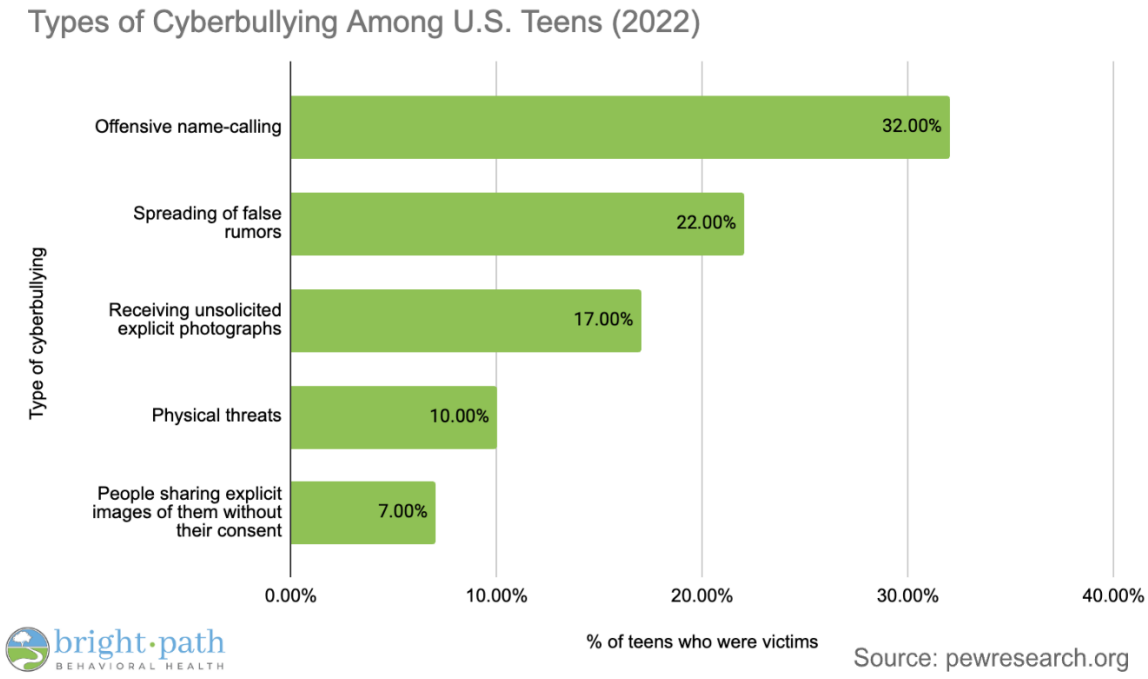

Рисунок 1.1 – Найпоширеніші форми кібербулінгу серед підлітків (2022)

Найчастіше підлітки стикаються з такими формами кібербулінгу:

– образливе прізвисько чи образи – 32% підлітків повідомили, що стали жертвами такого типу кібербулінгу і це є найпоширенішою формою цькування в інтернеті;

– поширення фейкових чуток – 22% респондентів зазначили, що про них розповсюджували неправдиву інформацію;

– отримання небажаних відвертих фотографій – 17% підлітків отримували контент сексуального характеру без згоди;

– фізичні погрози онлайн – 10% повідомили про погрози насильства в мережі;

– публікація інтимних фото без згоди – 7% стали жертвами поширення зображень без дозволу, що є формою серйозного цифрового насильства.

Ці дані демонструють, що вербальні форми цькування (словесні образи, чутки) залишаються найпоширенішими, однак сексуалізовані форми насильства та залякування також є серйозною проблемою, яка зачіпає значну частку молоді.

Найчастіше жертвами стають підлітки, які ще не навчилися до кінця розпізнавати токсичну поведінку. Вони активні в соціальних мережах, чутливі до думки інших і часто ще не мають сформованих навичок психологічного захисту. За даними ЮНІСЕФ (2023), 67% українських підлітків віком 11–17 років зіштовхувалися з булінгом у тій чи іншій формі, зокрема й онлайн-цькуванням [3].

У США, за результатами дослідження BrightPath Behavioral Health, 26,5% підлітків віком 13–17 років повідомили, що стали жертвами кібербулінгу протягом останнього місяця. Тобто проблема є масштабною як в Україні, так і в країнах з вищим рівнем цифрової освіти – просто її форми і масштаби можуть дещо різнитися [1].

Цікаво, що в США хлопці та дівчата повідомляють про кібербулінг майже з однаковою частотою: 23,7% дівчат і 21,9% хлопців зазначили, що ставали жертвами цькування в інтернеті [4]. Це свідчить про відносно рівномірний розподіл агресії між статями – ймовірно, з різними типами проявів: наприклад, дівчата частіше зазнають соціального приниження, а хлопці – прямих погроз або насмішок.

В Україні ж ситуація дещо інша: близько 80% постраждалих від кібербулінгу – дівчата, згідно з даними ЮНІСЕФ за 2020 рік [5]. Причини

цього можуть бути різні: від більшої відкритості дівчат у соцмережах до гендерних стереотипів, які ще досі залишаються сильними в українському суспільстві.

Підлітки мають різні погляди на проблему кібербулінгу, залежно від того, чи стикалися вони з ним особисто. Згідно з дослідженням Pew Research Center, підлітки, які стали жертвами кібербулінгу, значно критичніше оцінюють дії дорослих щодо боротьби з цією проблемою. Наприклад, 53% таких підлітків вважають, що обрані посадовці погано справляються з вирішенням проблеми онлайн-цькування, порівняно з 38% серед тих, хто не зазнавав цькування онлайн [6].

Згідно з даними Центру контролю та профілактики захворювань США, частота кібербулінгу серед підлітків суттєво варіюється залежно від етнічного походження (рисунок 1.2) [1]. Найбільш вразливою групою виявилися американські індіанці та корінні жителі Аляски – 21% опитаних представників цієї категорії повідомили, що стали жертвами онлайн-цькування.

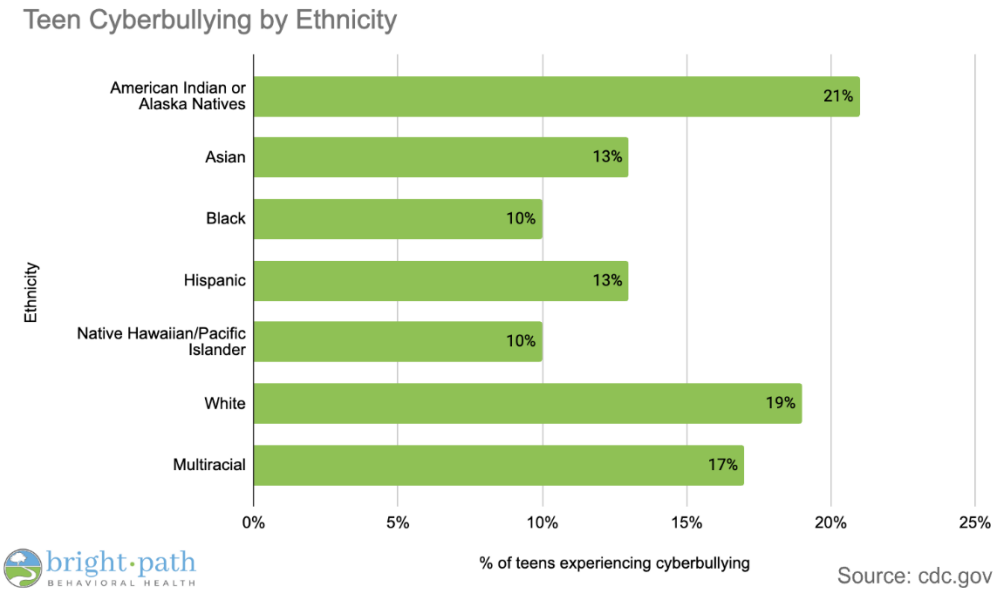

Рисунок 1.2 – Частота кібербулінгу серед підлітків в залежності від походження

На другому місці – білі підлітки, серед яких 19% мали досвід кібербулінгу. За ними йдуть мультирасові підлітки – 17% респондентів з цієї групи також постраждали від цифрового насильства.

Порівняно нижчі показники зафіксовано серед азіатських (13%), іспаномовних (13%), афроамериканських (10%) підлітків та представників народів Океанії (10%).

Ці дані свідчать, що кібербулінг – це не лише загальна проблема молоді, але й питання, яке має етнокультурну складову. Різний рівень уразливості вказує на необхідність врахування культурного контексту в політиках профілактики та підтримки жертв.

1.1.1 Причини та наслідки кібербулінгу

Коли говориться про кібербулінг, важливо не лише бачити його зовнішні прояви, а й намагатися зрозуміти, чому люди взагалі вдаються до подібної поведінки. У більшості випадків це не про «просто погану людину», а про цілий набір факторів – внутрішніх і зовнішніх, які штовхають людину на агресію в мережі.

Одна з найпоширеніших причин – відчуття безкарності та анонімності. В інтернеті легше сховатися за вигаданим ім’ям або фейковим акаунтом. Це створює ілюзію, що твої дії не матимуть наслідків. Про це сказали 17% людей, що брали участь в опитування університету Депол, Чикаго [7]. Людина, яка ніколи не наважилася б образити когось віч-на-віч, в онлайні почувається сміливішою. А якщо ще й ніхто не реагує на це як на проблему – така поведінка стає нормою.

Ще один важливий фактор – потреба самоствердитися за рахунок інших. Це особливо характерно для підлітків, які шукають своє місце в соціальній групі. Якщо в компанії прийнято жартувати над кимось або «травити» когось, інші можуть долучатися просто, щоб не випадати з

колективу. У таких випадках кібербулінг може виглядати як частина групової динаміки, а не як щось «особисте».

Іноді агресія в інтернеті – це вихід для внутрішніх проблем. Людина може відчувати себе нереалізованою, ображеною, знехтуваною – і починає «зриватися» на інших онлайн. Таке трапляється не лише серед дітей, а й серед дорослих. Соціальні мережі стають своєрідним «випускним клапаном», де люди зливають негатив, не задумуючись, кого це може поранити. Крім того, агресія може мати мотиви помсти: 44% підлітків, які здійснюють кібербулінг, вказують, що вони хочуть помститися за те, що їх ображали раніше [7].

Також варто згадати про вплив культури та медіа. У деяких онлайн-середовищах токсичність буквально вшита в «правила гри». Наприклад, у певних геймерських або мем-спільнотах висміювання інших подається як розвага. І чим жорсткіше, тим смішніше. Молодь, яка виростає в такому середовищі, може просто не бачити нічого поганого в булінгу – це здається їм звичним.

Низький рівень цифрової грамотності – ще один ключовий чинник. Люди не завжди усвідомлюють, що навіть «лайк» під образливим коментарем або поширення принизливого мему вже робить їх частиною цькування. Особливо це стосується дітей, які ще не вміють аналізувати наслідки своїх дій у мережі.

Окремо варто згадати про вплив дорослих та авторитетів. Якщо в сім’ї або серед дорослих авторитетних людей нормалізується агресія, приниження або знецінення, діти можуть переймати цю модель поведінки. Онлайн – просто ще один простір, де ця модель реалізується.

Іноді бувають і технічні фактори: наприклад, алгоритми соцмереж, які підсилюють видимість «скандальних» або агресивних дописів, лише підливають олії у вогонь. Чим більше лайків чи коментарів набирає провокація – тим ширше вона розповсюджується, і це створює заохочення для подальшої агресії.

Причини кібербулінгу можуть бути різними, але, незалежно від мотивації агресора, його дії рідко минають безслідно для жертви. Кібербулінг – це не просто «неприємне повідомлення» чи «жарт у коментарях» і його наслідки можуть бути глибокими і довготривалими. Іноді одна фраза, написана в злісному тоні або поширена в інтернеті, залишає вагоміший слід, ніж слова, сказані в обличчя. Особливо якщо це відбувається не один раз, а систематично.

Найперше, що страждає в жертви кібербулінгу – це психіка. Людина починає сумніватися в собі, відчуває сором, страх, тривогу. Вона може боятися перевіряти повідомлення або заходити в соцмережі, де зазвичай проводила багато часу. З'являється постійне напруження, яке важко пояснити оточенню. У деяких випадках це переростає в депресію, панічні атаки або розлади сну.

Підлітки – найуразливіша група, тому що їхня особистість ще формується. Якщо дитина постійно чує, що вона «нікчемна», «потворна» або «нікому не потрібна», то з часом вона може почати в це вірити. Це впливає на самооцінку, мотивацію, здатність будувати стосунки з іншими людьми. Іноді все доходить до того, що дитина починає уникати школи, ізолюється, перестає говорити з батьками чи друзями.

Кібербулінг також може викликати фізіологічні симптоми. Це, наприклад, головний біль, втрата апетиту, нудота, тремор, проблеми зі шлунком. Організм реагує на стрес так, як у ситуації реальної небезпеки. Якщо такий стан триває тижнями або місяцями – здоров'я починає страждати серйозно.

У складніших випадках кібербулінг може призвести до самоушкоджень або навіть суїцидальних думок. На жаль, у світі є чимало трагічних історій, коли через онлайн-цькування підлітки накладали на себе руки. Найстрашніше в тому, що зовні все могло виглядати «нормально»: хороша сім'я, оцінки, друзі. Але в душі – постійний тиск, біль і відчуття, що це ніколи не закінчиться.

Наслідки торкаються не тільки жертви. Спостерігачі, які бачать кібербулінг, але не втручаються, можуть відчувати провину або безсилля. Вони починають боятися, що самі стануть наступною мішенню. Це створює атмосферу загального страху і напруги, особливо в замкнутих онлайн-групах або шкільних спільнотах.

Також страждає й сам агресор, хоча це не завжди помітно одразу. Люди, які регулярно принижують інших, самі часто мають проблеми з емоційною регуляцією, емпатією або навичками комунікації. Якщо таку поведінку не зупинити на ранньому етапі, з часом вона стає звичкою. Це може призвести до конфліктів у реальному житті, труднощів у побудові здорових стосунків і навіть юридичних наслідків, якщо мова йде про серйозні форми переслідування.

Кібербулінг має також соціальні наслідки. Люди втрачають довіру до онлайн-платформ, закривають свої акаунти, обмежують контакти. У деяких випадках ціла спільнота або клас розділяється на «табори», і конфлікт з віртуального простору переноситься в реальне життя.

1.1.2 Вплив технологій на поширення кібербулінгу

Технології відіграють ключову роль у тому, як розгортається кібербулінг. Анонімність, швидкість поширення інформації, велика аудиторія – усе це створює ідеальні умови для агресивної поведінки. Навіть одне фото або фраза, викладена в інтернет, може «піти в народ» за кілька годин і перетворити життя людини на кошмар. І що найгірше – іноді кібербулінг не зупиняється навіть після звернення до адміністрацій платформ або блокування агресора. Він може повернутися під іншим акаунтом або продовжитись через інших користувачів.

Загальний доступ до цифрових пристроїв, таких як смартфони, планшети та комп’ютери, значно збільшився серед підлітків. Це дозволяє їм бути онлайн практично цілодобово, що збільшує ймовірність стикнутися з

кібербулінгом. Дослідження показують, що доступ до мобільних телефонів та цифрових технологій у підлітковому віці пов'язаний як з наявністю жертв, так і з поширенням самого кібербулінгу [8].

Соціальні мережі стали основним майданчиком для проявів кібербулінгу. Платформи, такі як Instagram, TikTok та YouTube, дозволяють користувачам публікувати контент, коментувати та взаємодіяти, що може бути використано для цькування.

Залежність від соціальних мереж серед підлітків також пов’язана з підвищеним ризиком стати жертвою кібербулінгу. Дослідження показують, що підлітки, які проводять понад 7 годин на день в онлайні, частіше стикаються з цькуванням в інтернеті [9].

Згідно з даними Cyberbullying.org (рисунок 1.3), більшість американських підлітків, які стикалися з кібербулінгом, повідомляють про доволі різноманітні та часто повторювані форми цифрового насильства [1].

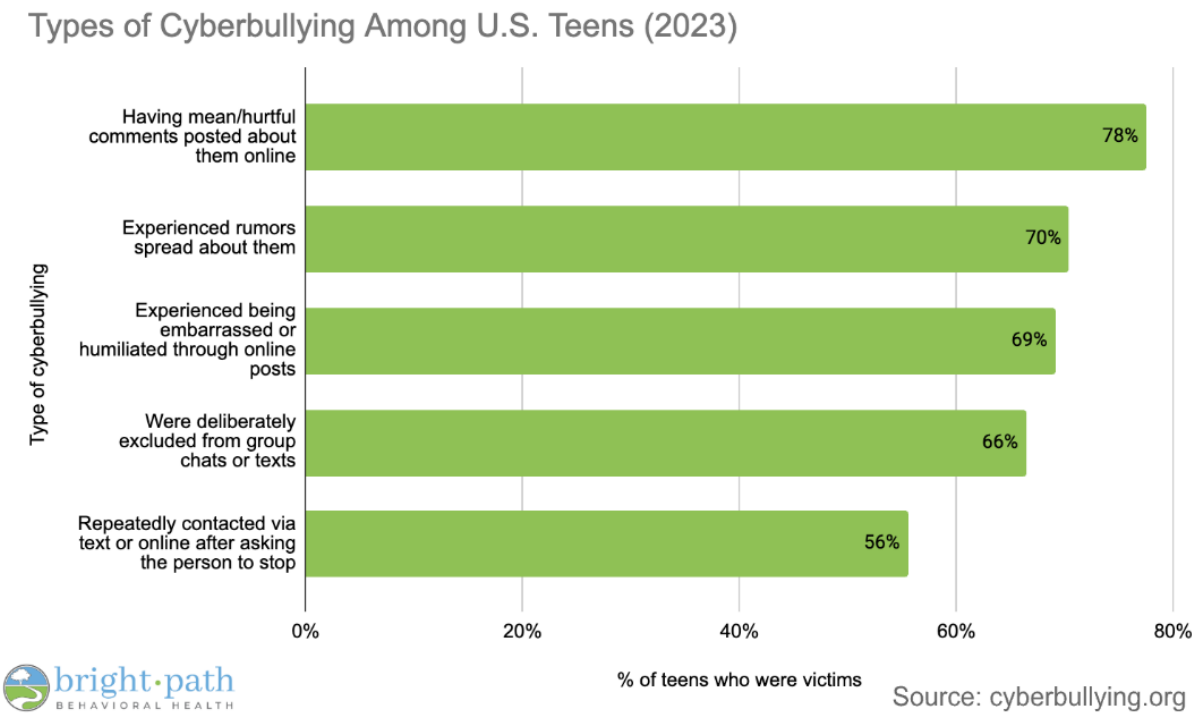

Рисунок 1.3 – Найчастіші випадки кібербулінгу серед підлітків (2023)

У 2023 році дослідження показало такі результати:

– 78% постраждалих підлітків розповіли, що стикалися з образливими або принизливими коментарями, які публікувались про них в інтернеті;

– 70% заявили, що про них поширювали неправдиві чутки через онлайн-платформи;

– 69% зазнали публічного приниження або зневаги у вигляді постів, що мали на меті викликати сором чи осміяння;

– 66% повідомили, що їх навмисно виключали з онлайн-чатів або групових обговорень;

– 56% підлітків зазначили, що після прохання припинити спілкування, їм продовжували надсилати повідомлення або звертатися онлайн.

Ці дані показують, що форми кібербулінгу охоплюють не лише прямі образи, але й соціальну ізоляцію, повторюване переслідування та порушення особистих меж. Значну роль у поширенні таких форм відіграють цифрові платформи – месенджери, соціальні мережі, анонімні форуми. Технології, що надають можливість миттєвого зв’язку, створення контенту та широкого охоплення аудиторії, одночасно стають і інструментом для булінгу.

1.1.3 Міжнародний досвід боротьби з кібербулінгом

У деяких країнах кібербулінг вже визнаний правопорушенням і за нього можуть бути передбачені серйозні наслідки. Наприклад, Фінляндія є одним із лідерів у боротьбі з кібербулінгом завдяки програмі KiVa, запровадженій у 2009 році. Ця програма спрямована на попередження та втручання у випадки цькування в школах, включаючи онлайн-цькування. Вона поєднує навчання учнів, підтримку вчителів та активну участь

батьків. Дослідження показують, що KiVa ефективно знижує рівень цькування серед учнів [10].

У 2025 році уряд Австралії запропонував законопроєкт, який забороняє дітям до 16 років користуватися соціальними мережами. Цей крок спрямований на захист молоді від кібербулінгу, насильницького контенту та залежності від інтернету [11]. Згідно з проєктом закону, платформи повинні будуть перевіряти вік користувачів, а компанії, які не виконуватимуть вимоги, можуть бути оштрафовані до 2 мільйонів австралійських доларів.

У США законодавство щодо кібербулінгу варіюється в залежності від штату. Принаймні 45 штатів ухвалили закони, що криміналізують цифрове переслідування. Наприклад, у Каліфорнії ухвалено закон, який надає адміністраціям шкіл право карати учнів за цькування як онлайн, так і офлайн. Однак відсутність єдиного федерального закону ускладнює ефективну боротьбу з кібербулінгом на національному рівні.

В Іспанії діє програма Prev@cib, спрямована на зниження рівня цькування та кібербулінгу серед підлітків [12]. Програма охоплює як класні, так і віртуальні середовища, і показала результати у зменшенні випадків цькування та жертвування серед учнів.

У Малайзії уряд активно співпрацює з соціальними мережами для боротьби з кіберзлочинами, включаючи кібербулінг. У першому кварталі 2024 року було надіслано понад 51 тисячу запитів на видалення контенту, пов'язаного з кібербулінгом. Однак ефективність цих заходів залежить від співпраці платформ, де рівень відповідності варіюється від 25% до 88% [13].

В Україні питання кібербулінгу набуває все більшої актуальності. Однак на законодавчому рівні відсутні чітко визначені норми, що регулюють це явище. Натомість, це питання часто розглядаються в контексті загальних норм про захист дітей та боротьбу з насильством. Це свідчить про необхідність розробки спеціалізованого законодавства та впровадження програм профілактики в освітніх установах.

1.2 Огляд існуючих рішень

У зв’язку з активним поширенням кібербулінгу в цифровому просторі, наукова спільнота все більше зосереджується на розробці автоматизованих методів його виявлення. Традиційні способи модерації вмісту в соціальних мережах виявляються недостатньо ефективними через великий обсяг даних, швидку динаміку комунікації та складність мовного контексту. У цьому контексті штучний інтелект, зокрема машинне навчання та обробка природної мови (NLP), відкривають нові можливості для створення точних та масштабованих систем моніторингу [14].

Дослідження «Assessing Text Classification Methods for Cyberbullying Detection on Social Media Platforms», представлене на arXiv у грудні 2024 року, зосереджено на оцінці ефективності великих мовних моделей, таких як BERT, RoBERTa, XLNet, DistilBERT та GPT-2, у задачах виявлення кібербулінгу. Результати (рисунок 1.4) показали, що BERT забезпечує баланс між продуктивністю, ефективністю за часом та використанням обчислювальних ресурсів, досягаючи точності 95% при низькому споживанні пам'яті та енергії. Це підкреслює потенціал трансформерних моделей у реальному застосуванні, хоча їх ефективність може залежати від якості та обсягу навчальних даних [15].

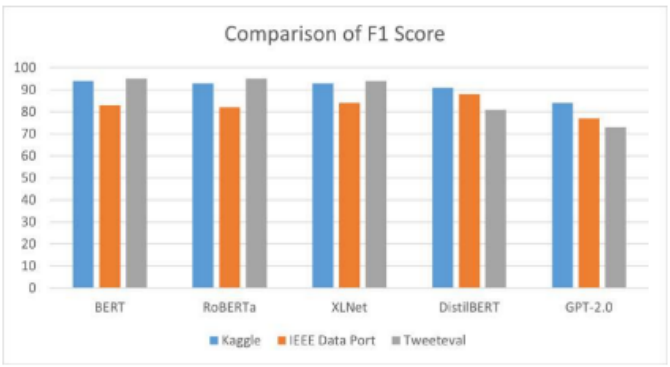

Рисунок 1.4 – Показники F1-міри для різних стратегій навчання трансформерних моделей

У іншому дослідженні «Cyberbullying Detection: Hybrid Models Based on Machine Learning and Natural Language Processing Techniques», опублікованому в журналі Electronics у 2021 році, автори представили порівняльний аналіз одинадцяти класифікаційних методів, включаючи як традиційні алгоритми машинного, так і глибокі нейронні мережі. Використовуючи два реальні набори даних, вони виявили, що бінаправлені рекурентні нейронні мережі з увагою та векторними представленнями слів, наприклад, GloVe, досягають найвищих результатів, з точністю та F1-мірою до 98% (рисунок 1.5). Це свідчить про ефективність глибоких моделей у задачах класифікації тексту, хоча варто зазначити, що такі моделі можуть вимагати значних обчислювальних ресурсів та часу на навчання [16].

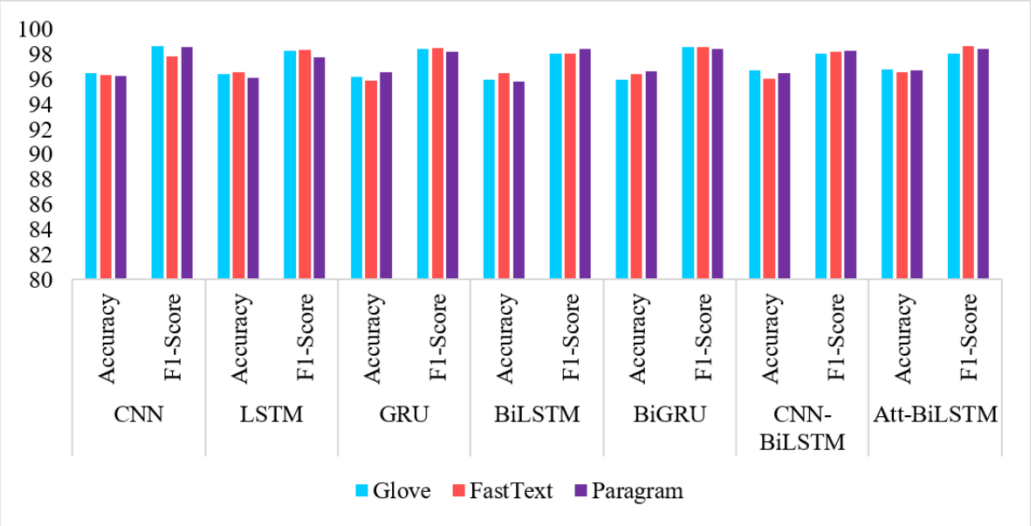

Рисунок 1.5 – Показники точності та F1-міри для різних моделей

У дослідженні «Improving automatic cyberbullying detection in social network environments by fine-tuning a pre-trained sentence transformer language model», опублікованому в журналі Social Network Analysis and Mining у липні 2024 року, запропоновано підхід до виявлення кібербулінгу шляхом донавчання попередньо натренованої моделі Sentence-BERT (рисунок 1.6). Створивши нові навчальні набори даних, що складаються з пар речень,

автори досягли покращення результатів у порівнянні з попередніми методами. Цей підхід демонструє ефективність у виявленні нюансів у текстах, пов'язаних з кібербулінгом, хоча потребує ретельного підбору навчальних даних для досягнення оптимальних результатів [17].

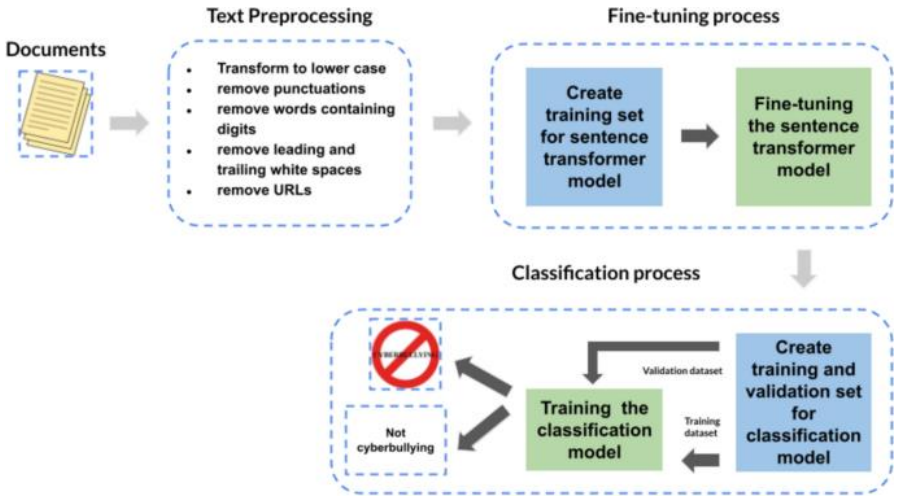

Рисунок 1.6 – Схема реалізації моделі виявлення кібербулінгу

Дослідження «Detecting harassment and defamation in cyberbullying with emotion-adaptive training», представлене на arXiv у січні 2025 року, зосереджено на виявленні різних форм кібербулінгу, таких як переслідування та наклеп, з використанням емоційно-адаптивного навчання. Автори створили новий набір даних, що охоплює ці аспекти, та застосували різні трансформерні моделі, включаючи RoBERTa, BERT, Electra та інші. Запропонований ними підхід дозволив покращити середні значення F1-міри, точності та повноти на 20% у задачах виявлення кібербулінгу в умовах обмежених ресурсів.

Візуалізація результатів (рисунок 1.7) демонструє порівняння ефективності трьох стратегій: повного донавчання (Fine Tuning), нульового-shot навчання (Zero-shot) та навчання на декількох прикладах (Few-shot). Найвищу точність модель показала в режимі повного донавчання, досягаючи чіткої класифікації у всіх трьох класах. У режимі Zero-shot

спостерігається значне зниження якості для класів 0 і 2, тоді як Few-shot дає збалансованіші результати, демонструючи покращену класифікацію порівняно з Zero-shot, але дещо нижчу, ніж при Fine Tuning. Це свідчить про важливість врахування емоційного контексту у виявленні непрямих форм кібербулінгу, хоча реалізація таких моделей може бути складною у практичному застосуванні [18].

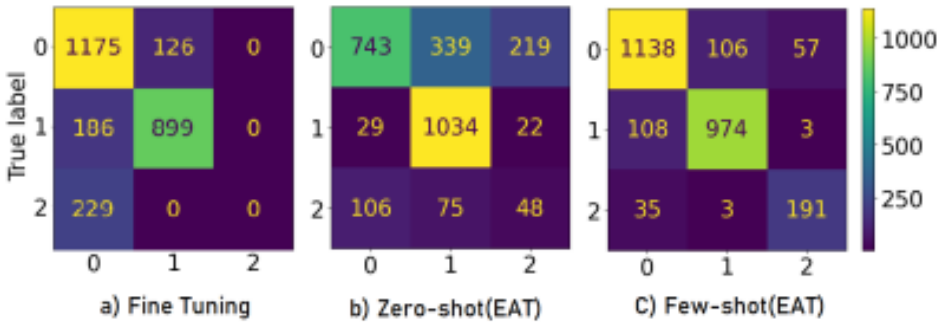

Рисунок 1.7 – Матриці неточностей для трьох стратегій навчання

Нарешті, дослідження, опубліковане на arXiv у лютому 2024 року, представило ансамблеву модель машинного навчання для виявлення кібербулінгу. Об’єднавши п’ять різних алгоритмів, включаючи дерева рішень, випадкові ліси, SVM, логістичну регресію та метод найближчих сусідів, автори досягли точності 94% у класифікації агресивних твітів. Архітектура запропонованої системи (рисунок 1.8) передбачає багаторівневу обробку: спочатку здійснюється векторизація тексту з використанням BoW, TF-IDF, Word2Vec або GloVe, після чого отримані ознаки подаються на вхід базових моделей. Результати їхніх передбачень комбінуються за допомогою стекінгу, де як мета-алгоритм виступає випадковий ліс. Цей підхід демонструє переваги комбінування різних моделей для покращення загальної продуктивності, хоча може бути чутливим до вибору та налаштування окремих компонентів ансамблю [19].

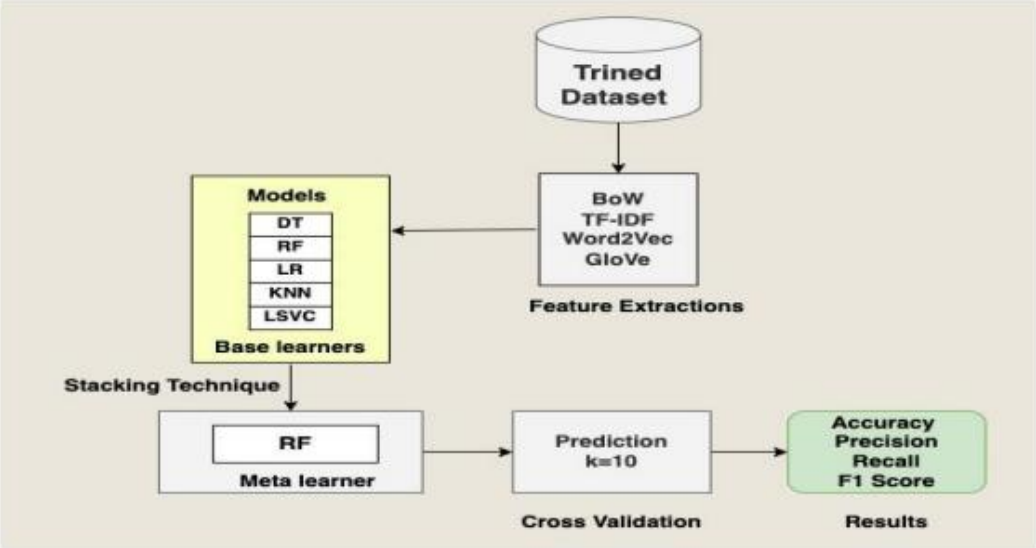

Рисунок 1.8 – Структура багаторівневої ансамблевої моделі з використанням стекінгу

1.3 Постановка задачі

Метою даної кваліфікаційної роботи є розробка і порівняння моделей для виявлення кібербулінгу різного рівня складності для автоматичного виявлення ознак кібербулінгу у текстових повідомленнях із застосуванням сучасних методів NLP і ML. Очікується, що результати цього дослідження можуть бути використані в реальних системах моніторингу соціальних мереж, месенджерів, освітніх платформ або навіть в ком’ютерних іграх.

В рамках роботи поставлено наступні завдання:

– здійснити аналіз сучасних підходів до автоматичного виявлення кібербулінгу, включаючи методи машинного навчання, обробки природної мови та штучного інтелекту;

– дослідити типові проблеми, які виникають при розробці таких систем;

– обрати корпус даних, придатний для задачі виявлення кібербулінгу, та здійснити його попередній аналіз і базову обробку для подальшого використання у моделюванні;

– обрати методи машинного навчання, які підходять для класифікації текстів із ознаками агресії або цькування;

– побудувати кілька моделей з використанням різних підходів та рівнів складності;

– забезпечити навчання моделей на підготовлених даних, враховуючи особливості формулювання задачі;

– здійснити порівняльний аналіз отриманих моделей за якісними та кількісними метриками, як-от: точність, повнота та F1-міра;

– дослідити, як змінюється ефективність залежно від складності моделі, та надати рекомендації щодо доцільності їх використання в залежності від конкретних умов застосування.

У дослідженні використовуються методи обробки природної мови, машинного навчання, а також підходи до проєктування багаторівневих моделей класифікації текстів. Враховуючи специфіку задачі, акцент буде зроблено на використанні сучасних технік для аналізу та класифікації текстових даних, зокрема, на застосуванні попередньо навчених векторних представлень слів. Для реалізації практичної частини буде застосовано мову програмування Python, яка забезпечує високий рівень гнучкості та інтеграції з провідними бібліотеками для обробки тексту та машинного навчання. Зокрема, використовуватимуться бібліотеки Scikit-learn для базових алгоритмів машинного навчання, PyTorch для розробки глибоких нейронних мереж, а також попередньо навчені векторні представлення слів, що дозволяють значно покращити результати класифікації при обробці текстових даних.

Проєкт передбачає реалізацію трьох моделей виявлення кібербулінгу, кожна з яких буде орієнтована на різні рівні складності задачі. Основна мета – дослідити, які методи показують найкращі результати для різних сценаріїв використання.

2 МЕТОДИ ОБРОБКИ ПРИРОДНЬОЇ МОВИ

2.1. Основи обробки природної мови

Обробка природної мови (англ. Natural Language Processing, NLP) – це напрям штучного інтелекту, який займається взаємодією між комп’ютерами та людською мовою. Головна мета NLP полягає в тому, щоб дати комп’ютерам можливість розуміти, інтерпретувати, аналізувати й генерувати природну мову таким чином, який є зрозумілим для людини.

Цей напрям поєднує в собі знання з лінгвістики, інформатики та статистики. З одного боку, потрібно враховувати мовні особливості: граматику, синтаксис, семантику. З іншого – розробляти алгоритми, які можуть працювати з текстовими даними, що є неоднорідними, шумними й часто непередбачуваними. Наприклад, люди можуть писати з помилками, використовувати сленг або навіть емодзі, і система має навчитися правильно це інтерпретувати.

Перші спроби автоматичної обробки мови з'явилися ще в середині XX століття, коли дослідники намагалися створити системи машинного перекладу. Проте через обмеженість комп’ютерних ресурсів і недостатнє розуміння мовної природи такі системи довгий час залишалися примітивними. Наприклад, англійський ідіоматичний вираз «It’s raining cats and dogs» одна з ранніх систем могла перекласти буквально як «Йде дощ із котів і собак», що в українській зовсім не має сенсу.

Значного прориву вдалося досягти з розвитком машинного навчання, зокрема завдяки нейронним мережам, які дозволили працювати з мовними даними на новому рівні. Сучасні мовні моделі, такі як BERT або GPT, навчаються на мільярдах слів і здатні виконувати широкий спектр задач – від автоматичного резюмування тексту до генерації зв’язних відповідей у чаті.

Сьогодні NLP використовується в широкому спектрі додатків: від голосових помічників, наприклад, Siri або Google Assistant, і чат-ботів у службах підтримки, до систем автоматичного перекладу, виявлення емоцій у відгуках клієнтів, фільтрації спаму в електронній пошті, пошукових систем та інструментів модерації контенту в соціальних мережах.

Одним із напрямів застосування NLP є також виявлення токсичної поведінки в мережі, зокрема кібербулінгу – коли алгоритми навчаються розпізнавати мову ворожнечі, погрози чи знущання на основі аналізу тексту. Наприклад, система може ідентифікувати повідомлення типу «ти нічого не вартий» або «краще б ти зник» як образливі.

Сучасні NLP-системи ґрунтуються на великій кількості мовних даних, або корпусів, які дозволяють моделям навчатися контексту та структури мови. Наприклад, корпуси новин, твіти, діалоги в месенджерах або тексти форумів дають змогу враховувати різноманітні мовні стилі. Також активно використовуються попередньо натреновані мовні моделі, які демонструють високі результати навіть у складних задачах без потреби в надвеликих спеціалізованих датасетах.

2.1.1 Основні задачі обробки тексту

Обробка природної мови охоплює широкий спектр задач, спрямованих на те, щоб комп’ютерні системи могли не лише аналізувати, а й розуміти, інтерпретувати та генерувати людську мову. Застосування таких технологій охоплює як побутові цифрові сервіси, так і складні наукові чи бізнесові системи.

Однією з найважливіших задач є семантичний аналіз тексту, тобто виявлення смислу висловлювань. Цей процес використовується у віртуальних асистентах, які повинні розуміти запити користувачів незалежно від формулювань. Наприклад, системи аналізу відгуків

споживачів повинні автоматично визначати, чи є відгук позитивним чи негативним, навіть якщо в ньому не вжито явних оціночних слів.

Ще однією поширеною задачею є машинний переклад – автоматичне перетворення текстів з однієї мови на іншу. Сучасні перекладачі на основі трансформерних моделей, таких як Google Translate або DeepL, здатні зберігати контекст та адаптуватися до стилю тексту, що є величезним досягненням у порівнянні з першими спробами перекладу, які оперували переважно словниками та жорсткими правилами.

Ще одним важливим напрямом є семантичний аналіз тексту, який дозволяє системі розпізнавати значення слів і речень у конкретному контексті. Це особливо актуально для завдань на кшталт автоматичної класифікації повідомлень, виявлення фейкових новин, визначення тону повідомлення тощо. У контексті боротьби з кібербулінгом семантичний аналіз допомагає виявити приховану агресію або сарказм, які важко розпізнати простими лексичними методами.

У сфері безпеки важливим напрямом є виявлення токсичного чи агресивного контенту, зокрема в соціальних мережах. Такі системи автоматично ідентифікують прояви мови ворожнечі, кібербулінгу або дезінформації. Вони використовуються для модерації коментарів, виявлення потенційних загроз або для захисту вразливих груп користувачів.

Інформаційний пошук і витяг фактів – ще одна ключова задача. Вона полягає у знаходженні релевантних фрагментів тексту або конкретних фактів у великих корпусах даних. Наприклад, системи запитання-відповідь у пошукових платформах або цифрові помічники здатні витягувати точну інформацію із мільйонів документів у реальному часі.

Розпізнавання іменованих сутностей, таких як імена людей, організацій, географічні назви, також має важливе значення в інформаційній аналітиці. Цей інструмент широко застосовується в журналістиці, безпеці, фінансовому аналізі та, зокрема, у виявленні агресивних повідомлень, коли

важливо розуміти, кого саме згадують у негативному контексті. Приклад розпізнавання іменованих сутностей наведений на рисунку 2.1 [20].

Рисунок 2.1 – Розпізнавання іменованих сутностей у spaCy

Не менш значущою є автоматична генерація тексту. Вона використовується в створенні новин, рефератів, рекомендацій, а також у чат-ботах, які здатні вести осмислений діалог. Генеративні моделі типу GPT можуть формулювати зв’язні тексти на задану тему, адаптуючи стиль, мову і навіть емоційний тон.

2.1.2 Особливості обробки неформального тексту

Тексти, що створюються в соціальних мережах, месенджерах або онлайн-форумах, суттєво відрізняються від традиційного письмового мовлення, з яким зазвичай працюють класичні мовні моделі. Вони коротші, менш структуровані, часто емоційно забарвлені та включають елементи розмовної мови. Це створює додаткові виклики для систем обробки природної мови.

Однією з ключових особливостей є нестандартна орфографія. Користувачі можуть навмисно спотворювати слова, наприклад, «крууууто», для емоційного підсилення. Такі слова не входять до словників стандартної мови, тому система не завжди правильно їх розпізнає або розуміє їхню силу емоційного впливу.

Ще один виклик – використання сленгу, жаргону та скорочень. Наприклад, у підлітковому середовищі поширені слова на кшталт «крінж», «зашквар» у значеннях, які відсутні в класичних словниках. Те саме стосується англійських скорочень типу «LOL», «SMH», «IDK», або українських варіантів, як-от «спс», що означає «спасибі».

Крім того, неформальні тексти часто містять велику кількість емодзі, знаків пунктуації або їх дублювання, що додає емоційного контексту, але ускладнює автоматичну інтерпретацію. У багатьох випадках саме емодзі замінюють слова або підсилюють їх, ігнорування цих елементів може призвести до втрати важливої інформації про тон або зміст повідомлення.

Ще одна характерна риса – граматична неструктурованість. Через швидкий темп комунікації користувачі часто нехтують правилами пунктуації, використовують уривчасті речення, не дотримуються порядку слів. Наприклад, фраза «Ти що серйозно капець» є граматично неправильною, але цілком зрозумілою у відповідному контексті – однак для машини вона виглядає як набір випадкових слів.

Кібербулінг часто проявляється саме в такому неформальному стилі: агресія, образи, погрози або приниження передаються не тільки через конкретні слова, але й через манеру письма, прихований сарказм або натяки. Це значно ускладнює автоматичне виявлення подібних проявів, оскільки звичайні підходи, що працюють на формалізованому тексті, можуть бути неефективними.

2.2. Попередня обробка текстових даних

Перш ніж подавати текстові дані на вхід алгоритмам машинного навчання чи моделям обробки природної мови, їх необхідно привести до уніфікованого, структурованого формату. Цей етап відомий як попередня обробка тексту і є критично важливим для ефективності майбутньої моделі. Особливо це стосується завдань, пов’язаних із виявленням кібербулінгу, де

тексти часто емоційно забарвлені, неформальні та різноманітні за стилістикою.

Мета попередньої обробки – зменшити шум у даних і виділити суттєві ознаки, які сприятимуть кращому розпізнаванню змісту повідомлень. Це включає усунення або нормалізацію таких елементів, як надлишкова пунктуація, емодзі, HTML-теги, спеціальні символи або дублікатні пробіли. У випадку з кібербулінгом часто доводиться мати справу з навмисно зміненими словами – такими, що містять додаткові літери, пропуски або помилки. У таких випадках використовуються спеціальні методи очищення тексту або словники з варіантами написання образливих термінів.

Важливо також враховувати мову тексту, оскільки автоматичне виявлення мови допомагає застосувати відповідні правила обробки, словники та морфологічні інструменти.

2.2.1 Токенізація

Токенізація є одним з базових і водночас ключових етапів попередньої обробки текстових даних у задачах обробки природної мови. Її суть полягає в розбитті суцільного тексту на окремі складові – токени, які в подальшому можуть бути оброблені або проаналізовані автоматичними алгоритмами. Залежно від поставленого завдання токенами можуть бути слова, символи, речення або навіть підслова.

Найпоширенішим типом є токенізація за словами. Вона передбачає поділ тексту на окремі лексеми, що зазвичай співпадають із словами. Наприклад, фраза «Це приклад речення.» буде перетворена на масив токенів: [«Це», «приклад», «речення», «.»]. Такий тип токенізації є придатним для багатьох завдань, зокрема класифікації текстів, виявлення тональності тощо.

У випадках, коли необхідно працювати з морфологічними особливостями або аналізувати структуру фраз на рівні символів,

використовується символьна токенізація. Вона розбиває текст на окремі символи, наприклад, зі слова «Привіт» ми отримаємо масив токенів [«П», «р», «и», «в», «і», «т»]. Цей підхід часто застосовується в задачах генерації тексту, обробки орфографічних помилок або роботи з невідомими словами.

Ще одним важливим підходом є токенізація на основі підслів. Вона широко використовується у сучасних трансформерних моделях, таких як BERT або GPT, і дозволяє ефективно працювати з новими або рідковживаними словами, яких немає в словнику. У цьому випадку слова розбиваються на частини, що повторюються в інших словах. Наприклад, слово «неприємний» може бути розбите на підслова «не», «приєм», «ний». Ще один приклад токенізації наведений на рисунку 2.2 [21].

Рисунок 2.2 – Приклад токенізації речення різними способами

Токенізація речень використовується рідше, але буває корисною у завданнях, де аналіз проводиться на рівні абзаців або діалогів. Вона передбачає поділ тексту на речення, наприклад, на основі розділових знаків або граматичних структур.

Завдання токенізації ускладнюється при роботі з неформальними текстами, такими як пости в соціальних мережах або чат-повідомлення. Тут часто зустрічаються сленг, емодзі, скорочення, помилки, відсутність пунктуації. Саме тому для таких джерел краще використовувати

спеціалізовані токенізатори або попередньо навчені моделі, адаптовані до специфіки мови користувачів інтернету.

У загальному випадку правильна токенізація значною мірою визначає якість подальших етапів обробки тексту та навчання моделей, оскільки саме на рівні токенів відбувається представлення тексту у форматі, зрозумілому для комп’ютера.

2.2.2 Стемінг та лематизація

У процесі обробки текстових даних важливо зменшити варіативність мовних форм, щоб різні граматичні форми одного й того ж слова розпізнавались як єдине поняття. Для цього застосовують стемінг і лематизацію – методи нормалізації слів, що дозволяють привести їх до базової форми, придатної для аналізу, порівняння чи класифікації.

Стемінг – це процес відсікання суфіксів або закінчень слів з метою отримання їх основи. Наприклад, слова «працював», «працюємо», «працювати» можуть бути скорочені до форми «прац». Стемери зазвичай використовують прості евристики та правила, що діють швидко, але іноді призводять до помилок. Наприклад, слова «важливий» і «важливо» можуть бути обрізані до різних або некоректних форм. Серед популярних алгоритмів стемінгу – Porter Stemmer, Snowball Stemmer та інші.

Лематизація – більш точний, але й складніший метод, що передбачає приведення слова до його леми – словникової, граматично правильної форми. На відміну від стемінгу, лематизація враховує контекст слова, його частину мови та морфологічні характеристики. Наприклад, лематизатор розпізнає, що «біг», «бігти», «бігав» – це форми дієслова «бігти», а не відсіче суфікси механічно. Для цього часто використовуються словники та морфологічні аналізатори.

У контексті обробки природної мови лематизація зазвичай дає кращі результати, особливо для завдань семантичного аналізу, класифікації

текстів або машинного перекладу. Однак вона є більш ресурсомісткою і складною в реалізації.

Для української та інших слов'янських мов, що мають багату морфологію, лематизація є особливо корисною, оскільки дозволяє зменшити кількість форм одного слова до однієї уніфікованої леми. Наприклад, «студент», «студента», «студенту», «студентом», «студенти» – усе це одна лема «студент».

У практичних задачах обидва методи можуть застосовуватись залежно від потреб: якщо потрібна висока точність – лематизація; якщо важлива швидкість обробки – стемінг. Також можливе їх комбіноване використання у попередній обробці великих корпусів тексту.

2.2.3 Видалення стоп-слів

Під час попередньої обробки тексту одним із поширених кроків є видалення стоп-слів – найуживаніших слів, які не несуть суттєвого смислового навантаження для аналізу. До таких слів зазвичай належать службові частини мови: прийменники, сполучники, займенники, допоміжні дієслова, а також загальновживані слова, що рідко мають самостійну цінність у задачах класифікації або тематичного аналізу. Наприклад, в українській мові це слова «і», «в», «на», «це», «якщо», «але».

Основна мета видалення стоп-слів полягає у зменшенні «шуму» в текстових даних і скороченні розмірності векторного простору при побудові моделей. Зменшення кількості загальновживаних слів дозволяє алгоритмам зосередитися на інформативних одиницях, які краще відображають зміст повідомлення.

Процес зазвичай реалізується шляхом порівняння кожного слова з попередньо визначеним списком стоп-слів. Такі списки доступні в багатьох мовах у вигляді бібліотек або можуть бути сформовані вручну з урахуванням специфіки конкретної задачі чи домену.

Проте, важливо враховувати, що не всі стоп-слова є марними у будь-якому контексті. Наприклад, у задачах визначення тону або наміру повідомлення слово «не» може мати критичне значення, змінюючи смисл фрази. Також у текстах кібербулінгу певні «нейтральні» слова можуть набувати токсичного відтінку залежно від оточення. Крім того, в неформальних текстах з соціальних мереж стоп-слова можуть мати додаткове значення – підсилювати емоційне забарвлення або ритм мовлення. Тому перед їх видаленням важливо враховувати специфіку задачі та тип текстів.

2.2.4 Нижній регістр, пунктуація, очищення

Ще одним важливим етапом попередньої обробки текстових даних є нормалізація тексту, яка передбачає приведення тексту до більш стандартизованого вигляду для полегшення подальшої обробки та аналізу.

Один із перших кроків нормалізації – перетворення всіх символів у нижній регістр. Це дозволяє уникнути дублювання однакових слів, записаних у різному регістрі. Переведення в нижній регістр є особливо важливим у задачах класифікації та пошуку, де точність у порівнянні термінів має велике значення.

Видалення пунктуації також сприяє зменшенню шуму в даних. Хоча деякі знаки, як-от !? або ..., можуть передавати емоційне забарвлення, у більшості задач, особливо при побудові моделей на рівні слів або токенів, вони не мають значення. Водночас у деяких задачах пунктуація може бути корисною, оскільки вона підсилює виразність повідомлення. У таких випадках її збереження або навіть спеціальне кодування може бути виправданим.

Також значну роль відіграє очищення тексту від зайвих символів та елементів, які не несуть семантичної цінності. Сюди входять:

– HTML-теги у випадку веб-даних;

– спеціальні символи;

– емодзі можуть бути вилучені або замінені відповідними мітками, якщо вони мають значення для контексту;

– числа та дати іноді нормалізуються до загальних категорій.

Варто зазначити, що ступінь очищення залежить від специфіки завдання. Наприклад, у задачах виявлення кібербулінгу надмірне очищення може призвести до втрати ключових ознак агресивності або сарказму, які проявляються у візуальному оформленні.

2.3. Векторизація тексту

Після попередньої обробки тексту наступним важливим кроком є векторизація, тобто перетворення текстових даних у числову форму, придатну для обробки алгоритмами машинного навчання. Це необхідно тому, що більшість моделей працюють з числовими вхідними даними, а не з текстами у природному вигляді.

Текстові дані, навіть після очищення й нормалізації, все ще залишаються послідовностями символів або слів, які не мають числового представлення. Завдання векторизації полягає в тому, щоб надати кожному елементу тексту – слову, фразі чи навіть символу – відповідне числове представлення, яке зберігає важливу інформацію про його зміст або контекст.

2.3.1 Bag of Words

Bag of Words або BoW – один із найпростіших та водночас найпоширеніших методів векторизації текстових даних. Його основна ідея полягає в тому, щоб представити текст у вигляді набору окремих слів без урахування порядку їх появи, граматики чи синтаксису. Кожен документ перетворюється на вектор, де кожна позиція відповідає певному слову зі

словника, а значення – кількості разів, коли це слово зустрічається в документі.

Процес побудови моделі BoW починається з формування словника – списку всіх унікальних слів, які зустрічаються у корпусі документів. Після цього кожен окремий текст кодується у вектор фіксованої довжини, де кожна компонента показує, наскільки часто відповідне слово з словника трапляється в цьому тексті.

Часто замість абсолютної кількості входжень використовують бінарні значення 0 та 1, щоб лише відображати факт присутності чи відсутності слова у тексті. Це може бути корисним у випадках, коли неважливо, скільки разів слово згадане, а важливо лише його наявність.

Попри свою простоту, такий підхід добре працює для задач, де контекст або порядок слів не є критичними. Крім того, BoW забезпечує стабільність та передбачуваність: кожен текст представляється вектором фіксованої довжини, що спрощує подальшу обробку. Однак ключовий недолік BoW полягає в тому, що він повністю ігнорує порядок слів, синтаксис і семантичні зв’язки між ними. Також BoW створює розріджені вектори дуже великої розмірності, що ускладнює обробку та потребує значних обчислювальних ресурсів при роботі з великими корпусами тексту.

2.3.2 TF-IDF

TF-IDF або Term Frequency – Inverse Document Frequency – це більш вдосконалений підхід до векторизації текстів, який не лише враховує частотність слів у межах одного документа, а й оцінює важливість кожного слова в контексті всього корпусу. Такий підхід дозволяє зменшити вагу слів, що часто трапляються у всіх документах, і підкреслити ті, що є інформативними саме для конкретного тексту.

TF-IDF складається з двох основних компонентів. Перша – term frequency – це частота появи певного терміна в документі. Вона показує,

наскільки важливим є слово всередині одного тексту. Це може бути просто кількість входжень слова або нормалізоване значення.

Друга складова – inverse document frequency – зворотна частота документів. Вона визначає, наскільки рідкісним є слово у всьому наборі документів. Якщо слово трапляється майже в кожному тексті, його IDF буде низьким, а якщо зустрічається рідко – високим. Це дозволяє знизити вагу загальновживаних слів, які не несуть специфічної інформації.

При перемноженні TF і IDF отримуємо вагу, що характеризує значимість слова для конкретного документа. Наприклад, слово «образа» може траплятися нечасто в усіх текстах, але у коментарях, що містять ознаки кібербулінгу, воно зустрічається частіше. TF-IDF дозволить такому слову отримати вищу вагу саме в цих випадках, що сприяє кращому виявленню образливих повідомлень. Приклад TF-IDF матриці наведений на рисунку 2.3 [22]. На ньому можна побачити, що слово «increased» має значення 0, бо зустрічається в кожному реченні, тобто не таке важливе для результату.

	inflation	company	increased	sales	fear	pulse	unemployment
inflation increased unemployment	0.159	0	0	0	0	0	0.159
company increased sales	0	0.159	0	0.159	0	0	0
fear increased pulse	0	0	0	0	0.159	0.159	0

Рисунок 2.3 – Приклад TF-IDF матриці

TF-IDF вирішує частину проблем BoW, зокрема – проблему однакової ваги поширених та менш значущих слів. Завдяки цьому TF-IDF краще справляється з фільтрацією інформаційного шуму і дозволяє точніше виявляти ключові терміни для кожного тексту. Він також зберігає простоту реалізації, що робить його привабливим для багатьох базових моделей.

Незважаючи на вдосконалення у порівнянні з BoW, TF-IDF також не враховує порядок слів і глибокий контекст. Також цей метод створює

вектори високої розмірності й не може ефективно працювати з неочищеним або неоднорідним текстом.

2.3.3 Word Embeddings (Word2Vec, GloVe, FastText)

Word Embeddings – це методи перетворення слів у вектори чисел так, щоб ці вектори зберігали семантичні зв’язки між словами. Основна ідея полягає в тому, щоб представлення слова відображало його контекст у великому корпусі тексту.

Основна перевага embedding’ів полягає в тому, що вони зберігають семантичні зв’язки між словами. Це означає, що подібні за значенням слова мають схожі векторні представлення, що дуже корисно для задач класифікації, тематичного аналізу чи виявлення тональності. Крім того, моделі на основі embedding’ів мають компактніші (щільні) вектори, що полегшує обчислення та підвищує продуктивність.

Проте для використання embedding’ів потрібна попередня підготовка: або завантаження готових моделей, або їх навчання на великому корпусі текстів, що може вимагати значних ресурсів. Також embedding’и іноді менш прозорі для інтерпретації – складно зрозуміти, чому модель приймає певне рішення, якщо воно ґрунтується на віддалених контекстних зв’язках.

Одними з найпоширеніших способів створення embedding’ів є Word2Vec, GloVe та FastText. Кожен із цих підходів має власні особливості та принципи роботи.

Word2Vec – це один із перших і найвідоміших підходів до створення векторних представлень слів, запропонований дослідниками з Google у 2013 році. Метод ґрунтується на припущенні, що слова, які з’являються в подібних контекстах, мають подібне значення. Модель будується за допомогою нейронної мережі, яка навчається прогнозувати слово за його контекстом або навпаки – контекст за словом. Для цього існують дві архітектури: CBOW (Continuous Bag of Words) і Skip-gram.

CBOW намагається передбачити слово за сусідніми словами, а Skip-gram – навпаки, прогнозує контекстні слова на основі цільового. Після навчання модель надає кожному слову вектор, у якому зберігається інформація про семантичну близькість.

GloVe, розроблений у Стенфорді, поєднує в собі ідеї Word2Vec та інформацію про глобальну статистику співпояви слів. На відміну від Word2Vec, який навчається на локальних контекстах, GloVe аналізує, як часто слова з'являються разом у всьому корпусі текстів. Модель створює матрицю співпояви, де кожен елемент відображає, скільки разів пара слів трапляється поруч одне з одним, і намагається відобразити ці співвідношення у векторах. Наприклад, GloVe дозволяє виявляти аналогії, такі як: «король» - «чоловік» + «жінка» = «королева» (рисунок 2.4) [23]. Результат – векторне представлення, яке також зберігає семантичні властивості, але ґрунтується на ширшому статистичному аналізі.

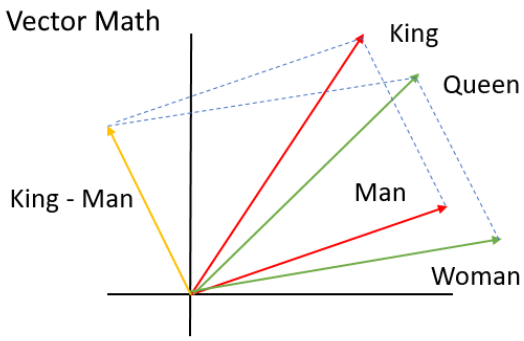

Рисунок 2.4 – Приклад аналогій, які може виявляти GloVe

FastText, розроблений Facebook AI, є продовженням і вдосконаленням Word2Vec. Його головна перевага – врахування субслівних одиниць, тобто модель не просто створює вектор для цілого слова, а аналізує його складові частини.

3 МОДЕЛІ ДЛЯ ВИЯВЛЕННЯ КІБЕРБУЛІНГУ ТА ЇХ МЕТРИКИ

Для виявлення кібербулінгу застосовуються як класичні алгоритми машинного навчання, так і сучасні нейронні мережі. До перших належать моделі, що добре працюють на структурованих текстових представленнях, зокрема логістична регресія, наївний байєсівський класифікатор, дерева рішень або підтримуючі вектори. Вони показують хорошу ефективність у поєднанні з методами векторизації, як-от TF-IDF.

Сучасні підходи базуються на глибокому навчанні, що дозволяє моделі виявляти складні залежності в тексті. Рекурентні мережі, LSTM і трансформери, зокрема BERT, стали особливо популярними для аналізу неформального мовлення та розпізнавання тонких форм агресії. Завдяки здатності враховувати контекст і семантику, вони часто перевершують традиційні підходи, особливо в задачах з великою кількістю вхідних даних.

3.1 Класичні моделі машинного навчання

Класичні моделі машинного навчання залишаються актуальними в задачах текстової класифікації завдяки своїй простоті, швидкості навчання та інтерпретованості результатів. Вони особливо ефективні при наявності якісно підготовлених ознак, таких як TF-IDF. У контексті виявлення кібербулінгу ці моделі дозволяють побудувати базові, але надійні рішення, які можуть бути використані для швидкої оцінки чи впровадження в системи з обмеженими ресурсами.

3.1.1 Логістична регресія

Логістична регресія є однією з найпростіших і водночас потужних моделей для задач бінарної класифікації, зокрема для виявлення кібербулінгу. Вона використовується для прогнозування ймовірності

належності об'єкта до одного з двох класів. У випадку з кібербулінгом – наприклад, класифікація тексту як такий, що містить ознаки булінгу, або як нейтральний.

На відміну від лінійної регресії, яка підходить для передбачення числових значень, логістична регресія моделює ймовірність приналежності до класу за допомогою сигмоїдної функції активації (рисунок 3.1), яка стискає вихід у діапазон від 0 до 1 [24]. Це дозволяє інтерпретувати результат як ймовірність позитивного класу.

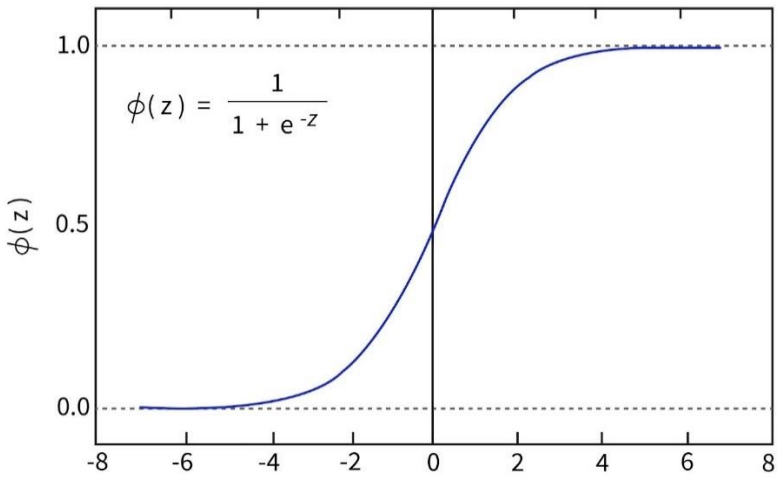

Рисунок 3.1 – Сигмоїдна функція активації

Формально модель логістичної регресії визначається так:

$$P(y = 1|x) = \sigma(w^T x + b) = \frac{1}{1 + e^{-(w^T x + b)}}, \quad (3.1)$$

де x – вектор ознак;

$\sigma(z)$ – сигмоїдна функція;

w – ваговий вектор моделі;

b – зміщення.

У задачі виявлення кібербулінгу текст повідомлення, наприклад, коментар у соціальній мережі, перетворюється в числове представлення – наприклад, за допомогою TF-IDF. Далі логістична регресія обчислює ймовірність того, що цей текст належить до категорії «кібербулінг». Якщо значення перевищує певний поріг, наприклад, 0.5, повідомлення класифікується як образливе або агресивне.

Логістична регресія привабливa для таких задач через свою швидкість та хорошу інтерпретованість: можна проаналізувати ваги w, щоб зрозуміти, які слова найбільше впливають на рішення моделі. Це робить її особливо корисною на початкових етапах розробки системи або в ситуаціях, де потрібна проста, надійна модель із невисокими обчислювальними витратами.

3.1.2 Наївний байєсівський класифікатор

Наївний байєсівський класифікатор – це проста, але напрочуд ефективна модель для задач класифікації, особливо коли йдеться про текстові дані. Його основою є теорема Байєса, яка дозволяє обчислювати ймовірність того, що об’єкт належить до певного класу, враховуючи наявні ознаки.

Теорема Байєса має вигляд:

$$P(C_k|x) = \frac{P(x|C_k) \cdot P(C_k)}{P(x)}, \tag{3.2}$$

де $P(C_k|x)$ – ймовірність того, що об’єкт з ознаками x належить до C_k;

$P(x|C_k)$ – ймовірність спостерігати ознаки x при умові, що клас – C_k;

$P(C_k)$ – апріорна ймовірність класу C_k;

$P(x)$ – ймовірність ознак x незалежно від класу.

У «наївному» варіанті цієї моделі припускається, що всі ознаки незалежні одна від одної, що рідко буває в реальних даних, але ця спрощена гіпотеза працює досить добре на практиці, особливо в текстовій класифікації.

Існує кілька варіантів наївного байєсівського класифікатора, які відрізняються способом оцінювання $P(x|C_k)$:

– Multinomial Naive Bayes найпоширеніший для текстів і працює з кількісними ознаками, наприклад, кількістю слів;

– Bernoulli Naive Bayes базується на булевих ознаках (чи присутнє слово в тексті чи ні);

– Gaussian Naive Bayes підходить для безперервних числових ознак, але рідше застосовується в NLP.

Наївний байєс добре масштабується на великі корпуси, потребує мінімальної кількості ресурсів і швидко навчається. Його часто використовують як базову модель, до якої порівнюють інші підходи. Наприклад, у задачах класифікації електронної пошти як «спам» чи «не спам», цей метод показує конкурентні результати, незважаючи на свою простоту.

Головне обмеження моделі – її припущення про незалежність ознак, що рідко виконується в реальних текстах, де значення слів сильно залежать одне від одного. Проте на практиці це обмеження не заважає наївному байєсу часто демонструвати високу ефективність у задачах класифікації текстів.

3.1.3 Дерева рішень

Дерева рішень – це інтуїтивно зрозумілий і потужний інструмент для класифікації та регресії, який моделює процес прийняття рішень у вигляді дерева зі структурою «якщо-то». У випадку задачі класифікації, таких як

виявлення кібербулінгу, дерево намагається розділити дані на підмножини, які максимально чисті за належністю до певного класу.

Основна ідея полягає в тому, що дерево рішень будується шляхом послідовного розбиття простору ознак. У кожній внутрішній вершині дерево обирає ознаку і поріг, щоб розділити вибірку на дві частини, максимально відмінні за складом класів. Це розбиття виконується до тих пір, поки не буде досягнута певна зупиняюча умова, наприклад, максимальна глибина дерева або мінімальна кількість прикладів у листі.

Щоб визначити, яке розбиття є найкращим, використовують певні математичні критерії. Один з таких – ентропія, яка оцінює міру невизначеності або «змішаності» класів у вибірці. Якщо всі елементи належать до одного класу – ентропія нульова. Якщо класи розподілені рівномірно – ентропія максимальна. Ентропія розраховується за формулою:

$$H(S) = -\sum_{i=1}^{n} p_i log_2 p_i\,, \tag{3.3}$$

де S – поточна множина прикладів;

p_i – ймовірність появи класу i у множині S;

n – кількість класів

Щоб оцінити, наскільки корисним було розбиття за певною ознакою, використовують інформаційний приріст. Він показує, наскільки зменшилася ентропія після поділу даних і обчислюється за формулою.

$$IG(S, A) = H(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} H(S_v)\,, \tag{3.4}$$

де $IG(S, A)$ – інформаційний приріст для ознаки A;

$Values(A)$ – можливі значення ознаки A;

S_v – підмножина S, у якій ознака A має значення v;

$|S|, |S_v|$ – кількість прикладів у множинах S та S_v;

$H(S_v)$ – ентропія підмножини S_v.

Іншим поширеним критерієм є індекс Джіні, який також вимірює неоднорідність множини. На відміну від ентропії, він менш чутливий до рідкісних класів і обчислюється за формулою:

$$G(S) = 1 - \sum_{i=1}^{n} p_i^2 , \tag{3.5}$$

де S – поточна множина прикладів;

p_i – ймовірність появи класу i у множині S;

n – кількість класів.

Наприклад, якщо дерево рішень навчається виявляти кібербулінг, то перше розгалуження може стосуватися наявності в повідомленні конкретного образливого слова. Далі дерево може враховувати кількість знаків оклику, довжину повідомлення чи частоту використання капітальних літер. У результаті кожен лист дерева асоціюється з остаточним прогнозом – чи є повідомлення кібербулінгом, чи ні.

До переваг можна віднести те, що дерева рішень легко інтерпретуються – їх можна візуалізувати, що робить їх цінними у задачах, де важливо пояснити, чому система зробила те чи інше рішення. Водночас вони схильні до перенавчання, особливо якщо не обмежувати їхню глибину або кількість листів. Дерева також можуть обробляти як числові, так і категоріальні ознаки без необхідності масштабування чи векторизації, що робить їх досить зручними в багатьох задачах.

3.2 Нейронні мережі для обробки тексту

Застосування нейронних мереж у задачах обробки природної мови стало справжнім проривом у розвитку методів класифікації тексту,

машинного перекладу, генерації тексту та, зокрема, виявлення кібербулінгу. Нейромережі здатні автоматично виділяти складні залежності між словами, враховувати контекст та взаємозв’язки у реченнях, що значно підвищує точність аналізу.

3.2.1 Базові feed-forward мережі

Базові feed-forward нейронні мережі є найпростішим типом штучних нейронних мереж. У таких мережах інформація проходить лише в одному напрямку – від вхідного шару до вихідного – без зворотних зв’язків. Кожен шар складається з певної кількості нейронів, які отримують вхідні значення, застосовують до них вагові коефіцієнти, передають результат через активаційну функцію та передають далі на наступний шар.

Цей тип мереж часто називають багатошаровим перцептроном, особливо коли є один або більше прихованих шарів. Мережі такого типу добре підходять для задач класифікації, зокрема й для виявлення кібербулінгу в текстах, якщо використовуються відповідні векторні представлення тексту – наприклад, TF-IDF або word embeddings. У таких випадках вектор тексту просто подається на вхід мережі, і вона навчається розпізнавати шаблони, пов’язані з образливим чи нейтральним змістом.

Основу таких мереж складають лінійні шари. Кожен вхід перетворюється згідно з формулою:

$$z = Wx + b, \tag{3.6}$$

де z – лінійне перетворення;

w – матриця ваг моделі;

x – вхідний вектор ознак;

b – вектор зміщення.

Після лінійного шару до виходу часто застосовують функцію активації, яка дозволяє мережі моделювати складні залежності. У багатьох NLP-задачах для останнього шару використовується softmax-функція, яка перетворює необроблені значення у ймовірності для кожного з можливих класів:

$$softmax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \tag{3.7}$$

де z_i – вихід нейрона для класу i;

K – загальна кількість класів.

Основна перевага feed-forward мереж полягає в їх простоті та швидкості навчання. Водночас вони мають обмежену здатність до роботи з послідовними або контекстно-залежними даними, оскільки не зберігають інформацію про порядок елементів. Тому для задач, де порядок слів має значення, такі як аналіз емоцій чи виявлення сарказму, MLP можуть поступатися більш складним архітектурам на кшталт RNN чи трансформерів.

Попри це, базові feed-forward мережі залишаються актуальними для побудови базових моделей, а також як елемент складніших архітектур або для порівняння продуктивності з іншими підходами.

3.2.2 Рекурентні нейронні мережі

У задачах обробки природної мови надзвичайно важливо враховувати послідовність слів, адже значення вислову часто залежить не тільки від самих слів, а й від їхнього порядку. Наприклад, фрази «я не люблю булінг» і «я люблю булінг» мають протилежне значення, хоча містять майже ті самі слова. Щоб ефективно працювати з подібними випадками,

використовуються рекурентні нейронні мережі, або RNN, які здатні обробляти послідовності довільної довжини.

На відміну від feed-forward мереж, які отримують фіксований вхідний вектор, RNN по черзі обробляють кожен елемент вхідної послідовності, зберігаючи інформацію про попередні кроки у прихованому стані. Це дозволяє моделі збудувати контекст для кожного слова з урахуванням його попередників.

На рисунку 3.2 зображено стандартну структуру RNN, де вхідні слова подаються послідовно, а прихований стан передається від одного кроку до іншого, зберігаючи контекст [25].

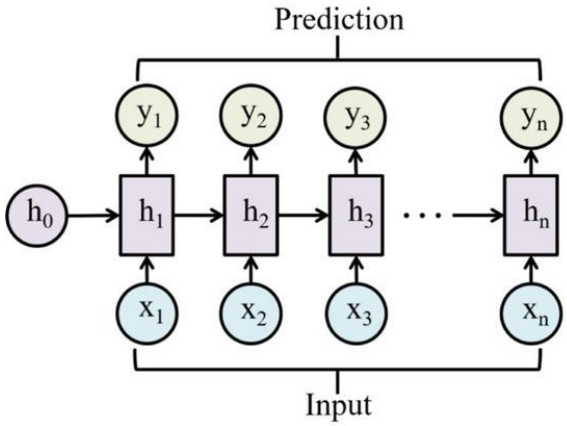

Рисунок 3.2 – Структура рекурентної нейронної мережі

Процес обчислення на кожному кроці можна описати наступною формулою:

$$h_t = tanh(W_{xh}x_t + W_{hh}h_t + b_h), \quad (3.8)$$

де h_t – прихований стан у момент часу t;

$tanh$ – активаційна функція, яка надає нелінійності;

W_{xh} – матриця ваг для поточного входу;

x_t – вхід у момент часу t;

W_{hh} – матриця ваг для попереднього стану;

b_h – вектор зміщення.

На завершальному кроці прихований стан передається до вихідного шару, наприклад, для класифікації всього тексту.

Перевага RNN полягає в тому, що вони можуть запам’ятовувати інформацію з попередніх слів, що особливо важливо для виявлення контексту в повідомленнях.

Водночас стандартні RNN мають і суттєві обмеження. Основна проблема полягає у затуханні градієнтів – коли інформація про ранні елементи послідовності «забувається» під час навчання. Щоб подолати це, були розроблені вдосконалені архітектури, які забезпечують кращу пам’ять і стабільність під час навчання довгих послідовностей.

3.2.3 Довготривала короткочасна пам’ять

Хоча рекурентні нейронні мережі здатні працювати з послідовними даними, вони мають суттєве обмеження – складність у збереженні довготривалої залежності між словами через проблему затухання або вибуху градієнтів. Це означає, що модель може забувати важливу інформацію на початку послідовності, коли обробляє її кінець. Для подолання цієї проблеми були запропоновані покращені архітектури, зокрема мережа з довготривалою короткочасною пам’яттю – LSTM, або Long Short-Term Memory.

LSTM-моделі мають спеціальну структуру, яка дозволяє керувати потоком інформації через так звані «вікна пам’яті». У структурі LSTM є три основні гейти, які виконують роль фільтрів:

- забуваючий гейт визначає, яку частину старої інформації слід зберегти;
- вхідний гейт вирішує, які нові дані додати до пам’яті;

– вихідний гейт визначає, яка частина поточного стану буде передана на вихід.

Це дозволяє LSTM утримувати релевантну інформацію протягом тривалого часу та ефективніше обробляти довгі послідовності, що особливо корисно для задач виявлення кібербулінгу, де образливий контекст може проявлятися не в окремому слові, а в комбінації фраз, що розкидані по тексту.

На рисунку 3.3 можна побачити внутрішню структуру одного блоку LSTM: взаємодію між гейтами, вхідним сигналом, пам’яттю та виходом [26].

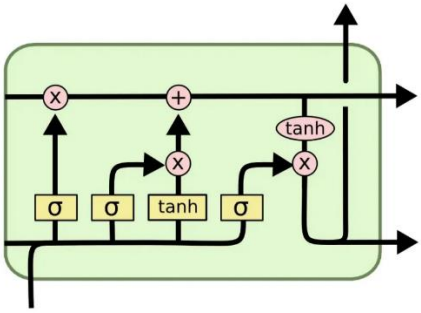

Рисунок 3.3 – Внутрішня структура блоку LSTM

LSTM широко застосовується в NLP-задачах, таких як машинний переклад, генерація тексту, аналіз емоцій і виявлення образливих повідомлень. Її здатність запам’ятовувати контекст навіть при довгих входах робить її цінною у боротьбі з кібербулінгом, де значення часто залежить не від одного слова, а від загальної тональності та історії спілкування.

3.3 Моделі на основі трансформерів

У сучасній обробці природної мови трансформерна архітектура стала основою більшості найефективніших моделей. Трансформери були

запропоновані у статті «Attention is All You Need» і революціонізували підхід до обробки послідовностей. Їх ключова особливість – повна відмова від рекурентних з’єднань, що дозволило значно прискорити обробку тексту завдяки паралельності обчислень [27].

Основу трансформера становить механізм self-attention, або самоувага, який дозволяє кожному елементу вхідної послідовності взаємодіяти з усіма іншими елементами незалежно від їх позиції. Завдяки цьому модель може враховувати контекст як зліва, так і справа від слова, що важливо для розуміння складних конструкцій мови.

Кожен шар трансформера складається з двох основних блоків:

– multi-head self-attention;

– повнозв'язна нейронна мережа.

Обидва блоки супроводжуються операціями нормалізації і залишковими зв’язками, що покращує стабільність навчання. Крім того, оскільки трансформер не обробляє дані послідовно, у модель вводяться позиційні ембедінги, які кодують порядок слів у реченні.

На рисунку 3.4 показано, як вхідні токени проходять через послідовні шари уваги та нейронної мережі, формуючи вихідні представлення з урахуванням контексту.

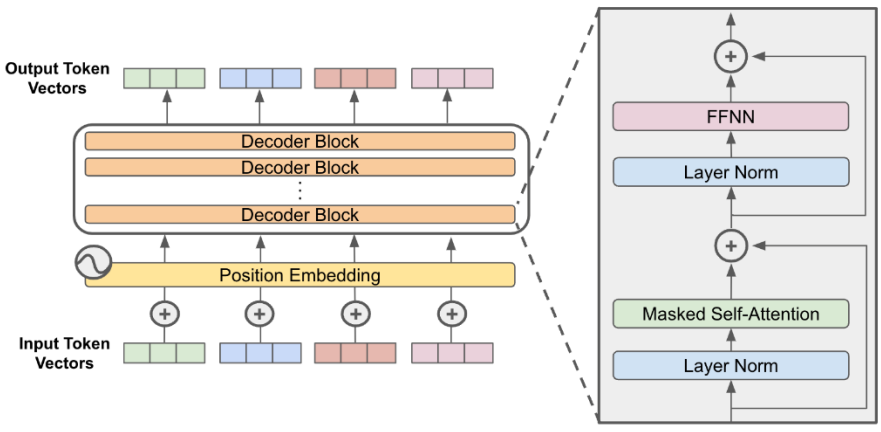

Рисунок 3.4 – Архітектура трансформеру

Однією з найвідоміших моделей, заснованих на трансформерах, є BERT. Її особливість – двонаправлений контекст, тобто під час навчання враховується не лише попередній, а й наступний контекст слова. Це дозволяє моделі краще розуміти значення слів у різних ситуаціях.

BERT навчається за допомогою двох завдань:

– Masked Language Modeling – випадкові слова маскуються, і модель повинна їх передбачити;

– Next Sentence Prediction – визначення, чи є одне речення наступним після іншого в тексті.

Після попереднього навчання на великому корпусі текстів BERT може бути донавчений на конкретну задачу, наприклад, класифікацію повідомлень на наявність кібербулінгу. Для цього до виходу моделі додається додатковий класифікаційний шар.

Використання трансформерних моделей значно підвищує точність у задачах аналізу тексту, включно з виявленням кібербулінгу. Їх гнучкість, контекстна чутливість і можливість масштабування роблять їх незамінними у сучасних NLP-системах.

3.4 Метрики якості моделей

У процесі розробки та оцінки моделей машинного навчання, зокрема для завдань виявлення кібербулінгу, надзвичайно важливо визначити, наскільки добре модель виконує своє призначення. Для цього застосовуються спеціальні метрики якості, які дозволяють кількісно оцінити точність передбачень моделі та порівняти між собою різні алгоритми.

3.4.1 Accuracy, recall, precision, F1-міра

Точність, або accuracy – це одна з найпростіших і найбільш інтуїтивно зрозумілих метрик, яка відображає частку правильних передбачень серед

усіх зроблених. Інакше кажучи, вона показує, наскільки часто модель класифікує приклади правильно, незалежно від класу. Ця метрика є особливо корисною тоді, коли класи в наборі даних збалансовані, тобто прикладів кожного класу приблизно однакова кількість.

Точність розраховується за формулою:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \quad (3.9)$$

де TP – кількість істинно позитивних передбачень;

TN – кількість істинно негативних передбачень;

FP – кількість хибнопозитивних передбачень;

FN – кількість хибнонегативних передбачень.

Попри свою простоту, точність може вводити в оману в умовах незбалансованих даних. Наприклад, якщо у 95% прикладів кібербулінгу немає, модель, яка завжди передбачає «немає кібербулінгу», матиме 95% точності, хоча нічого корисного фактично не робить.

Повнота, або recall, показує, яку частку справжніх випадків кібербулінгу модель змогла правильно виявити. Ця метрика є критично важливою у завданнях, де пропуск позитивних випадків є неприйнятним, наприклад, у системах модерації.

Повнота розраховується за формулою:

$$Recall = \frac{TP}{TP + FN}, \quad (3.10)$$

де TP – кількість істинно позитивних передбачень;

FN – кількість хибнонегативних передбачень.

Висока повнота означає, що модель здатна виявляти більшість випадків кібербулінгу, навіть якщо це іноді супроводжується помилками – хибнопозитивними спрацюваннями. Головна перевага повноти – вона

дозволяє знизити ризик того, що небезпечні або образливі повідомлення залишаться непоміченими.

Precision відображає, яку частку серед усіх виявлених випадків кібербулінгу становлять ті, що дійсно є такими. Інакше кажучи, це показник точності передбачень саме позитивного класу. Вона є важливою в контекстах, де хибні звинувачення є проблемою.

Precision розраховується за формулою:

$$Precision = \frac{TP}{(TP + FP)}, \tag{3.11}$$

де TP – кількість істинно позитивних передбачень;

FP – кількість хибнопозитивних передбачень.

Висока precision означає, що модель рідко помиляється, коли стверджує наявність кібербулінгу. Це особливо важливо в системах, де автоматичні дії – наприклад, попередження чи блокування – мають серйозні наслідки. Однак надмірна орієнтація на прецизійність може призвести до втрати багатьох справжніх випадків кібербулінгу, бо модель буде «обережною» і не ризикуватиме помилитися.

F1-міра – це гармонічне середнє між precision та recall. Вона дозволяє знайти баланс між цими двома метриками і є особливо корисною у випадках, коли необхідно враховувати як помилкові пропуски, так і помилкові спрацювання. Гармонічне середнє використовується тому, що воно «штрафує» занадто низьке значення однієї з метрик: якщо precision дуже висока, а recall – дуже низька, F1 буде низькою.

Метрика розраховується за формулою:

$$F1 = \frac{2(Precision \cdot Recall)}{(Precision + Recall)}. \tag{3.12}$$

Перевага F1-міри – її здатність адекватно оцінювати якість моделей у незбалансованих задачах, де звичайна точність може бути неінформативною.

3.4.2 Матриця неточностей

Матриця неточностей – це інструмент, який широко використовується у задачах класифікації для оцінки якості роботи моделі. Вона дозволяє зрозуміти, наскільки правильно модель класифікує об’єкти, порівнюючи передбачення з реальними мітками. Основна ідея полягає в тому, що матриця відображає кількість випадків, коли модель вгадала клас правильно, а також кількість помилок різних типів.

Ця матриця має розмірність, що відповідає кількості класів у задачі, і кожен її елемент відображає кількість прикладів, які належать до певного класу, але були класифіковані як інший. Наприклад, у двокласовій задачі можна легко побачити, скільки позитивних об’єктів було класифіковано як негативні, і навпаки. Завдяки цьому можна отримати детальнішу інформацію, ніж просто загальна точність моделі, і виявити, які саме помилки вона робить найчастіше.

Матриця неточностей є важливою для аналізу, оскільки допомагає виявити перекоси в роботі моделі, наприклад, переважання певного класу серед помилок або недостатню чутливість до рідкісних класів. Це дає змогу не лише оцінити ефективність класифікатора, але й знайти шляхи для її покращення, коригуючи модель або дані, на яких вона навчається.

3.4.3 Особливості оцінювання багатокласової класифікації

Оцінювання моделей багатокласової класифікації має свої особливості, пов’язані з тим, що в такій задачі модель одночасно розпізнає кілька класів замість двох. Це ускладнює процес вимірювання якості,

оскільки треба враховувати продуктивність моделі по кожному класу окремо, а також узагальнювати результати на весь набір даних. Для цього використовують різні метрики, які допомагають отримати повну картину роботи класифікатора.

Одними з ключових підходів до узагальнення метрик багатокласової класифікації є мікро- та макро-оцінки. Вони відрізняються способом обробки результатів по кожному класу і мають різні переваги.

Мікро-оцінка агрегує всі результати по класах, рахує загальні суми істинно позитивних, істинно негативних, хибно позитивних та хибно негативних результатів і на їх основі обчислює метрику. Такий підхід дає більшу вагу класам, які зустрічаються частіше, і тому мікро-оцінка є корисною, коли важливо оцінити загальну якість моделі по всіх прикладах, незалежно від розподілу класів.

Макро-оцінка навпаки спочатку обчислює метрики окремо для кожного класу, а потім знаходить їх середнє арифметичне без урахування частоти класів. Це означає, що макро-оцінка дає рівну вагу всім класам, навіть якщо деякі з них зустрічаються рідко. Такий підхід допомагає оцінити, наскільки модель добре працює на всіх класах, включно з менш представленими, і є особливо корисним, коли важливо, щоб модель була збалансованою і не ігнорувала рідкісні категорії.

3.5 Вибір моделей для задачі розпізнавання кібербулінгу

У межах цієї роботи розв’язується задача багатокласової класифікації текстів, що передбачає не лише виявлення факту наявності кібербулінгу, але й визначення його конкретного типу. Такий підхід дозволяє більш точно аналізувати зміст повідомлень і потенційно формувати диференційовані стратегії реагування.

Для вирішення задачі було обрано три моделі з різним рівнем складності. Базовою моделлю стала комбінація наївного байєсівського

класифікатора та TF-IDF-векторизації. Цей варіант є обчислювально ефективним, простим у реалізації та забезпечує прийнятну якість на базових задачах класифікації текстів. Завдяки своїй швидкості, він є зручним орієнтиром для оцінки продуктивності складніших моделей.

Для побудови двох інших моделей використовувалися векторні представлення слів, згенеровані за допомогою Word2Vec. Перша модель – це звичайна feed-forward нейронна мережа, що складається з декількох лінійних шарів із нелінійними активаціями. Вона дозволяє узагальнювати інформацію з фіксованих векторів і забезпечує гнучкість у налаштуванні структури мережі.

Друга модель – рекурентна нейронна мережа з використанням LSTM. Її перевага полягає в здатності ефективно враховувати послідовність слів у повідомленнях, що особливо важливо для виявлення латентних форм агресії або сарказму, де контекст може змінювати значення окремих слів.

Незважаючи на велику популярність і ефективність трансформерних моделей, таких як BERT, вони не були використані в рамках цієї роботи. Основна причина – висока обчислювальна складність і потреба у значних ресурсах для навчання, що виходить за межі цієї роботи.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ

4.1 Інструменти та бібліотеки

Для реалізації моделей було використано мову програмування Python, яка є однією з найпопулярніших у сфері машинного навчання та обробки природної мови завдяки великій кількості спеціалізованих бібліотек і простоті синтаксису.

Для обробки табличних даних використовувалась бібліотека pandas, яка забезпечує зручний інтерфейс для роботи з датафреймами, включаючи фільтрацію, об’єднання, групування та обчислення статистичних показників. Регулярні вирази для очищення тексту застосовувалися за допомогою модуля re, що дозволило ефективно видаляти зайві символи, посилання, емодзі та інші небажані елементи з тексту.

Попередня обробка текстів здійснювалась за допомогою бібліотеки nltk, яка надала інструменти для токенізації, видалення стоп-слів, стемінгу та лематизації.

Для візуалізації результатів використовувалися бібліотеки matplotlib і seaborn. Вони дозволили побудувати графіки розподілу класів, матриці неточностей та інші візуалізації, необхідні для аналізу даних і моделей.

Моделі класичного машинного навчання були реалізовані за допомогою бібліотеки scikit-learn , яка включає в себе готові реалізації таких алгоритмів, як логістична регресія, наївний байєс, дерева рішень, а також засоби для розбиття вибірки, побудови конвеєрів та обчислення метрик класифікації.

Для побудови нейронних мереж було використано бібліотеку PyTorch – один із провідних фреймворків глибинного навчання, який дозволяє зручно реалізовувати як прості, так і складні архітектури, а також оптимізувати їх параметри за допомогою GPU-прискорення.

Окремо для роботи з векторними представленнями слів була використана бібліотека Gensim, яка забезпечує ефективну реалізацію алгоритмів Word2Vec і підтримує збереження та завантаження попередньо навчених моделей.

4.2 Опис набору даних

Для побудови моделей виявлення кібербулінгу у цій роботі використано відкритий датасет, що містить понад 46000 повідомлень із соціальної мережі Twitter. Його створення стало відповіддю на стрімке зростання використання соціальних медіа, яке охоплює практично всі вікові групи та дедалі більше впливає на повсякденне спілкування. Разом із цим зростає і поширення кібербулінгу – агресивної чи дискримінаційної поведінки, яка завдає шкоди користувачам онлайн-середовища.

Особливу увагу до проблеми привернув період пандемії COVID-19, коли через масове закриття шкіл, збільшення часу перед екранами та зменшення особистої соціальної взаємодії ризики кібербулінгу значно зросли.

Набір даних, використаний у дослідженні, містить твіти, які класифіковано за типом кібербулінгу. Кожне повідомлення віднесено до однієї з наступних категорій:

- булінг за ознакою віку;
- булінг за етнічною приналежністю;
- булінг за статтю;
- булінг на релігійному ґрунті;
- інші форми кібербулінгу;
- повідомлення, які не є кібербулінгом.

З метою забезпечення рівноваги вибірки всі класи були збалансовані – кожен з них представлений приблизно 8000 повідомлень. Такий підхід

дозволяє уникнути зміщення моделей у бік більш представлених класів і дає змогу коректно порівнювати результати для різних типів кібербулінгу [28].

4.3 Попередня обробка даних

Перед початком побудови моделей виявлення кібербулінгу було здійснено ретельну попередню обробку датасету з метою забезпечення якості вхідних текстових даних. Насамперед було видалено записи, позначені як «інші форми кібербулінгу». Попри те, що вони також можуть містити агресивні чи образливі висловлювання, їхня неоднорідність і подібність до інших категорій ускладнює автоматичну класифікацію. Тому для зменшення шуму в даних та покращення якості навчання моделі було прийнято рішення зосередитися лише на чітко визначених п’яти класах.

Основна очистка тексту здійснювалася за допомогою спеціально створеної функції, яка включає в себе кілька послідовних етапів і наведена в додатку А. Ця функція виконує комплексну обробку: приводить всі символи до нижнього регістру, видаляє емодзі, скорочення автоматично розгортаються, а також фільтруються неангломовні записи. Крім того, з тексту прибираються посилання, хештеги, знаки пунктуації, зайві символи, стоп-слова та числа. Застосовується лематизація для приведення слів до базових форм, а також відсікаються надто короткі слова, що часто не несуть змістовного навантаження. Окремим етапом усуваються повтори однакових символів у словах, характерні для емоційних висловлювань у соцмережах.

Наступним кроком було проведення базового аналізу довжини текстів. Для цього візуалізовано розподіл кількості слів у кожному твіті (рисунок 4.1). Виявлено, що переважна більшість повідомлень є короткими – менше 30 слів. Щоб зменшити вплив рідкісних довгих твітів, було відкинуто 0.5% найдовших повідомлень.

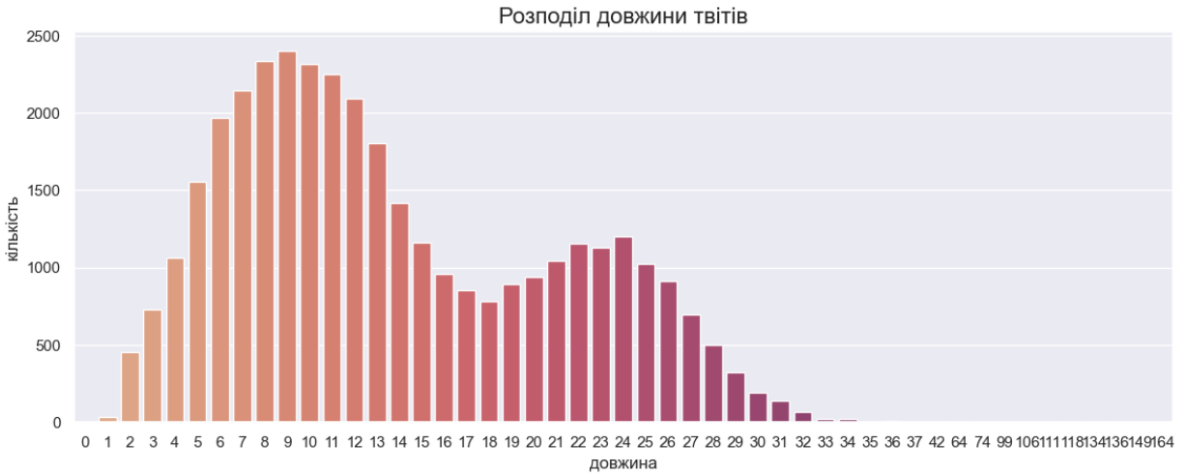

Рисунок 4.1 – Аналіз довжини текстів

Наступна візуалізація стосується розподілу класів у датасеті. Як видно з результатів (рисунок 4.2), кількість прикладів у кожному класі є майже рівномірною, що дозволяє уникнути проблеми дисбалансу класів при навчанні моделі.

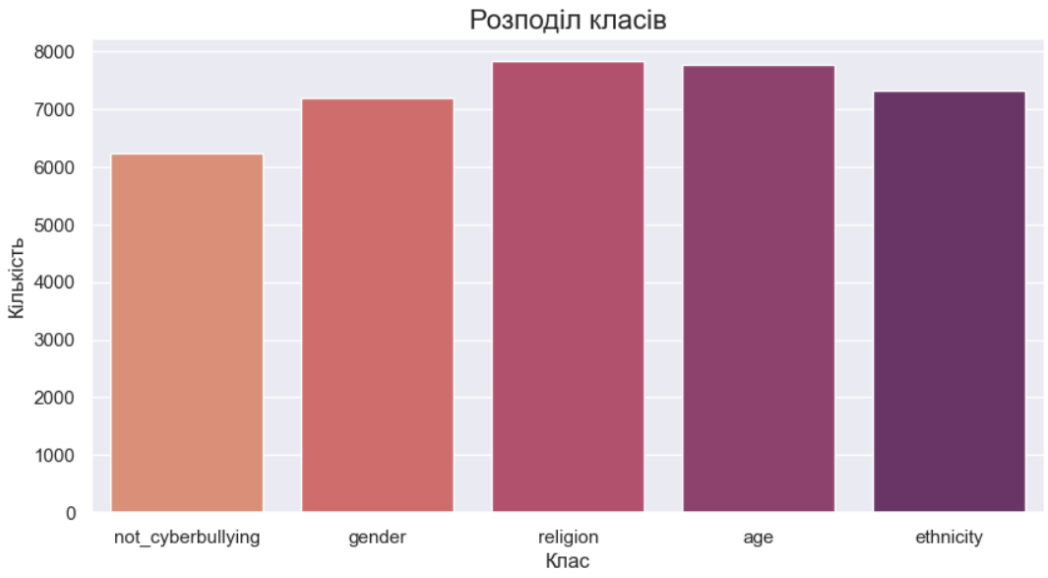

Рисунок 4.2 – Аналіз розподілу класів

Додатково було побудовано хмару слів для найчастіше вживаних термінів у повідомленнях, які не містять кібербулінгу (рисунок 4.3). Це

дозволяє краще зрозуміти типові патерни мови у нейтральних або доброзичливих твітах.

Рисунок 4.3 – Хмара слів для повідомлень, які не містять кібербулінгу

На завершення, очищений датасет було розділено на три частини: тренувальну, валідаційну та тестову вибірки. Такий поділ забезпечує об’єктивне оцінювання моделей на даних, які не використовувались під час навчання, що дозволяє коректно оцінити узагальнюючу здатність обраних підходів.

4.4 Векторизація тексту

Перед тим як передати текстові дані до моделей машинного навчання, їх необхідно перетворити у числовий формат. У цій роботі було використано два підходи до векторизації: TF-IDF як базовий метод та Word2Vec – для побудови більш гнучких моделей на основі семантичних ознак.

4.4.1 TF-IDF

Першим кроком було застосування методу TF-IDF, який дозволяє оцінити важливість кожного терміна в документі відносно всього корпусу.

Спочатку текст перетворюється у матрицю частот за допомогою CountVectorizer, після чого застосовується трансформер TfidfTransformer, який обчислює ваги термінів. Код перетворення наведений у лістингу 4.1.

Лістинг 4.1 – Програмний код застосування TF-IDF

```
clf = CountVectorizer()
X_train_cv = clf.fit_transform(X_train)
X_test_cv = clf.transform(X_test)
tf_transformer =
TfidfTransformer(use_idf=True).fit(X_train_cv)
X_train_tf = tf_transformer.transform(X_train_cv)
X_test_tf = tf_transformer.transform(X_test_cv)
```

Результатом є розріджена матриця, яка відображає вагу кожного слова у контексті документа – чим рідше слово трапляється, тим більшу вагу воно має.

4.4.2 Word2Vec

Другим підходом стала побудова ембедінгів слів за допомогою моделі Word2Vec. Цей метод дозволяє отримати векторне представлення слів у багатовимірному просторі, де подібні за змістом слова мають схожі координати.

Перш за все було реалізовано функцію Tokenize, яка виконує побудову словника, перетворення текстів у послідовності індексів, а також їх доповнення нулями до фіксованої довжини. Код функції наведений у лістингу 4.2.

Лістинг 4.2 – Програмний код токенізації текстів

```
def Tokenize(column, seq_len):
    corpus = [word for text in column for word in
text.split()]
```

Продовження лістингу 4.2.

```
        count_words = Counter(corpus)
        sorted_words = count_words.most_common()
        vocab_to_int = {w: i + 1 for i, (w, c) in
enumerate(sorted_words)}
        text_int = []
        for text in column:
            r = [vocab_to_int[word] for word in text.split()]
            text_int.append(r)
        features = np.zeros((len(text_int), seq_len),
dtype=int)
        for i, review in enumerate(text_int):
            if len(review) <= seq_len:
                zeros = list(np.zeros(seq_len - len(review)))
                new = zeros + review
            else:
                new = review[:seq_len]
            features[i, :] = np.array(new)
        return sorted_words, features
```

На основі токенізованих текстів було натреновано модель Word2Vec з розмірністю векторів 200. Далі формується ембедінг-матриця розміром (VOCAB_SIZE, EMBEDDING_DIM), яка містить векторне представлення кожного слова з лексикону. Якщо слово присутнє у словнику Word2Vec, його вектор вставляється у відповідний рядок матриці. Повний код наведений у лістингу 4.3.

Лістинг 4.3 – Програмний код застосування Word2Vec

```
    vocabulary, tokenized_column = Tokenize(df.text, max_len)
    Word2vec_train_data = list(map(lambda x: x.split(),
X_train))
    word2vec_model = Word2Vec(Word2vec_train_data,
vector_size=EMBEDDING_DIM)
    VOCAB_SIZE = len(vocabulary) + 1
```

Продовження лістингу 4.3.

```
embedding_matrix = np.zeros((VOCAB_SIZE, EMBEDDING_DIM))
for word, token in vocabulary:
    if word in word2vec_model.wv.key_to_index:
        embedding_vector = word2vec_model.wv[word]
        embedding_matrix[token] = embedding_vector
```

Отримана ембедінг-матриця буде згодом використана як ваги для першого шару у нейронних мережах, що дозволяє моделям опрацьовувати тексти з урахуванням семантичної близькості слів.

4.5 Побудова моделей

4.5.1 Наївний Байєс

Першою моделлю, яка була реалізована, стала модель наївного байєсівського класифікатора – Multinomial Naive Bayes. Вона застосовується до TF-IDF векторів, отриманих у попередньому кроці. Модель навчалася на тренувальному наборі даних і пізніше використовувалася для прогнозування на тестовому наборі. Код створення та навчання моделі наведений в лістингу 4.4.

Лістинг 4.4 – Програмний код створення та навчання наївного Байєса

```
nb_clf = MultinomialNB()
nb_clf.fit(X_train_tf, y_train)
```

4.5.2 Багатошарова лінійна мережа

Оскільки наступні моделі реалізовані на основі бібліотеки PyTorch, текстові дані були перетворені у формат, зручний для обробки нейронними мережами. Кожна підмножина – тренувальна, валідаційна та тестова – була

конвертована у TensorDataset, що дозволяє ефективно працювати з даними під час навчання за допомогою DataLoader.

Крім того, було встановлено розмір пакета, рівний 32, а також налаштовано параметри для перемішування даних і відкидання неповних батчів. Це дозволяє забезпечити стабільність навчання та контроль за розміром кожної ітерації. Програмний код створення TensorDataset та DataLoader поданий в лістингу 4.5.

Лістинг 4.5 – Програмний код створення TensorDataset та DataLoader

```
train_data = TensorDataset(torch.from_numpy(X_train), torch.from_numpy(y_train))
test_data = TensorDataset(torch.from_numpy(X_test), torch.from_numpy(y_test))
valid_data = TensorDataset(torch.from_numpy(X_valid), torch.from_numpy(y_valid))
train_loader = DataLoader(train_data, shuffle=True, batch_size=BATCH_SIZE, drop_last=True)
valid_loader = DataLoader(valid_data, shuffle=False, batch_size=BATCH_SIZE, drop_last=True)
test_loader = DataLoader(test_data, shuffle=False, batch_size=BATCH_SIZE, drop_last=True)
```

Наступна модель, яка була реалізована у межах глибокого навчання, – це багатошарова лінійна нейронна мережа, яка приймає на вхід послідовності токенів, представлені через Word2Vec-ембедінги. Архітектура моделі побудована таким чином, щоб кожне вхідне повідомлення конвертувалося в послідовність векторів, які потім об'єднуються у єдиний вхідний вектор.

Далі вектор проходить через кілька повнозв’язних шарів з активацією ReLU, нормалізацією та дропаутою для зменшення перенавчання. Завершальний шар – це лінійна проєкція на кількість класів, що відповідає задачі мультикласової класифікації.

Для навчання моделі використовувалась попередньо побудована ембедінг-матриця, сформована на основі навчання моделі Word2Vec. Вона була завантажена в шар Embedding, причому ваги були позначені як ті, що підлягають навчанню, що дозволяє адаптувати їх під специфіку конкретної задачі в процесі оптимізації. Програмний код архітектури моделі поданий у лістингу 4.6.

Лістинг 4.6 – Програмний код архітектури багатошарової лінійної мережі

```
class LinearClassifier(nn.Module):
    def __init__(self, vocab_size, embedding_dim, sequence_length, num_classes):
        super().__init__()
        self.embedding = nn.Embedding(vocab_size, embedding_dim)
        input_dim = sequence_length * embedding_dim
        self.model = nn.Sequential(
            nn.Flatten(), nn.Linear(input_dim, 1024),
            nn.BatchNorm1d(1024), nn.ReLU(), nn.Dropout(0.3),
            nn.Linear(1024, 512), nn.BatchNorm1d(512), nn.ReLU(),
            nn.Dropout(0.3), nn.Linear(512, 256), nn.BatchNorm1d(256),
            nn.ReLU(), nn.Linear(256, num_classes))
    def forward(self, x):
        x = self.embedding(x)
        return self.model(x)
```

З метою аналізу ефективності навчання моделі, а також виявлення можливих проблем, таких як перенавчання або недонавчання, було реалізовано візуалізацію історії навчання. Побудова графіків зміни втрат і точності на тренувальній і валідаційній вибірках дозволила наочно оцінити,

як модель навчається з епохи в епоху, і вчасно коригувати гіперпараметри у разі потреби. Код для побудови історії навчання наведений в лістингу 4.7.

Лістинг 4.7 – Програмний код побудови графіків історії навчання

```
    def plot_history(train_loss_history, valid_loss_history,
train_acc_history, valid_acc_history):
        plt.figure(figsize=(12, 5))
        plt.subplot(1, 2, 1)
        sns.lineplot(train_loss_history, label='Train Loss')
        sns.lineplot(valid_loss_history, label='Val Loss')
        plt.xlabel('Epoch')
        plt.ylabel('Loss')
        plt.title('Training and Validation Loss')
        plt.legend()
        plt.subplot(1, 2, 2)
        sns.lineplot(train_acc_history, label='Train Acc')
        sns.lineplot(valid_acc_history, label='Val Acc')
        plt.xlabel('Epoch')
        plt.ylabel('Accuracy (%)')
        plt.title('Training and Validation Accuracy')
        plt.legend()
```

Після побудови архітектури моделі розпочався етап її тренування. Було проведено повноцінний навчальний цикл із використанням кросентропійної функції втрат, яка є стандартною для задач багатокласової класифікації. В якості оптимізатора обрано Adam, оскільки він добре справляється з нестабільністю градієнтів, особливо в задачах обробки природної мови.

Модель навчалась протягом максимум 20 епох, однак для запобігання перенавчанню було впроваджено механізм ранньої зупинки. Якщо протягом 4 послідовних епох точність на валідаційній вибірці не покращувалась, тренування припинялося достроково. Це дозволяє уникнути

зайвих обчислювальних витрат і покращити узагальнюючу здатність моделі.

Під час кожної епохи виконувалися дві основні фази:

– у фазі навчання модель переводиться в режим навчання, після чого здійснюється прямий прохід вхідних даних, обчислення втрат, зворотнє поширення помилки та оновлення ваг за допомогою оптимізатора;

– на етапі валідації модель переводиться в режим оцінювання, вимикається обчислення градієнтів і виконується прогнозування на валідаційній вибірці.

Повний код створення та навчання моделі наведений в додатку Б.

В результаті навчання було побудовано графіки історії навчання, який наведений на рисунку 4.4.

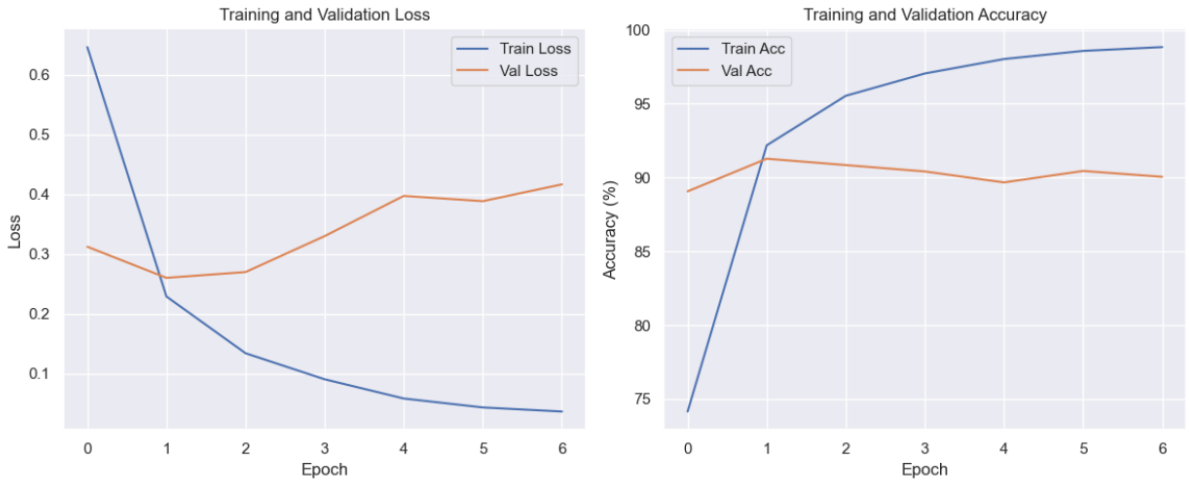

Рисунок 4.4 – Графіки історії навчання багатошарової лінійної мережі

На графіках зображено динаміку зміни втрат і точності протягом навчання моделі. З них можна зробити такі висновки:

– спостерігається стабільне зменшення втрат та зростання точності на тренувальних даних, що свідчить про успішне навчання моделі;

– після початкового покращення валідаційна точність починає знижуватись, а втрати – зростати, починаючи приблизно з третьої епохи.

Це є типовою ознакою перенавчання: модель добре запам’ятовує тренувальні дані, але погіршує узагальнення на валідаційному наборі. У цьому випадку доцільним було використання ранньої зупинки, що й було зроблено – тренування завершилось на 6-й епосі до подальшого погіршення.

4.5.3 LSTM мережа з механізмом уваги

Для побудови останньої моделі класифікації було реалізовано архітектуру на основі рекурентної нейронної мережі з довготривалою короткочасною пам’яттю, доповнену механізмом уваги. Такий підхід дозволяє враховувати послідовність слів у тексті та зосереджуватись на найважливіших його фрагментах, що є особливо корисним у задачах аналізу природної мови.

На вхід моделі подається закодований текст, який спочатку проходить через шар векторного представлення слів. Цей шар ініціалізовано вагами, отриманими попередньо з моделі Word2Vec. Далі ембеддінги надходять у LSTM-шар, який обробляє послідовність і формує приховані стани. Виходи LSTM передаються в механізм уваги, який обчислює ваги важливості для кожного елемента послідовності. Таким чином формується контекстне представлення тексту, що узагальнює найрелевантнішу інформацію. Отримане представлення подається на повнозв’язний шар, який виконує остаточну класифікацію, після чого до результату застосовується функція активації LogSoftmax. Код архітектури моделі наведений в додатку В.

На відміну від попередньої лінійної моделі, де використовувалась функція втрат CrossEntropyLoss, для LSTM-класифікатора було обрано NLLLoss. Такий вибір зумовлений тим, що вихідним шаром моделі є LogSoftmax, який повертає логарифмічні ймовірності для кожного класу. NLLLoss ідеально поєднується з такою активацією, оскільки вона очікує на вхід саме логарифмовані значення ймовірностей, що дозволяє забезпечити коректне обчислення втрат.

Крім того, для оптимізації ваг мережі замість класичного алгоритму Adam, який використовувався у попередній моделі, тут застосовано AdamW. Цей оптимізатор є модифікацією Adam з чітко відокремленим впливом регуляризації через L2-норму ваг. Такий підхід дозволяє уникнути надмірного зменшення навчальної швидкості, що часто трапляється в стандартному Adam при використанні регуляризації. Завдяки цьому AdamW демонструє кращу стабільність та узагальнювальну здатність у глибших або складніших моделях, зокрема тих, що містять рекурентні шари.

Сам процес навчання майже ніяк не відрізняється від процесу навчання попередньої моделі і наведений в додатку Г.

Після навчання було побудовано графіки історії навчання, які зображені на рисунку 4.5.

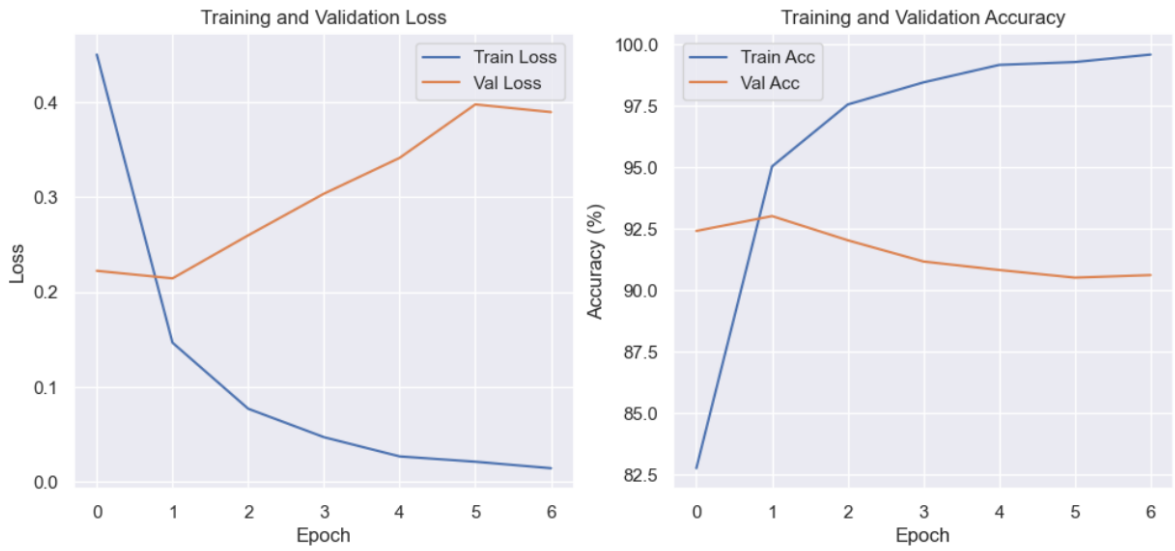

Рисунок 4.5 – Графіки навчання LSTM мережі з механізмом уваги

З аналізу результатів видно, що втрати на тренувальній вибірці послідовно зменшуються, а точність упевнено зростає, що свідчить про ефективне засвоєння моделлю навчальних прикладів. Водночас, починаючи приблизно з третьої епохи, валідаційна точність поступово знижується, а валідаційні втрати зростають – це свідчить про початок перенавчання.

Модель надто точно відтворює структуру тренувальних даних, втрачаючи здатність до узагальнення на нових прикладах. Щоб запобігти подальшому погіршенню результатів, було застосовано механізм ранньої зупинки, завдяки чому тренування завершилося на шостій епосі.

Порівнюючи цю модель з попередньою, можна зробити кілька важливих спостережень. У порівнянні багатошаровою лінійною мережею, LSTM-модель демонструє вищу точність на тренувальних даних, має менші втрати, а також швидше досягає високого рівня навчання. Проте ознаки перенавчання тут більш виражені.

4.6 Оцінка ефективності моделей

Для кількісної оцінки ефективності кожної з розроблених моделей було реалізовано спеціальну функцію evaluate_model, яка дозволяє отримати прогнозні значення на тестовій вибірці, порівняти їх із справжніми мітками та згенерувати детальний звіт за допомогою метрики classification_report з бібліотеки sklearn. Код функції наведений в лістингу 4.8.

Лістинг 4.8 – Програмний код функції оцінки моделей

```
def evaluate_model(model, test_loader):
    model.eval()
    y_pred_list = []
    y_test_list = []
    with torch.no_grad():
        for inputs, labels in test_loader:
            inputs, labels = inputs.to('cpu'),
labels.to('cpu')
            try:
                h = model.init_hidden(labels.size(0))
                output, _ = model(inputs, h)
```

Продовження лістингу 4.8.

```
                except AttributeError:
                    output = model(inputs)
                y_pred = torch.argmax(output, dim=1)
                y_pred_list.extend(y_pred.cpu().tolist())
                y_test_list.extend(labels.cpu().tolist())
        print(classification_report(y_test_list, y_pred_list,
digits=4))
        return y_pred_list, y_test_list
```

Крім того, для візуального аналізу якості класифікації було побудовано матриці неточностей для кожної моделі. Для цього створена функція conf_matrix, яка відображає співвідношення між реальними та передбаченими класами у вигляді теплової карти. Вона допомагає краще зрозуміти, які саме класи найчастіше плутає модель, та де саме виникають помилки. Код цієї функції поданий в лістингу 4.9.

Лістинг 4.9 – Програмний код функції для побудови матриці неточностей

```
    def conf_matrix(y, y_pred, title):
        labels = ['not_cyberbullying', 'gender', 'religion',
'age', 'ethnicity']
        fig, ax = plt.subplots(figsize=(7, 7))
        ax = sns.heatmap(confusion_matrix(y, y_pred),
                         annot=True,
                         cmap='flare',
                         fmt='g',
                         cbar=False)
        plt.title(title, fontsize=16)
        ax.xaxis.set_ticklabels(labels)
        ax.yaxis.set_ticklabels(labels,)
        ax.set_ylabel('True')
```

Продовження лістингу 4.9.

```
ax.set_xlabel('Predicted')
plt.show()
```

Для оцінки ефективності кожної з розроблених моделей були використані evaluate_model і conf_matrix. Це забезпечило однакові умови для аналізу продуктивності всіх моделей.

На основі класифікаційного звіту (рисунок 4.6) для моделі Наївного Байєса видно, що загальна точність класифікації на тестовій вибірці становить 83%, що є досить непоганим результатом з огляду на простоту моделі.

Classification Report for Naive Bayes:

	precision	recall	f1-score	support
0	0.86	0.33	0.48	1248
1	0.89	0.86	0.87	1437
2	0.83	0.97	0.90	1566
3	0.74	0.99	0.85	1554
4	0.90	0.91	0.91	1465
accuracy			0.83	7270
macro avg	0.84	0.81	0.80	7270
weighted avg	0.84	0.83	0.81	7270

Рисунок 4.6 – Класифікаційний звіт для наївного Байєсу

Однак, розглядаючи показники по окремих класах, можна помітити суттєвий дисбаланс у якості розпізнавання. Зокрема, клас 0 (not_cyberbullying) має низьке значення повноти – лише 33%, що свідчить про те, що модель не вміє ефективно виявляти повідомлення, які не є прикладами кібербулінгу, і часто класифікує їх неправильно як такі, що містять агресію. Це особливо критично, оскільки саме цей клас є фоновим і найважливішим для уникнення хибнопозитивних спрацювань. Цей факт також видно і на матриці неточностей, що зображена на рисунку 4.7.

Натомість класи 1–4, що відповідають різним типам кібербулінгу, розпізнаються значно краще. Наприклад, класи 2 та 3 демонструють дуже високі значення повноти (97% і 99% відповідно), що вказує на те, що модель майже не пропускає випадки відповідного булінгу. Водночас, висока F1-міра для класу 4 свідчать про хорошу збалансованість між правильними позитивними передбаченнями та виявленими істинними позитивами. На матриці неточностей також можна побачити, що для цих класів модель майже усі записи визначила правильно.

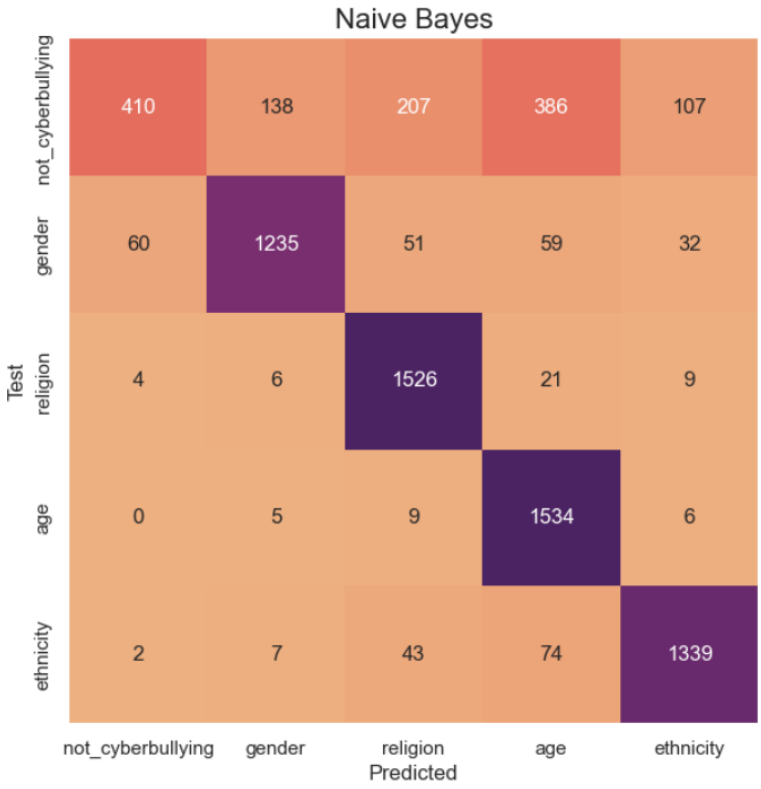

Рисунок 4.7 – Матриця неточностей для наївного Байєса

Таким чином, хоча модель наївного Байєса показує задовільні результати в цілому, її головний недолік полягає в низькій ефективності розпізнавання класу «не є кібербулінгом». Це обмежує її практичну застосовність у реальних сценаріях, де важливо не лише виявляти агресивні повідомлення, а й мінімізувати хибні тривоги.

Багатошарова лінійна модель продемонструвала високі результати за всіма основними метриками оцінювання якості класифікації (рисунок 4.8), значно перевершивши наївний Байєс. Загальна точність моделі склала 92.7%, що свідчить про її здатність ефективно розрізняти між собою п’ять класів, пов’язаних із різними типами контенту.

Classification Report for Linear model:

	precision	recall	f1-score	support
0	0.7568	0.9103	0.8265	1248
1	0.9594	0.8410	0.8963	1434
2	0.9674	0.9290	0.9478	1564
3	0.9755	0.9749	0.9752	1554
4	0.9807	0.9720	0.9763	1464
accuracy			0.9269	7264
macro avg	0.9280	0.9254	0.9244	7264
weighted avg	0.9341	0.9269	0.9284	7264

Рисунок 4.8 – Класифікаційний звіт для багатошарової лінійної мережі

Найкращі результати були досягнуті для класів 3 (вік) та 4 (етнічна належність), де значення f1-міри склали 0.9752 та 0.9763 відповідно. Це означає, що модель з великою впевненістю як правильно виявляє, так і не плутає приклади, що належать до цих категорій. Також дуже сильними виявились показники для класу 2, де f1-міра склала 0.9478. Ці три класи є найменш проблемними для моделі, що підтверджується як високими значеннями precision і recall, так і матрицею неточностей.

Клас 1 показав дещо нижчі, хоча все одно високі результати: precision – 0.9594, recall – 0.8410, f1 – 0.8963. Це свідчить про те, що модель рідко плутає інші приклади з цим класом, але часом не вловлює частину реальних випадків, що належать до нього.

Найслабшими виявилися показники для класу 0 – «не є кібербулінгом». Хоча recall тут досить високий і рівний 0.9103, тобто більшість прикладів класу 0 виявляються правильно, precision – лише

0.7568, що вказує на те, що серед передбачених випадків значна частина належить до інших класів. Це може бути пов’язано з тим, що модель у випадках невпевненості частіше відносить приклад до нейтрального класу, намагаючись уникнути помилкового виявлення булінгу.

Матриця неточностей (рисунок 4.9) демонструє, що найбільша кількість помилок спостерігається саме у випадках класу 0, який часто плутається з класами 1, 2 і 3. Наприклад, 35 прикладів класу 0 були класифіковані як клас 2, ще 37 – як клас 1. Водночас, помилок у межах класів 3 і 4 майже немає, що підкреслює високу стабільність роботи моделі для цих категорій.

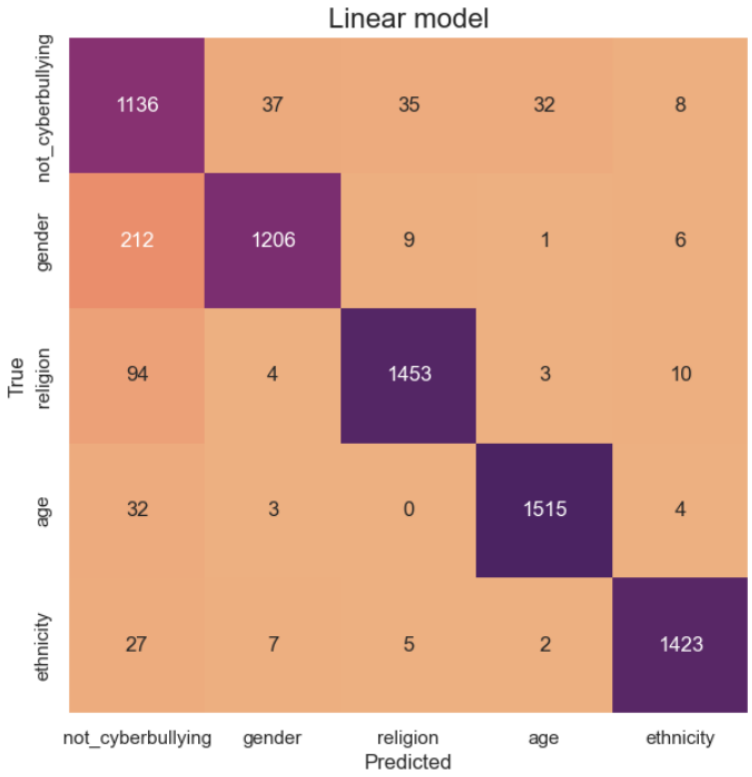

Рисунок 4.9 – Матриця неточностей для багатошарової лінійної мережі

LSTM-модель показала високі результати класифікації (рисунок 4.10), які практично не поступаються багатошаровій лінійній мережі, а за деякими показниками навіть дещо її перевищують. Загальна точність моделі

склала 92.86%, що свідчить про її здатність успішно узагальнювати знання на тестовому наборі даних, попри складність архітектури та тривалість навчання.

Найкращі результати досягнуті для класів 3 та 4, де f1-міра склала 0.9751 та 0.9776 відповідно. Це майже ідеальні значення, що демонструють надзвичайну точність і повноту розпізнавання цих типів кібербулінгу. Також дуже добре себе показала модель на класі 2 – f1-міра становить 0.9511, що перевищує відповідний показник лінійної моделі. Клас 1 було класифіковано з f1-мірою 0.8991 – показник дещо нижчий, ніж у лінійної мережі, але все ще свідчить про високий рівень якості розпізнавання.

Classification Report for LSTM model:

	precision	recall	f1-score	support
0	0.8154	0.8213	0.8184	1248
1	0.9100	0.8884	0.8991	1434
2	0.9445	0.9578	0.9511	1564
3	0.9786	0.9717	0.9751	1554
4	0.9729	0.9822	0.9776	1464
accuracy			0.9286	7264
macro avg	0.9243	0.9243	0.9243	7264
weighted avg	0.9285	0.9286	0.9285	7264

Рисунок 4.10 – Класифікаційний звіт для LSTM-моделі

Найменш впевнено модель поводиться з класом 0 – «не є кібербулінгом», хоча f1-міра тут також досить пристойна і рівна 0.8184, і є вищою, ніж у випадку з наївним байєсом. Основна проблема полягає в тому, що приклади цього класу доволі часто плутаються з іншими. Зокрема, матриця неточностей (рисунок 4.11) показує, що 109 прикладів класу 0 були класифіковані як клас 1, ще 67 – як клас 2. Це означає, що в деяких випадках нейтральні висловлювання помилково розпізнаються як прояви кібербулінгу, що може бути наслідком надмірної чутливості моделі.

Попри це, загальна картина є позитивною: модель демонструє чудовий баланс між точністю та повнотою для всіх п’яти класів, що підтверджується і збалансованими середніми значеннями.

У порівнянні з попередніми моделями, LSTM-класифікатор показує подібний рівень якості до багатошарової лінійної мережі, при цьому краще справляється з класами, де важлива послідовність і контекст. Це дозволяє зробити висновок, що додавання рекурентного механізму та механізму уваги позитивно впливає на здатність моделі вловлювати складні мовні залежності, що особливо корисно при аналізі текстових повідомлень у контексті кібербулінгу.

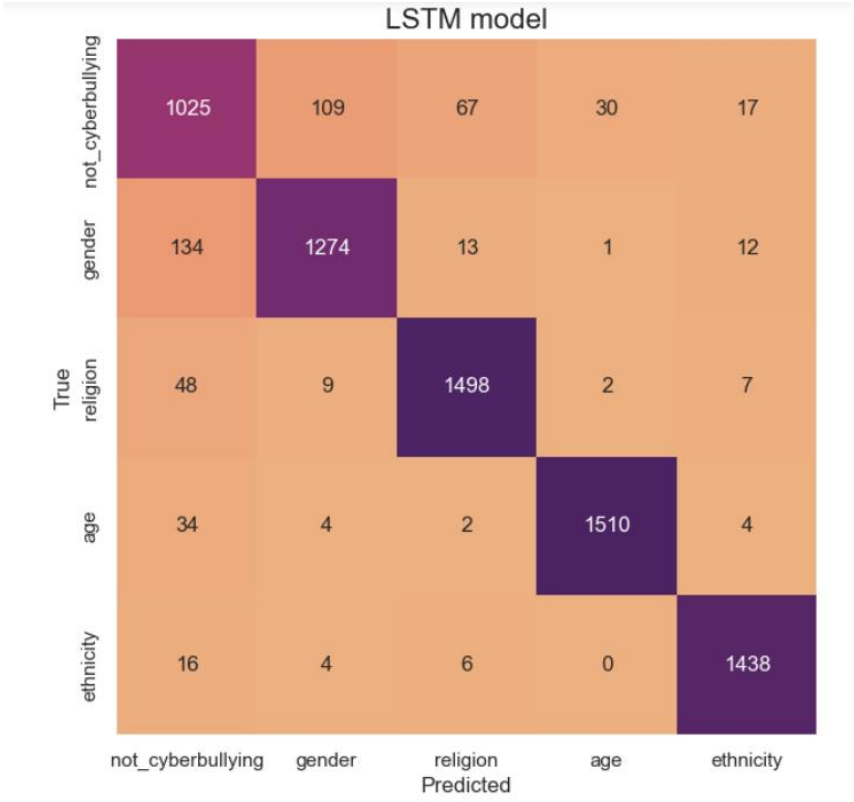

Рисунок 4.11 – Матриця неточностей для LSTM-моделі

Отже, модель наївного Байєса показала найнижчі результати серед усіх розглянутих варіантів. Зокрема, вона значно гірше справляється з класифікацією нейтральних повідомлень, маючи низьке значення recall у

цьому класі. Однак, як для простої моделі без складних механізмів навчання, її загальна точність на рівні понад 83% є цілком прийнятною.

Багатошарова лінійна мережа та LSTM-модель виявились значно ефективнішими. Обидві досягли точності понад 92%, демонструючи хорошу збалансованість між усіма класами. Лінійна модель трохи краще впоралася з нейтральними повідомленнями, тоді як LSTM точніше класифікувала повідомлення, пов’язані з віком та етнічністю. При цьому багатошарова мережа навчалася найдовше.

ВИСНОВКИ

У межах виконання кваліфікаційної роботи було проведено комплексне дослідження проблеми виявлення кібербулінгу на основі методів обробки природної мови. Було проаналізовано сутність явища кібербулінгу, виявлено його ключові риси, типи та визначено групи ризику, які найчастіше стають жертвами агресивної поведінки в інтернеті. Проведений огляд наукових джерел показав, що проблема є актуальною як у світовій, так і у вітчизняній практиці, а методи NLP активно використовуються для вирішення цієї проблеми.

У роботі було досліджено сучасні інструменти та технології, які застосовуються для побудови систем виявлення кібербулінгу. Зокрема, вивчено підходи до попередньої обробки тексту, токенізації, очищення даних, а також методи векторизації як спосіб перетворення тексту на числове представлення для машинного навчання. Окрему увагу приділено аналізу архітектур моделей, способам оптимізації навчання та оцінці результатів з використанням метрик precision, recall, f1-score, accuracy та матриць неточностей.

На основі досліджених методів було розроблено та порівняно три моделі класифікації текстів:

– наївний Байєс, як базова проста модель, показав загальну точність 83%, але мав суттєві труднощі з класифікацією нейтральних повідомлень;

– багатошарова лінійна нейронна мережа досягла точності понад 92%, продемонструвавши високу ефективність у класифікації всіх типів повідомлень, включаючи складні випадки;

– модель LSTM з увагою також досягла 92.86% точності, зокрема показавши кращу здатність вловлювати контекст і структуру речень, що робить її ефективною для реальних текстів.

Усі моделі були реалізовані та протестовані на основі одного узгодженого пайплайну, а результати були візуалізовані та інтерпретовані за допомогою класифікаційних звітів та матриць неточностей.

Варто зазначити, що повноцінна реалізація інтерактивного веб-сервісу для реального розгортання моделі не була виконана в межах цієї роботи, що можна розглядати як незначне обмеження. Проте це компенсується гнучкою архітектурою реалізованих моделей, які можуть бути легко адаптовані до будь-якої платформи або розширені для подальшого використання. Робота закладає фундамент для розширення у вигляді повноцінної системи модерації контенту.

Робота відповідає сучасним світовим тенденціям розвитку систем з розпізнавання шкідливого контенту, а застосовані підходи співвідносяться з передовими практиками галузі. Запропонований підхід демонструє, що навіть за обмежених ресурсів можливо досягти високої точності в задачах класифікації текстів агресивного змісту, що робить розробку цінною з точки зору практичного застосування. Проведене дослідження має зв’язок із науковою тематикою кафедри, зокрема в напрямах застосування ШІ для обробки природної мови, і може бути використано в навчальному процесі як приклад реалізації повного NLP-проєкту з практичним застосуванням.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Teenage Cyberbullying Statistics 2025 – BrightPath. *Bright Path Adolescent Mental Health*. URL: https://www.brightpathbh.com/teenage-cyberbullying-statistics/ (date of access: 06.05.2025).

2. Sayed F. R., Elnashar E. H., Omara F. A. Cyberbullying Detection in Social Media Using Natural Language Processing. *Scientific African*. 2025. P. e02713. URL: https://doi.org/10.1016/j.sciaf.2025.e02713 (date of access: 09.05.2025).

3. 67 % українських підлітків стикались з проблемою булінгу – ЮНІСЕФ | UACRISIS.ORG. *Uacrisis.org*. URL: https://uacrisis.org/uk/67-ukrayinskyh-pidlitkiv-stykalys-z-problemoyu-bulingu-yunisef (дата звернення: 09.05.2025).

4. Bottaro A. Cyberbullying: Negative Effects and How You Can Stop It. *Verywell Health*. URL: https://www.verywellhealth.com/cyberbullying-effects-and-what-to-do-5220584 (date of access: 11.04.2025).

5. Кожен п’ятий підліток визнавав себе жертвою онлайн-знущань, – дослідження про кібербулінг. *LB.ua*. URL: https://lb.ua/society/2020/10/09/467767_kozhen_pyatiy_pidlitok_viznavav.html (дата звернення: 09.05.2025).

6. Teens and Cyberbullying 2022. *Pew Research Center*. URL: https://www.pewresearch.org/internet/2022/12/15/teens-and-cyberbullying-2022/ (date of access: 09.05.2025).

7. Wright M. F. Adolescent Cyberbullies’ Attributions: Longitudinal Linkages to Cyberbullying Perpetration. *International Journal of Environmental Research and Public Health*. 2023.

8. Childhood Access to Technology and Cyberbullying. Journal of Pediatrics | *Pediatrics Journal*. URL: https://www.pediatricsresearchjournal.com/articles/childhood-access-to-technology-and-cyberbullying.html (date of access: 09.05.2025).

9. Social media addiction linked to cyberbullying. *UGA Today*. URL: https://news.uga.edu/social-media-addiction-linked-to-cyberbullying/ (date of access: 09.05.2025).

10. What is KiVa? | KiVa Program. *KiVa Program*. URL: https://www.kivaprogram.net/what-is-kiva/ (date of access: 09.05.2025).

11. Corlett E. New Zealand's prime minister proposes social media ban for under-16s. *the Guardian*. URL: https://www.theguardian.com/world/2025/may/06/new-zealand-pm-luxon-social-media-ban-children (date of access: 09.05.2025).

12. Program Profile: Prev@cib Program (Spain). CrimeSolutions, National Institute of Justice. URL: https://crimesolutions.ojp.gov/ratedprograms/prevcib-program-spain (date of access: 09.05.2025).

13. Malaysia seeks social media platforms' commitment to tackle cybercrimes. *Reuters*. URL: https://www.reuters.com/technology/malaysia-seeking-social-media-platforms-commitment-tackle-cybercrimes-2024-07-24/ (date of access: 09.05.2025).

14. Sliusarenko T., Pohurska M. Detecting cyberbullying with NLP. *Proceedings of the I International Scientific and Practical Conference*. Sofia, Bulgaria. 2025.

15. Philipo A. G., Sarwatt D. S., Ding J., Daneshmand M., Ning H.. Assessing Text Classification Methods for Cyberbullying Detection on Social Media Platforms. 2024. URL: https://doi.org/10.48550/arXiv.2412.19928 (date of access: 09.05.2025).

16. Cyberbullying Detection: Hybrid Models Based on Machine Learning and Natural Language Processing Techniques / C. Raj et al. *Electronics*. 2021. Vol. 10, no. 22. P. 2810. URL: https://doi.org/10.3390/electronics10222810 (date of access: 09.05.2025).

17. Gutiérrez-Batista K., Gómez-Sánchez J., Fernandez-Basso C. Improving automatic cyberbullying detection in social network environments by

fine-tuning a pre-trained sentence transformer language model. *Social Network Analysis and Mining*. 2024. Vol. 14, no. 1. URL: https://doi.org/10.1007/s13278-024-01291-0 (date of access: 09.05.2025).

18. Yi P., Zubiaga A., Long Y.. Detecting harassment and defamation in cyberbullying with emotion-adaptive training. 2025. URL: https://doi.org/10.48550/arXiv.2501.16925 (date of access: 09.05.2025).

19. Alqahtani A. F., Ilyas M. A Machine Learning Ensemble Model for the Detection of Cyberbullying. *International Journal of Artificial Intelligence & Applications*. 2024. Vol. 15, no. 1. P. 115–129. URL: https://doi.org/10.5121/ijaia.2024.15108 (date of access: 09.05.2025).

20. Kalidindi S. A Comprehensive guide for Natural Language Processing(NLP) for beginners. *Medium*. URL: https://medium.com/@saikirankalidindi/a-comprehensive-guide-for-natural-language-processing-nlp-for-beginners-39faa26ad4d9 (date of access: 21.05.2025).

21. Ashraf A. Tokenization in NLP: All you need to know. *Medium*. URL: https://medium.com/@abdallahashraf90x/tokenization-in-nlp-all-you-need-to-know-45c00cfa2df7 (date of access: 18.05.2025).

22. CR A. Word Embeddings in NLP | Word2Vec | GloVe | fastText. *Medium*. URL: https://medium.com/analytics-vidhya/word-embeddings-in-nlp-word2vec-glove-fasttext-24d4d4286a73 (date of access: 19.05.2025).

23. TF-IDF in NLP & How To Implement it in 4 Steps - Wisdom ML. *Wisdom ML*. URL: https://wisdomml.in/tf-idf-in-nlp-how-to-implement-it-in-4-steps/ (date of access: 18.05.2025).

24. Logistic Regression in Machine Learning. *Applied AI Blog*. URL: https://www.appliedaicourse.com/blog/logistic-regression-in-machine-learning/ (date of access: 21.05.2025).

25. Mienye I. D., Swart T. G., Obaido G. Recurrent Neural Networks: A Comprehensive Review of Architectures, Variants, and Applications.

Information. 2024. Vol. 15, no. 9. P. 517. URL: https://doi.org/10.3390/info15090517 (date of access: 23.05.2025).

26. KaNaung. Understanding LSTM: A Deep Dive into its Inner Workings with Code Walkthrough. *Medium*. URL: https://medium.com/@kanaung.academy/understanding-lstm-a-deep-dive-into-its-inner-workings-with-code-walkthrough-5337e4561e1b (date of access: 23.05.2025).

27. Vaswani A., Shazeer N., Parmar N. Attention Is All You Need. 2017. URL: https://doi.org/10.48550/arXiv.1706.03762 (date of access: 23.05.2025).

28. Cyberbullying Classification. *kaggle*. URL: https://www.kaggle.com/datasets/andrewmvd/cyberbullying-classification/data (date of access: 24.05.2025).