МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Західноукраїнський національний університет

Факультет комп'ютерних інформаційних технологій

Кафедра інформаційно-обчислювальних систем і управління

ЗАВОЙОВСЬКА Олександра Михайлівна

Програмний модуль для виявлення образливих слів та ненависті в текстових повідомленнях /
Software module for detecting offensive words and hatred in text messages

спеціальність: 122 - Комп'ютерні науки

освітньо-професійна програма - Комп'ютерні науки

Кваліфікаційна робота

Виконала студентка групи КН-41

О. М. Завойовська

Науковий керівник

к.т.н., Д. І. Загородня

Кваліфікаційну роботу допущено до захисту

«___»_____ 20___ р.

Завідувач кафедри

_____ М.П. Комар

ТЕРНОПІЛЬ – 2024

Факультет комп'ютерних інформаційних технологій

Кафедра інформаційно-обчислювальних систем і управління
Освітній ступінь «бакалавр»
спеціальність 122 – Комп'ютерні науки
освітньо-професійна програма – Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ М.П. Комар
«____» _____2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
ЗАВОЙОВСЬКА Олександра Михайлівна
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: Програмний модуль для виявлення образливих слів та ненависті в текстових повідомленнях / Software module for detecting offensive words and hatred in text messages
керівник роботи Загородня Діана Іванівна, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від 12 грудня 2023 р. № 753.
2. Строк подання студентом закінченої кваліфікаційної роботи 15 травня 2024 р.
3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.
4. Основні питання, які потрібно розробити:
- зробити аналіз предметної області;
- визначити вимоги до програмного модуля;
- розробити архітектуру програмного модуля;
- реалізувати алгоритми для виявлення образливих слів та ненависті;
- розробити програмне забезпечення;
- провести тестування та налагодження;
- здійснити документування;
- виконати оцінку результатів та зробити висновки.

5. Перелік графічного матеріалу в роботі:
- діаграма потоків даних;
- процес створення програмного модуля;
- діаграма станів.

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
	Загородня Д. І.		

7. Дата видачі завдання 12 грудня 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Затвердження теми кваліфікаційної роботи, ознайомлення з літературними джерелами та складання плану роботи.	до 01.01. 2024 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.03. 2024 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 01.04.2024 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 01.05. 2024 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 15.05.2024 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів.	до 20.05.2024 р.	
7	Перевірка кваліфікаційної роботи на оригінальність тексту у системі «Unicheck».	до 10.06.2024 р.	
8	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 14.06.2024 р.	
9	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії.	до 14.06. 2024 р.	

Студентка_____ О.М. Завойовська
(підпис) (прізвище та ініціали)

Керівник кваліфікаційної роботи_____ Д.І. Загородня
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота на тему «Програмний модуль для виявлення образливих слів та ненависті в текстових повідомленнях» на здобуття освітнього ступеня «бакалавр» зі спеціальності 122 «Комп’ютерні науки» освітньої програми «Комп’ютерні науки» написана обсягом в 51 сторінку і містить 13 ілюстрацій, 2 таблиці, 2 додатки та 19 використаних джерел.

Метою роботи є створення ефективного інструменту для автоматичного аналізу текстових даних з метою ідентифікації образливого контенту та висловлювань, що розпалюють ненависть.

У ході дослідження проведено аналіз існуючих методів і підходів до розпізнавання образливих слів і ненависних висловлювань, зокрема методів обробки природної мови (NLP) та алгоритмів машинного навчання. На основі цього аналізу були визначені вимоги до програмного модуля та розроблена його архітектура.

Основний код модуля реалізовано з використанням сучасних технологій та інструментів програмування. Розроблений модуль інтегровано з іншими системами, що дозволяє його використання в різних середовищах, таких як соціальні мережі та форуми. Для забезпечення зручності використання модуля створено інтуїтивно зрозумілий інтерфейс користувача.

Модуль пройшов тестування, результати якого підтвердили його ефективність у виявленні образливих слів та ненависті в текстах.

Результати дослідження можуть бути корисними для розробників програмного забезпечення, спеціалістів з інформаційної безпеки. Ключові слова: ПРОГРАМНИЙ МОДУЛЬ, МАШИННЕ НАВЧАННЯ, ВИЯВЛЕННЯ ОБРАЗЛИВИХ СЛІВ ТА НЕНАВИСТІ.

ANNOTATION

Qualification work on the topic “Software module for detecting offensive words and hatred in text messages” for the degree of bachelor in the specialty 122 “Computer Science” of the educational program “Computer Science” is written in 51 pages and contains 13 figures, 2 tables, 2 annexes and 19 sources.

The purpose of the work is to create an effective tool for automatic analysis of text data in order to identify offensive content and hate speech.

The study analyzed existing methods and approaches to recognizing offensive words and hate speech, including natural language processing (NLP) methods and machine learning algorithms. Based on this analysis, the requirements for the software module were determined and its architecture was developed.

The main code of the module was implemented using modern technologies and programming tools. The developed module is integrated with other systems, which allows it to be used in various environments, such as social networks and forums. An intuitive user interface has been created to ensure the module's ease of use.

The module has been tested, and the results have confirmed its effectiveness in detecting offensive words and hatred in texts.

The results of the study may be useful for software developers and information security specialists.

Keywords: SOFTWARE MODULE, MACHINE LEARNING, DETECTION OF OFFENSIVE WORDS AND HATRED.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

NLP - Natural language processing, обробка природної мови

DFD – Data Flow Diagrams, діаграма потоків даних

ERD – Entity-Relationship Diagrams, діаграма «суть-зв’язок»

STD - State Transition Diagrams, діаграма переходів станів

NLTK - Natural Language Toolkit, набір інструментів природної мови

LSTM - Довга короткочасна пам'ять, архітектура рекурентних нейронних мереж

ВСТУП

У сучасному інформаційному середовищі, що характеризується стрімким зростанням обсягів текстової інформації, питання виявлення та запобігання образливих слів та проявів ненависті в текстових повідомленнях стає все більш актуальним. Соціальні мережі, форуми, коментарі на сайтах новин, електронна пошта та інші платформи для обміну повідомленнями часто стають середовищем для поширення ненависницьких висловлювань, які можуть призвести до серйозних соціальних наслідків. Розробка програмного модуля для автоматичного виявлення таких проявів є важливим кроком у боротьбі з онлайн-насильством та кібербулінгом.

Проблема мови ненависті та образ у мережі є глобальною і торкається всіх аспектів суспільного життя. Соціальні платформи, форуми та чати часто стають майданчиками для розповсюдження ненависті, що може призвести до дискримінації, цькування та інших негативних явищ. Тому розробка ефективних інструментів для виявлення та блокування таких повідомлень є критично важливою.

Метою даної роботи є розробка та впровадження програмного модуля, який буде здатний ефективно ідентифікувати образливі слова та висловлювання ненависті у текстових повідомленнях. Зокрема, дослідження зосереджується на застосуванні сучасних методів обробки природної мови (NLP) та машинного навчання для створення точного і надійного інструменту, який зможе автоматично аналізувати великі обсяги текстових даних.

Об'єктом дослідження є текстові повідомлення в інтернеті та соціальних мережах, що можуть містити мову ненависті та образливі висловлювання. Це охоплює різноманітні платформи, такі як соціальні мережі, форуми, блоги, коментарі до новинних статей та інші онлайн-сервіси, де користувачі взаємодіють через текстовий контент.

Предметом дослідження є методи та алгоритми виявлення мови ненависті та образливих висловлювань у текстових повідомленнях за допомогою методів глибокого навчання. Це включає вивчення та застосування технологій обробки

природної мови (NLP), токенізації, лематизації, а також використання різних архітектур нейронних мереж, таких як RNN, LSTM та BERT, для класифікації тексту. Також досліджується процес навчання моделей, їх валідація, оптимізація та інтеграція в реальні системи модерації контенту.

Під час роботи необхідно вирішити кілька завдань. Перше завдання полягає в зборі та підготовці великого набору даних, що містять приклади мови ненависті та нейтральних повідомлень. Друге завдання включає розробку та навчання моделі глибокого навчання для класифікації тексту. Третє завдання полягає в оптимізації моделі для підвищення її точності та швидкості роботи. Четверте завдання включає інтеграцію розробленого модуля в реальні платформи для модерації контенту та забезпечення його регулярного оновлення для врахування нових форм вираження ненависті. Для досягнення поставленої мети дослідження були використані наступні методи: аналіз літератури, розробка алгоритмів NLP, машинне навчання, тестування та валідація

В результаті даної роботи буде створено програмний модуль, який буде здатний з високою точністю виявляти образливі слова та висловлювання ненависті в текстових повідомленнях. Такий модуль може бути використаний для автоматизованої модерації контенту в соціальних мережах, на форумах, у системах електронної пошти та інших платформах, що сприятиме зменшенню кількості випадків кібербулінгу та поширення ненависті онлайн.

Розробка програмного модуля для виявлення образливих слів та ненависті в текстових повідомленнях є важливим кроком у забезпеченні безпеки та комфортного середовища для користувачів інтернету. Застосування сучасних технологій машинного навчання та обробки природної мови дозволяє створити ефективні інструменти, які можуть значно зменшити негативний вплив онлайн-насильства.

РОЗДІЛ 1. АНАЛІЗ ПРОГРАМНОГО МОДУЛЯ

1.1 Опис предметної області

Тема кваліфікаційної роботи охоплює розробку програмного модуля для виявлення образливих слів та ненависті в текстових повідомленнях. Опис предметної області включає кілька важливих аспектів. Модуль призначений для аналізу текстових даних з метою ідентифікації образливих висловлювань та мови ненависті, що є критичним для забезпечення безпеки та комфортного середовища для користувачів інтернет-платформ, таких як соціальні медіа та форуми.

Модуль використовує методи обробки природної мови (NLP) та машинного навчання для автоматичного розпізнавання негативного контенту. Процес починається з попередньої обробки тексту, яка включає видалення URL-адрес, згадок користувачів, хештегів та непотрібних символів. Потім текст переводиться в нижній регістр і розбивається на окремі слова (токенізація). Кожне слово проходить через процес лематизації, що дозволяє звести його до базової форми.

Після попередньої обробки текстові дані використовуються для навчання моделей класифікації. Навчання здійснюється на великих наборах даних, що містять приклади ненависницьких висловлювань. Це дозволяє модулю ефективно розпізнавати подібні вирази в нових текстах. Моделі, як правило, включають в себе різні типи нейронних мереж, зокрема, рекурентні нейронні мережі (RNN) та довго-короткочасну пам'ять (LSTM), які добре підходять для обробки послідовних даних, таких як текст.

Результати класифікації текстів дозволяють виявляти потенційно шкідливі повідомлення. Коли повідомлення ідентифікується як ненависне або образливе, можуть бути вжиті різні заходи, такі як видалення контенту, блокування користувачів або повідомлення адміністраторам платформи. Це допомагає підтримувати безпечне та ввічливе середовище для всіх користувачів.

Розробка такого модуля включає також тестування та оцінку ефективності моделей класифікації. Це включає порівняння прогнозів моделі з реальними даними, аналіз показників точності, повноти та інших метрик, що характеризують продуктивність алгоритму.

Таким чином, програмний модуль для виявлення образливих слів та ненависті в текстових повідомленнях є важливим інструментом для забезпечення безпеки користувачів в онлайн-просторі. Використання сучасних методів NLP та машинного навчання дозволяє ефективно ідентифікувати негативний контент та вживати відповідні заходи для його обмеження.

Обробка природної мови (NLP) є критично важливою для успішного функціонування системи. Процес починається з попередньої обробки тексту, що включає видалення непотрібних символів, таких як URL-адреси, згадки користувачів та хештеги. Текст також перетворюється в нижній регістр для забезпечення одноманітності. Далі текст очищається від знаків пунктуації та спеціальних символів.

Наступним кроком є токенізація, де текст розбивається на окремі слова або токени. Кожне слово проходить через процес лематизації за допомогою WordNetLemmatizer, що дозволяє звести слово до його базової або кореневої форми, наприклад, "running" стає "run". Це допомагає зменшити кількість різних форм одного і того ж слова, що спрощує аналіз. Водночас видаляються стоп-слова, такі як "is", "the", "and", які не несуть значного смислового навантаження.

Після попередньої обробки текст перетворюється у числовий формат для подальшого аналізу за допомогою методів машинного навчання. Для цього використовується токенізатор, який створює словник з унікальних слів у тексті і замінює кожне слово на його числовий індекс. Ці числові представлення текстів потім використовуються для навчання моделей машинного навчання, таких як нейронні мережі.

Процес навчання моделі включає використання великих обсягів даних, що містять як приклади ненависницьких, так і нейтральних висловлювань. Модель навчається розпізнавати закономірності та характеристики, притаманні текстам з ненависницьким контентом. Для цього можуть використовуватися різні типи нейронних мереж, включаючи рекурентні нейронні мережі (RNN) та довго-короткочасну пам'ять (LSTM), які добре підходять для роботи з послідовними даними.

Після навчання модель тестується на нових текстах, щоб оцінити її ефективність у розпізнаванні ненависницьких висловлювань.

Таким чином, обробка природної мови є складним, багатокроковим процесом, що включає очищення, токенізацію, лематизацію та перетворення тексту в числовий формат, який можна використовувати для навчання моделей машинного навчання з метою виявлення образливих та ненависницьких висловлювань.

Модуль використовує методи машинного навчання для ефективного розпізнавання ненависницьких висловлювань. Спочатку текстові дані проходять попередню обробку, потім ці оброблені дані перетворюються в числові послідовності за допомогою токенізатора.

Для навчання моделі використовуються нейронні мережі, зокрема, LSTM (Long Short-Term Memory), які добре працюють з послідовними даними. Модель навчається на великому обсязі даних, що містять як приклади ненависницьких, так і нейтральних текстів. Після навчання модель може прогнозувати клас тексту, використовуючи двошаровий архітектурний підхід з Embedding і LSTM шарами.

Модель навчається на розділених даних (тренувальні та тестові набори) для оцінки її продуктивності. Процес навчання включає епохи та використання функції втрат categorical_crossentropy для оптимізації.

Таким чином, методи машинного навчання дозволяють модулю ефективно розпізнавати ненависть і образи в текстових повідомленнях, що підвищує безпеку та забезпечує кращий моніторинг соціальних платформ.

Інформаційна безпека програмного модуля є не менш важливою для забезпечення конфіденційності, цілісності та доступності даних. Модуль обробляє текстові повідомлення, які можуть містити конфіденційну інформацію, тому всі дані повинні бути захищені від несанкціонованого доступу.

Для забезпечення інформаційної безпеки використовується шифрування даних під час передачі та зберігання. Всі текстові повідомлення, які обробляються модулем, проходять через безпечні канали передачі даних. Використовується шифрування на стороні сервера для зберігання даних у базах даних.

Контроль доступу є ще одним важливим аспектом, який включає аутентифікацію користувачів та авторизацію дій, що дозволяє тільки авторизованим користувачам доступ до обробки даних. Ведення журналів дій користувачів та моніторинг підозрілої активності допомагає виявляти та запобігати потенційним загрозам безпеці.

Для захисту від шкідливих програм та атак, таких як SQL-ін'єкції або міжсайтовий скриптинг, застосовуються належні засоби кібербезпеки. Регулярні оновлення програмного забезпечення та безпекових патчів підтримують модуль у захищеному стані, знижуючи ризик експлуатації вразливостей.

Інформаційна безпека також забезпечується за рахунок регулярних аудитів безпеки та тестування проникнення, що дозволяє ідентифікувати та виправляти потенційні проблеми до того, як вони будуть використані зловмисниками. Це дозволяє підтримувати високий рівень безпеки і захищати конфіденційність та цілісність оброблюваних даних.

Лінгвістичний аналіз для програмного модуля включає декілька ключових кроків. Перший етап – це попередня обробка тексту, яка передбачає видалення зайвих символів, посилань, тегів і згадок користувачів. Це робиться для очищення тексту від непотрібної інформації, що може вплинути на результати аналізу. Використовуються регулярні вирази для видалення небажаних символів, а також приведення всіх слів до нижнього регістру.

Другий етап включає видалення стоп-слів, тобто слів, які не несуть значущої інформації для аналізу, таких як "і", "або", "але". Це дозволяє зменшити обсяг даних та зосередитися на значущих словах.

Наступним кроком є лемматизація, яка полягає в приведенні слів до їх базової форми. Це допомагає зменшити кількість унікальних слів, що підвищує ефективність подальшої обробки. Для цього використовується WordNetLemmatizer з бібліотеки NLTK.

Токенізація тексту передбачає розбиття тексту на окремі слова або токени. Цей процес є важливим для подальшої векторизації, де слова перетворюються у числові представлення, придатні для обробки алгоритмами машинного навчання.

Лінгвістичний аналіз також може включати виявлення частин мови (POS tagging) та розпізнавання сутностей (NER), що дозволяє визначити граматичні ролі слів у реченнях і виділити іменовані сутності, такі як імена, місця, організації.

В результаті лінгвістичного аналізу текст перетворюється у векторне представлення, яке можна використовувати для навчання моделі машинного навчання, що виявляє образливі слова та мову ненависті. Це забезпечує точність і надійність аналізу текстових повідомлень у програмному модулі.

Опис етичних аспектів включає три ключові аспекти. По-перше, важливо враховувати конфіденційність даних. Обробка текстових повідомлень користувачів повинна відбуватися в умовах, які гарантують захист особистої інформації від несанкціонованого доступу. Дані повинні бути анонімізовані, щоб забезпечити збереження конфіденційності користувачів.

По-друге, необхідно забезпечити справедливість і точність виявлення образливих слів і мови ненависті. Модель повинна бути навчена на різноманітних даних, щоб уникнути упередженості щодо будь-якої соціальної групи, раси, статі або релігії. Важливо уникати хибних спрацьовувань, які можуть призвести до необґрунтованих звинувачень або блокування користувачів.

По-третє, слід враховувати можливі наслідки використання модуля. Наприклад, рішення про блокування користувачів або видалення контенту повинні бути обґрунтованими і прозорими. Користувачі повинні мати можливість оскаржувати рішення, прийняті автоматизованою системою.

Нарешті, важливо забезпечити відкритість та прозорість процесу розробки та використання модуля. Користувачі повинні бути інформовані про те, як працює модуль, які дані збираються і як вони використовуються. Це сприятиме підвищенню довіри до технології та її етичного використання.

Розробка програмного модуля включає вибір відповідних технологій, таких як мови програмування (наприклад, Python), бібліотеки для обробки природної мови (наприклад, NLTK, SpaCy), та фреймворки для машинного навчання

(наприклад, TensorFlow, PyTorch). Також важливо забезпечити інтеграцію модуля з існуючими системами, такими як веб-платформи або мобільні додатки.

Розглянуті аспекти предметної області допоможуть зрозуміти контекст та значення розробки програмного модуля для виявлення образливих слів та ненависті в текстових повідомленнях, а також визначити основні виклики та завдання, що стоять перед розробниками в цій сфері.

1.2 Огляд і аналіз існуючих аналогів

Для розробки програмного модуля, корисно розглянути вже існуючі рішення на ринку. Існує кілька популярних інструментів та систем, які вирішують цю проблему, кожен з яких має свої особливості та підходи до аналізу тексту.

Інструменти обробки природної мови (NLP) наведені нижче. Google Perspective API (Рисунок 1.1). Є одним з найвідоміших інструментів для виявлення токсичного контенту. Він використовує моделі машинного навчання для аналізу коментарів і оцінки їхньої токсичності за шкалою. API може виявляти різні види образливого контенту, такі як токсичні коментарі, особисті атаки, ненависницькі висловлювання та інше.

До переваг відносять: простоту інтеграції через API, регулярні оновлення та підтримка з боку Google високу точність виявлення токсичних висловлювань.

До недоліків: залежність від зовнішнього сервісу та можливі проблеми з конфіденційністю даних.

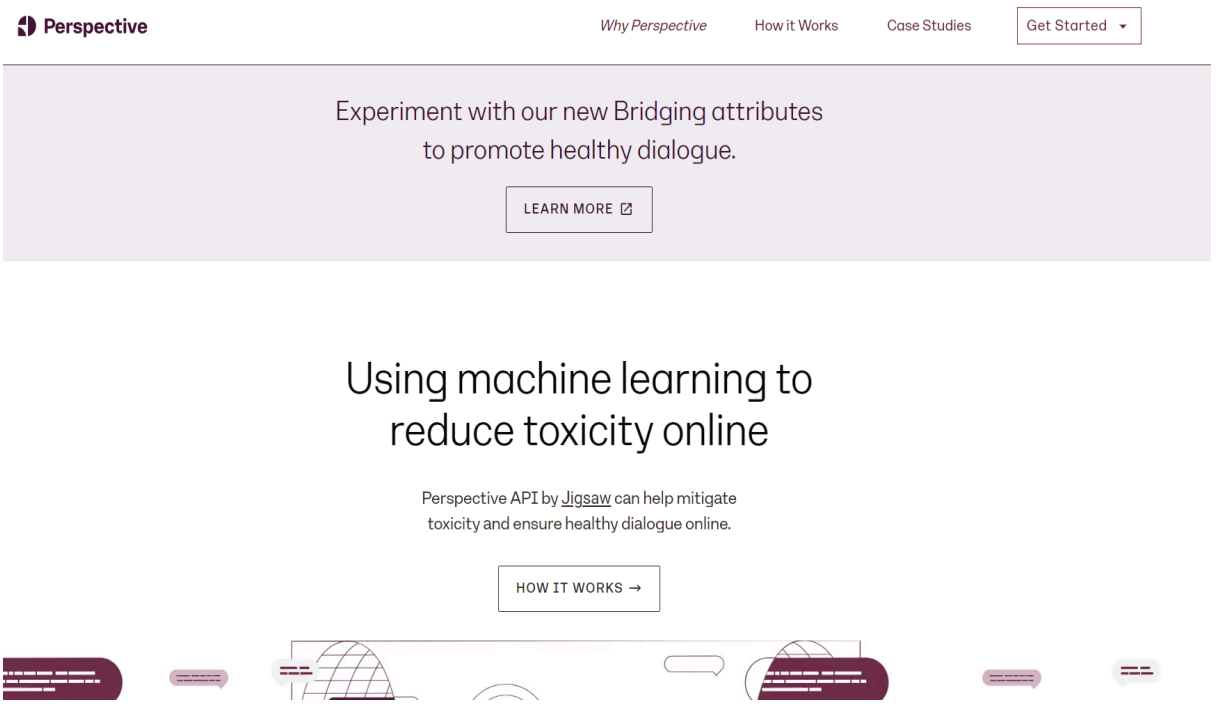

Рисунок 1.1 - Головна сторінка perspectiveapi.com

IBM Watson NLU (Рисунок 1.2) пропонує широкий спектр інструментів для аналізу тексту, включаючи виявлення тональності, категоризацію тексту та виявлення токсичних висловлювань. Watson використовує передові методи машинного навчання та аналізу природної мови. Його перевагами вважають: Гнучкість і можливість налаштування під конкретні завдання, підтримку багатьох мов та інтеграція з іншими сервісами Watson. Недоліки: вартість використання сервісу, вона може бути високою та складність налаштування для непрофесійних користувачів.

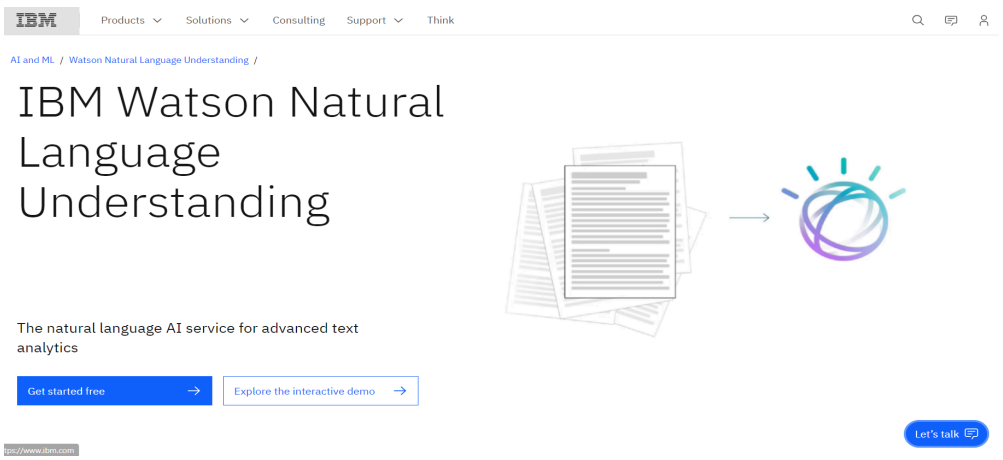

Рисунок 1.2 - Головна сторінка ibm.com

TextRazor (Рисунок 1.3) є ще одним інструментом для обробки природної мови, який пропонує функції виявлення образливих слів та ненависті. Він може аналізувати текст в режимі реального часу і надавати різноманітні дані про структуру та зміст тексту.

Переваги:

- Швидкість обробки тексту.
- Висока точність і надійність.
- Підтримка багатьох мов.

Недоліки:

- Потребує підключення до зовнішнього сервісу.
- Може бути дорогим для великих обсягів тексту.

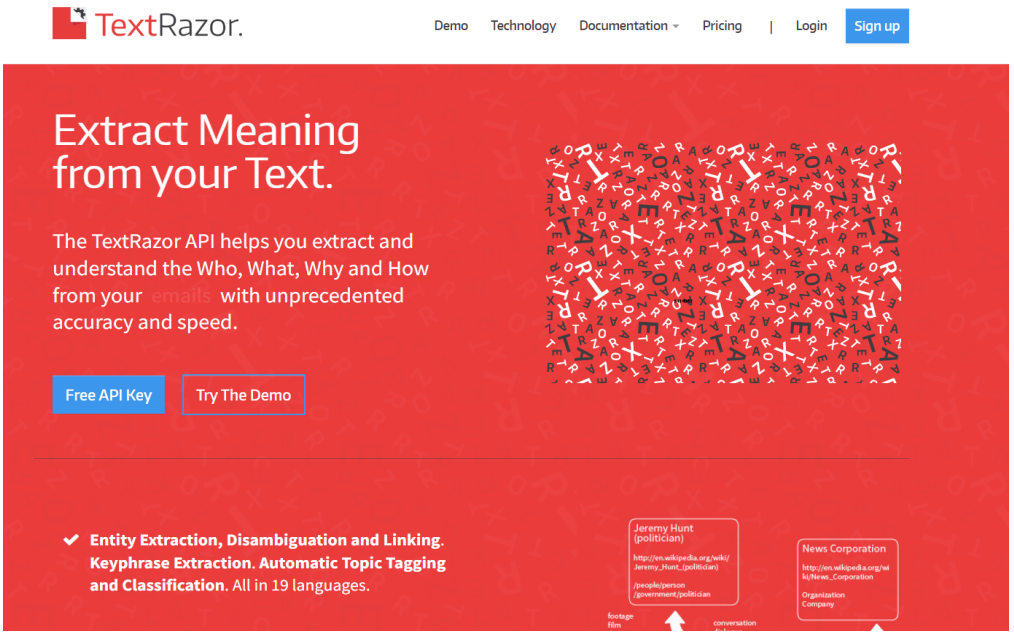

Рисунок 1.3 - Головна сторінка textrazor.com

Нижче наведені алгоритми і моделі машинного навчання. Так розроблений Facebook AI Research, є бібліотекою для векторного подання слів та текстової класифікації. FastText може використовуватися для виявлення образливих слів завдяки можливості швидко навчати моделі на великих корпусах тексту.

Переваги включають:

- Високу швидкість навчання і класифікації.
- Простота використання.
- Можливість роботи з багатьма мовами.

Недолік: обмежена гнучкість в порівнянні з більш складними моделями глибокого навчання.

BERT (Bidirectional Encoder Representations from Transformers). BERT, розроблений Google, є моделлю глибокого навчання, яка показала високі результати у багатьох завданнях NLP, включаючи виявлення токсичного контенту. BERT враховує контекст слів у реченні, що дозволяє точніше виявляти образливі висловлювання.

Перевагами є: висока точність завдяки врахуванню контексту та гнучкість для різних завдань NLP.

До недоліків додають:

- Високі вимоги до обчислювальних ресурсів.
- Складність налаштування і навчання моделей.

Також такі алгоритми використовують платформи соціальних мереж. Однією з більш популярних платформ є Facebook. Facebook використовує власні алгоритми машинного навчання для виявлення і блокування ненависті та образливих коментарів. Алгоритми постійно удосконалюються за допомогою великих обсягів даних, які надаються користувачами.

Інша платформа, не менш популярна, - Twitter. Також використовує алгоритми машинного навчання для виявлення ненависті та образливого контенту. Компанія активно працює над поліпшенням своїх моделей і впровадженням нових технологій для захисту користувачів.

Аналіз існуючих аналогів показує, що на ринку є багато різних рішень для виявлення образливих слів та ненависті в текстових повідомленнях. Вони варіюються від простих бібліотек до складних платформ соціальних мереж. Основні виклики включають забезпечення високої точності, обробку багатомовного контенту, збереження конфіденційності даних і етичний підхід до модерації контенту.

Виходячи з огляду і аналізу існуючих рішень, можна визначити ключові аспекти, які слід врахувати при розробці власного програмного модуля:

1. Висока точність і швидкість обробки тексту.

2. Гнучкість і можливість налаштування під специфічні потреби.
3. Забезпечення конфіденційності даних користувачів.
4. Можливість інтеграції з іншими системами та платформами.

Цей аналіз допоможе у визначенні найкращих підходів і технологій для створення ефективного програмного модуля, що відповідатиме сучасним вимогам і стандартам.

1.3 Постановка задачі дослідження

У сучасному світі стрімке зростання обсягів текстової інформації в Інтернеті, соціальних мережах, форумах і коментарях на різних платформах підвищує необхідність автоматичного моніторингу та виявлення ненависницьких і образливих висловлювань. Це важливо не лише для підтримки етичних норм спілкування, але й для забезпечення безпеки та комфорту користувачів. Програмний модуль для виявлення образливих слів та ненависті в текстових повідомленнях може значно полегшити цю задачу.

Метою цього дослідження є розробка програмного модуля, який зможе автоматично виявляти ненависницькі та образливі слова в текстових повідомленнях з високою точністю. Це дозволить своєчасно реагувати на такі повідомлення та вживати відповідних заходів для зменшення токсичності в онлайн-комунікаціях.

У зв'язку з ростом кількості користувачів соціальних мереж та інших онлайн-платформ, де користувачі можуть вільно обмінюватися повідомленнями, виникає необхідність у системах, що здатні ефективно виявляти і блокувати образливий контент. Така система допоможе зменшити рівень онлайн-насильства і забезпечити безпечне середовище для всіх користувачів.

Об'єктом дослідження є текстові повідомлення, опубліковані в онлайн-середовищах (соціальні мережі, форуми, коментарі до статей і т.д.).

До основних завдань дослідження входять:

- Аналіз існуючих підходів та інструментів: проведення огляду літератури та існуючих рішень у сфері виявлення ненависті та образливих висловлювань у текстових повідомленнях, а також визначення основних методів та алгоритмів, що використовуються для вирішення цієї проблеми.
- Розробка концептуальної моделі програмного модуля: для цього ми повинні визначити архітектуру програмного модуля, включаючи основні компоненти та їх взаємодію, а також розробити алгоритм для виявлення ненависницьких та образливих висловлювань на основі аналізу природної мови (NLP).
- Розробка та навчання моделей машинного навчання: збір та підготовка корпусу текстових даних, що містять приклади ненависницьких та образливих висловлювань, навчання моделі машинного навчання для класифікації текстових повідомлень, оцінка продуктивності моделей за допомогою метрик точності, повноти та інших.
- Інтеграція та тестування програмного модуля: розробка програмного забезпечення, що реалізує розроблені моделі та алгоритми та проведення тестування програмного модуля на різних платформах для оцінки його ефективності та надійності.
- Оптимізація та вдосконалення: проведення оптимізації моделей та алгоритмів для підвищення продуктивності та вдосконалення модуля на основі зворотного зв'язку та результатів тестування.
- Документування та підготовка рекомендацій: підготовка технічної документації та інструкції з використання програмного модуля та розробка рекомендації для подальшого вдосконалення та розширення функціональності.

Основною мовою програмування для розробки модулю буде Python, завдяки його багатій екосистемі бібліотек для глибинного навчання та обробки природної мови, таких як TensorFlow, Keras та Hugging Face Transformers.

В результаті виконаної роботи очікується створення ефективного програмного модуля, який зможе автоматично виявляти ненависницькі та образливі висловлювання в текстових повідомленнях з високою точністю.

Забезпечення можливості інтеграції модуля з різними онлайн-платформами для моніторингу та модерації контенту та зниження рівня токсичності в онлайн-комунікаціях, що сприятиме покращенню соціального клімату в Інтернеті.

Розробка програмного модуля для виявлення образливих слів та ненависті в текстових повідомленнях є актуальною та важливою задачею в сучасному світі. Вона спрямована на покращення якості онлайн-комунікацій та створення безпечного середовища для всіх користувачів.

РОЗДІЛ 2. АЛГОРИТМІЧНЕ ТА ІНФОРМАЦІЙНЕ ЗАБЕЗПЕЧЕННЯ МОДУЛЯ

2.1 Загальна структура модуля

На рисунку 2.1 представлено діаграму потоків даних, що відображає процес роботи програмного модуля для виявлення образливих слів та ненависті в текстових повідомленнях та дані які використовуються для цього. Діаграма складається з кількох основних етапів, кожен з яких виконує конкретну функцію у системі.

Користувач вводить текстове повідомлення, яке надходить до модуля введення даних. Цей модуль відповідає за первинне прийняття та перевірку коректності введеного тексту. Після цього текст проходить через модуль попередньої обробки, де здійснюється очищення даних, нормалізація тексту та підготовка його до аналізу.

Наступним етапом є модуль аналізу тексту, який проводить детальний аналіз введеного повідомлення, використовуючи алгоритми обробки природної мови та методи машинного навчання. Проаналізований текст надходить до етапу виявлення образливих слів та ненависті, де приймається рішення щодо наявності або відсутності образливих висловлювань у повідомленні.

Якщо виявлено образливі слова, модуль оповіщення користувача надсилає відповідне повідомлення користувачу, сповіщаючи його про небажаний контент. У випадку, якщо образливі слова не виявлено, текст зберігається у модулі зберігання чистого тексту, який взаємодіє з базою даних чистих текстів. Це дозволяє системі накопичувати та зберігати всі перевірені та безпечні текстові повідомлення.

Таким чином, діаграма наочно демонструє послідовність обробки текстових повідомлень у програмному модулі, починаючи з введення даних користувачем і закінчуючи або оповіщенням користувача про образливий контент, або зберіганням безпечного тексту в базі даних.

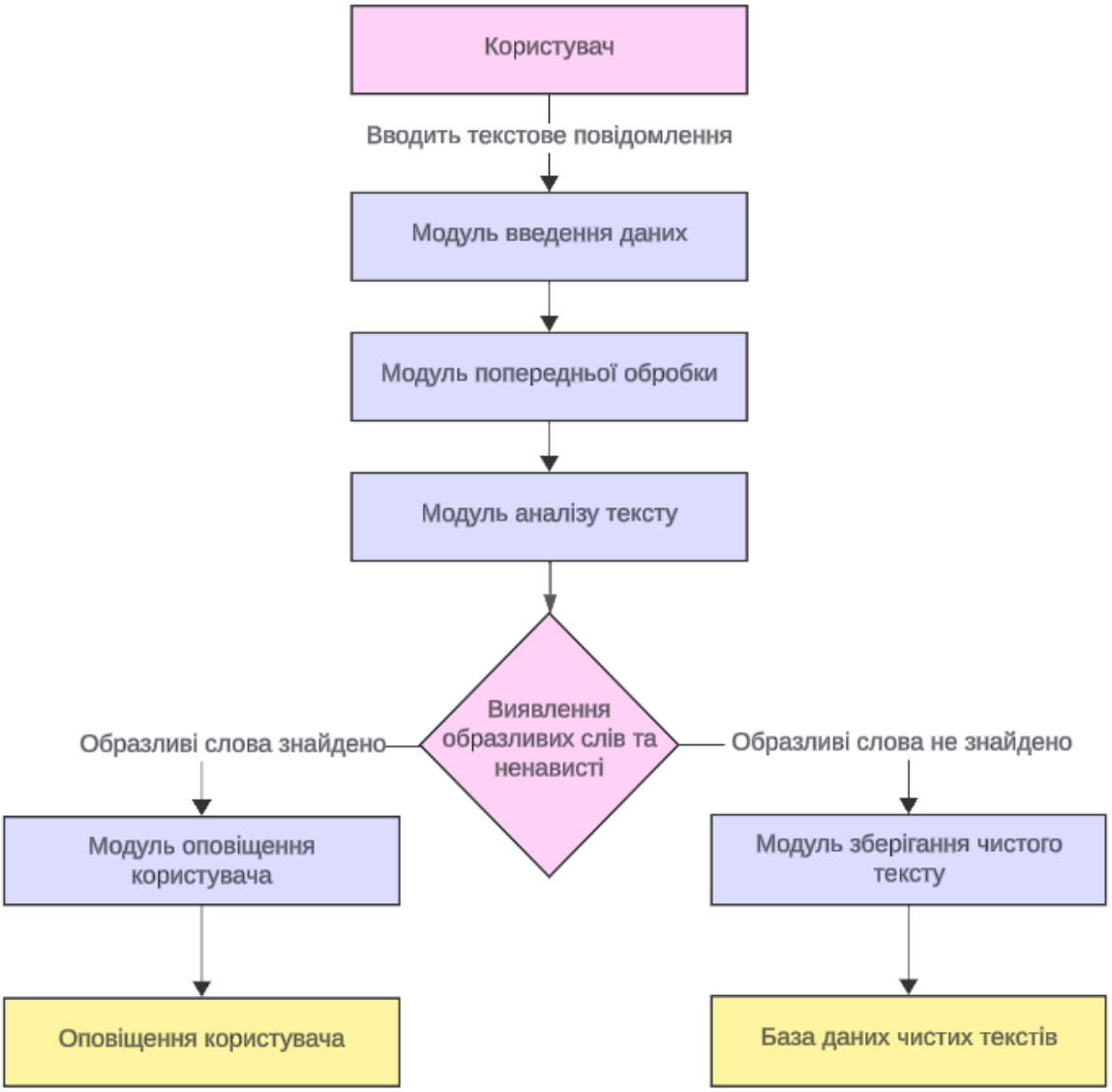

Рисунок 2.1 - Діаграма потоків даних

2.2 Алгоритмічне забезпечення

Для виявлення образливих слів та мови ненависті в текстових повідомленнях використовуються декілька алгоритмічних підходів. Спочатку дані проходять через процес попередньої обробки, включаючи очищення тексту, нормалізацію та токенізацію. Потім застосовується векторизація тексту, перетворюючи його в числовий формат за допомогою методів, таких як TF-IDF або вбудовування слів (Word2Vec, GloVe, BERT).

Після цього будується модель глибинного навчання, яка може використовувати різні архітектури, наприклад, згорткові нейронні мережі (CNN)

або рекурентні нейронні мережі (RNN), а також трансформери. Модель навчається на маркованих даних, де повідомлення позначені як образливі або ні. Під час навчання оптимізується функція втрат для покращення точності класифікації.

Після навчання модель оцінюється на тестових даних для перевірки її ефективності. Для цього використовуються метрики, такі як точність, повнота, F1-міра. На основі результатів моделі проводиться її вдосконалення та налаштування гіперпараметрів.

Після досягнення задовільних результатів модель впроваджується в продуктивне середовище, де вона в режимі реального часу аналізує текстові повідомлення та виявляє мову ненависті. Важливо також забезпечити моніторинг та періодичне оновлення моделі для підтримки її актуальності.

Для створення програмного модуля, можна використовувати сучасні методи глибинного навчання. Нижче наведена блок-схема (Рисунок 2.2), що ілюструє процес створення програмного модуля, що складається з наступних кроків.

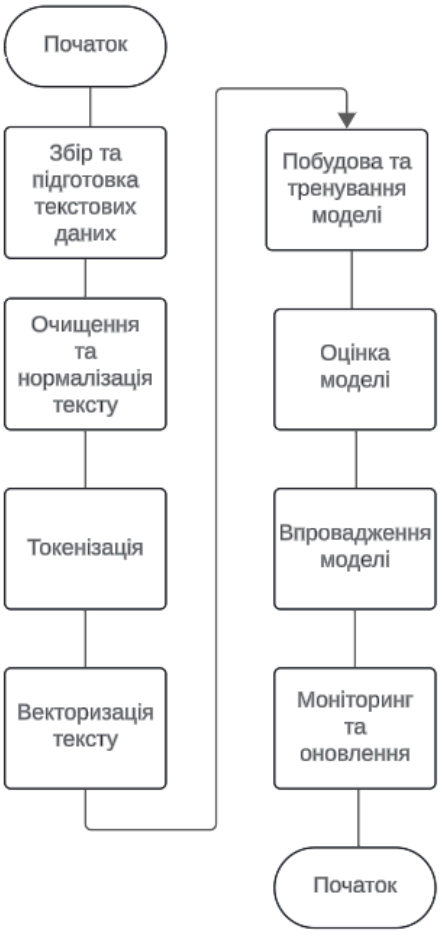

Рисунок 2.2 - Процес створення програмного модуля

Спочатку відбувається збір та підготовка текстових даних. Це включає в себе використання наявних датасетів, таких як ETHOS Hate Speech Detection Dataset, для збирання даних. Далі здійснюється аналіз розподілу даних для визначення частоти образливих та нейтральних повідомлень.

Після збору даних проводиться їх попередня обробка. Спочатку здійснюється очистка тексту, яка включає видалення зайвих символів, стоп-слів та нормалізацію тексту. Наступним кроком є токенізація, тобто розбивка тексту на окремі слова (токени). Після цього проводиться векторизація тексту, що полягає у перетворенні тексту в числовий формат за допомогою методів, таких як TF-IDF або векторизація на основі вбудованих слів (word embeddings).

Далі здійснюється побудова моделі. На цьому етапі здійснюється вибір моделі глибинного навчання, наприклад, LSTM, BERT або їх комбінація. Потім проводиться тренування моделі на підготовленому датасеті з використанням оптимізаторів, таких як Adam, і функцій втрат, таких як Binary Crossentropy. Після цього здійснюється оцінка моделі за допомогою метрик, таких як точність, precision, recall та F1-score для оцінки продуктивності моделі.

Заключним етапом є інтеграція та використання моделі. Це включає впровадження моделі у веб-додаток або API за допомогою фреймворків, таких як Flask. Крім того, створюється зручний інтерфейс для введення тексту та отримання результатів аналізу.

Для детального опису процесу була розроблена діаграма станів (STD) (Рисунок 2.3), що відображає процес створення програмного модуля для виявлення образливих слів та мови ненависті в текстових повідомленнях.

Процес починається зі збору даних, після чого слідує очищення тексту, токенізація, векторизація, тренування моделі, оцінка моделі, впровадження моделі та моніторинг з оновленням.

Кожен етап логічно випливає з попереднього, забезпечуючи послідовний та систематичний підхід до створення ефективної системи для виявлення мови ненависті. Переходи між етапами включають такі стани як "Дані зібрані", "Текст очищений", "Текст токенізований", "Текст векторизований", "Модель

натренована", "Модель оцінена", "Модель впроваджена", і завершуються станом "Оновлення виконане".

Рисунок 2.3 - Діаграма станів

2.3 Інформаційне забезпечення

Опис вхідної та вихідної інформації, яка обробляється в межах функцій предметної області, що автоматизується

Вхідна інформація:

Вхідною інформацією для програмного модуля є текстові дані, такі як повідомлення, коментарі та дописи з різних соціальних мереж, форумів та інших онлайн-платформ. Ці текстові дані надходять у вигляді звичайного тексту, зазвичай у форматах TXT або CSV. Також користувачі можуть вводити налаштування для обробки тексту, що включають вибір словників образливих слів та налаштування параметрів моделі.

Вихідна інформація:

Вихідною інформацією є результати класифікації тексту. Кожен проаналізований текст отримує позначку, що вказує на наявність або відсутність мови ненависті. Ці позначки можуть бути текстовими (наприклад, "образливий" або "не образливий") або числовими (0 - не образливий, 1 - образливий). Додатково створюються журнали та звіти, які включають інформацію про час обробки, кількість виявлених образливих текстів, загальну статистику та інші важливі дані. Ці журнали зберігаються у вигляді файлів (PDF, CSV тощо) для подальшого аналізу та перегляду.

Концептуальне інфологічне проєктування включає визначення основних сутностей, атрибутів і зв'язків між ними для створення логічної структури даних, необхідної для функціонування програмного модуля.

Основною сутністю є "Повідомлення", яка містить атрибути: ідентифікатор, текст повідомлення, дата створення, автор. Ці дані є вхідними і підлягають обробці для виявлення образливих слів та ненависті.

Сутність "Образливі слова" містить ідентифікатор, текстове представлення слова або фрази, категорію (наприклад, расова, гендерна, релігійна) та рівень серйозності. Цей словник використовується для порівняння з текстами повідомлень.

Сутність "Результат аналізу" зберігає результати перевірки повідомлень на наявність образливих слів. Вона включає ідентифікатор результату, посилання на

відповідне повідомлення, статус перевірки (виявлено/не виявлено образливі слова), список виявлених образливих слів, а також дату та час аналізу.

Сутність "Користувач" містить інформацію про користувачів системи: ідентифікатор, ім'я, електронну адресу та роль (наприклад, адміністратор, звичайний користувач). Користувачі можуть вводити повідомлення та переглядати результати аналізу.

Зв'язки між сутностями визначають їхню взаємодію. Кожне повідомлення може мати декілька результатів аналізу, пов'язаних з ним. Кожен результат аналізу може бути пов'язаний з декількома образливими словами. Користувачі створюють повідомлення та переглядають результати аналізу.

Ця модель забезпечує логічну структуру для збору, зберігання та обробки даних, необхідних для ефективної роботи програмного модуля, що виявляє образливі слова та ненависть в текстових повідомленнях.

Глобальна інфологічна модель даних визначає основні сутності та їхні зв'язки, необхідні для роботи програмного модуля.

Основною сутністю є "Текстове повідомлення", яке містить атрибути, такі як ідентифікатор повідомлення, вміст тексту, дата і час створення, автор повідомлення. Ці дані є вхідними і потребують обробки для виявлення образливих слів.

Сутність "Образливі слова" представляє словник слів і фраз, які вважаються образливими або такими, що поширюють ненависть. Кожне слово або фраза має свій унікальний ідентифікатор, текстову форму, категорію образи (наприклад, расова, гендерна, релігійна) та рівень серйозності.

Сутність "Результат аналізу" зберігає результати перевірки текстових повідомлень на наявність образливих слів. Вона включає ідентифікатор результату, посилання на відповідне текстове повідомлення, статус перевірки (виявлено/не виявлено образливі слова), список виявлених образливих слів, а також метадані, такі як дата і час проведення аналізу.

Сутність "Користувач" містить інформацію про користувачів системи, які вводять повідомлення або отримують результати аналізу. Вона включає

ідентифікатор користувача, ім'я, електронну адресу та роль користувача (наприклад, адмін, звичайний користувач).

Зв'язки між сутностями визначають, як ці сутності взаємодіють. Наприклад, кожне "Текстове повідомлення" може бути пов'язане з декількома "Результатами аналізу", а кожен "Результат аналізу" пов'язаний з одним або більше "Образливими словами". "Користувач" може створювати "Текстові повідомлення" та переглядати "Результати аналізу".

Таким чином, глобальна інфологічна модель даних забезпечує логічну структуру для збору, зберігання та обробки інформації, необхідної для ефективного функціонування програмного модуля з виявлення образливих слів та ненависті в текстових повідомленнях.

Проєктування глобальної даталогічної моделі даних.

Для зображення процесу роботи програмного модуля було створено . Entity-Relationship Diagram (ERD) (Рисунок 2.4) - діаграми «суть-зв’язок»

Процес починається з блоку "Налаштування", який включає два основних компоненти: "Текст" і "Параметри". Блок "Текст" представляє текстові повідомлення, які необхідно перевірити, а блок "Параметри" містить додаткові налаштування, такі як значення параметрів B і C.

Текст із блоку "Налаштування" надходить до "Словника образливих слів", де відбувається первинний аналіз на наявність образливих слів. Словник містить перелік слів, які класифікуються як образливі або такі, що поширюють ненависть.

Кожне слово з тексту перевіряється словником та передається до блоку "Класифікатор", який виконує класифікацію слова на основі його приналежності до образливих чи ненависних категорій. Класифікатор використовує певні алгоритми і методи машинного навчання для точного визначення категорії кожного слова.

Результати класифікації передаються до блоку "Журнал", де зберігаються всі дані про перевірені слова і результати їх класифікації. Це дозволяє створити історію перевірок та зберегти всі результати для подальшого аналізу або звітності.

Таким чином, діаграма демонструє послідовність роботи програмного модуля: від введення тексту та параметрів, через аналіз і класифікацію слів, до збереження результатів у журналі.

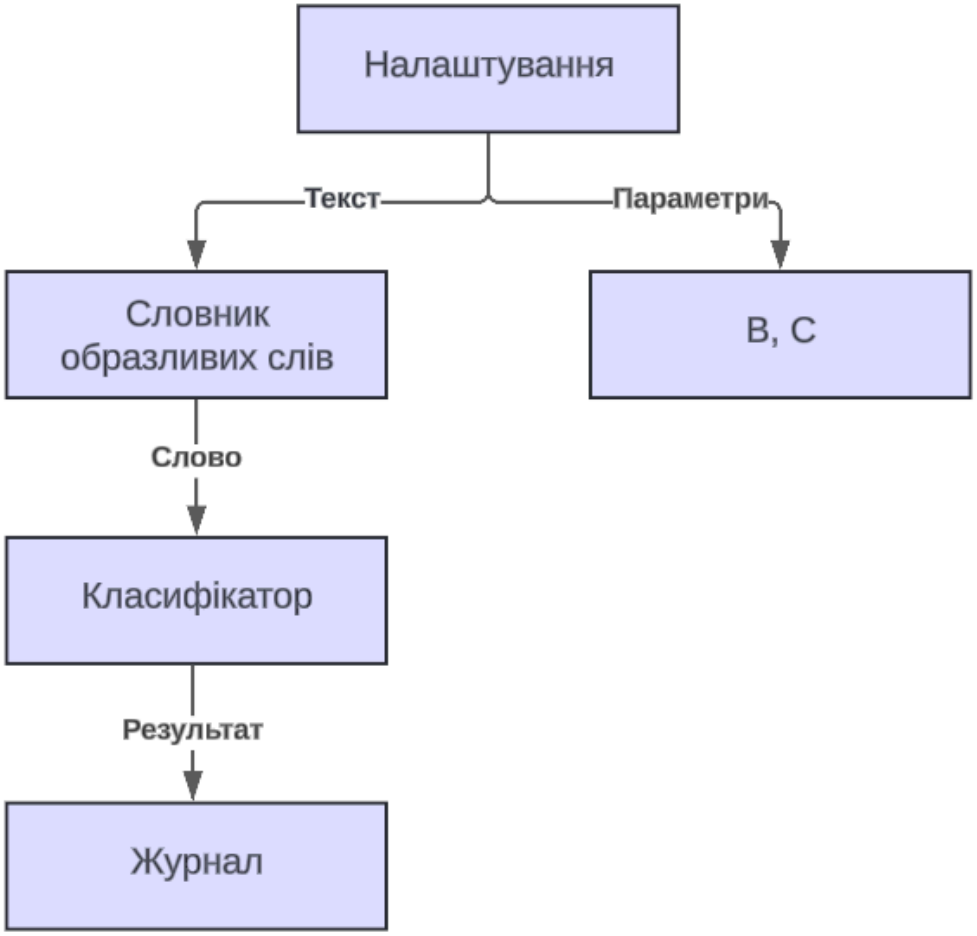

Рисунок 2.4 - Entity-Relationship Diagram

Проєктування фізичної моделі даних для програмного модуля є важливим етапом у створенні ефективної системи для боротьби з ненавистю в Інтернеті. Модель має бути оптимізована для швидкої обробки великих обсягів текстових даних і забезпечення точного виявлення ненависницького контенту.

Першим кроком є визначення структури бази даних, яка буде використовуватися для зберігання текстових повідомлень та результатів аналізу. Основні таблиці включають таблицю користувачів, таблицю текстових повідомлень і таблицю результатів аналізу. Таблиця користувачів містить інформацію про користувачів, включаючи їхні унікальні ідентифікатори, імена та

контактну інформацію. Таблиця текстових повідомлень містить ідентифікатор повідомлення, текст самого повідомлення, часовий штамп і посилання на користувача, який надіслав повідомлення. Таблиця результатів аналізу містить результати аналізу повідомлень, включаючи ідентифікатор повідомлення, результати класифікації (образливе чи не образливе), тип ненависті (якщо застосовно), рівень впевненості моделі та часовий штамп аналізу.

Для ефективної обробки текстових даних необхідно використовувати методи попередньої обробки тексту, такі як токенізація, нормалізація, видалення стоп-слів і стемінг. Це допомагає зменшити розмір тексту і покращити продуктивність моделей глибокого навчання. Після попередньої обробки текстів можна використовувати різні архітектури нейронних мереж для виявлення образливих слів і ненависті. Одним із підходів є використання рекурентних нейронних мереж (RNN) або трансформерів, таких як BERT або GPT, для класифікації тексту.

Фізична модель даних також повинна враховувати питання масштабованості та продуктивності. Для цього можна використовувати індексацію на ключових полях таблиць, таких як ідентифікатори користувачів і повідомлень, щоб забезпечити швидкий доступ до даних. Крім того, розподілена архітектура бази даних, яка підтримує горизонтальне масштабування, може бути корисною для обробки великих обсягів текстових даних у реальному часі.

Резервне копіювання та відновлення даних є критично важливими аспектами фізичної моделі даних. Регулярне резервне копіювання бази даних забезпечує збереження важливих даних у випадку збою системи або втрати даних. Для забезпечення безпеки даних варто використовувати шифрування як у стані зберігання, так і під час передачі даних.

Загалом, фізична модель даних для програмного модуля виявлення образливих слів та ненависті у текстових повідомленнях повинна бути добре спроєктована, щоб забезпечити ефективну обробку великих обсягів текстових даних, точне виявлення ненависті, масштабованість та безпеку даних.

Проєктування фізичної моделі даних включає створення структури бази даних, яка дозволяє ефективно зберігати та обробляти дані.

В цьому проєкті базою даних керує система PostgreSQL. Основні таблиці бази даних включають:

Таблиця користувачів (`users`):

- Поля: `user_id` (унікальний ідентифікатор користувача, первинний ключ), `username` (ім'я користувача), `email` (електронна адреса користувача, унікальне поле), `created_at` (час створення користувача).

- Призначення: Зберігання інформації про користувачів, які відправляють текстові повідомлення.

Таблиця повідомлень (`messages`):

- Поля: `message_id` (унікальний ідентифікатор повідомлення, первинний ключ), `user_id` (ідентифікатор користувача, зовнішній ключ, який посилається на таблицю `users`), `content` (текст повідомлення), `created_at` (час створення повідомлення).

- Призначення: Зберігання текстових повідомлень, які будуть проаналізовані на предмет ненависті.

Таблиця результатів аналізу (`analysis_results`):

- Поля: `result_id` (унікальний ідентифікатор результату, первинний ключ), `message_id` (ідентифікатор повідомлення, зовнішній ключ, який посилається на таблицю `messages`), `is_hate_speech` (булеве значення, яке показує, чи містить повідомлення ненависть), `hate_speech_type` (тип ненависті, якщо застосовно), `confidence_level` (рівень впевненості моделі у визначенні), `analyzed_at` (час проведення аналізу).

- Призначення: Зберігання результатів аналізу текстових повідомлень.

Структура бази даних:

- Таблиця `users` містить інформацію про користувачів, що дозволяє зв'язати текстові повідомлення з конкретними користувачами. Це допомагає відстежувати джерела ненависті та потенційно вживати заходів проти порушників.

- Таблиця `messages` містить текстові повідомлення, які потребують аналізу. Вона зв'язана з таблицею `users` через зовнішній ключ `user_id`, що дозволяє зберігати контекст повідомлення.

- Таблиця `analysis_results` зберігає результати аналізу текстових повідомлень. Це дозволяє зберігати інформацію про те, чи було виявлено ненависть у повідомленні, який тип ненависті був визначений, та з якою впевненістю модель прийняла це рішення.

Методологія: Перед початком аналізу текстові повідомлення проходять попередню обробку. Це включає в себе токенізацію (розбиття тексту на слова або фрази), нормалізацію (приведення тексту до стандартного вигляду), видалення стоп-слів (слова, які не несуть значущої інформації, наприклад, "і", "або"), та стемінг (зведення слів до їх основної форми).

Після цього оброблені повідомлення подаються на вхід моделі глибокого навчання. Модель може використовувати різні архітектури, такі як рекурентні нейронні мережі (RNN) або трансформери (BERT, GPT), для аналізу тексту та виявлення ненависті. Результати аналізу зберігаються у таблиці `analysis_results`.

Масштабованість та продуктивність: Для забезпечення швидкого доступу до даних та ефективної обробки великих обсягів текстових повідомлень, база даних повинна підтримувати індексацію ключових полів, таких як ідентифікатори користувачів та повідомлень. Це дозволяє швидко знаходити та обробляти необхідні дані.

Також важливо забезпечити розподілену архітектуру бази даних, яка підтримує горизонтальне масштабування. Це дозволяє системі зростати разом зі збільшенням обсягів даних та кількості користувачів.

Безпека та резервне копіювання: Забезпечення безпеки даних є критично важливим аспектом. Це включає в себе шифрування даних як під час зберігання, так і під час передачі. Регулярне резервне копіювання бази даних забезпечує збереження важливих даних у випадку збою системи або втрати даних.

Ця фізична модель даних для програмного модуля виявлення образливих слів та ненависті в текстових повідомленнях забезпечує ефективну обробку

великих обсягів текстових даних, точне виявлення ненависті, масштабованість та безпеку даних.

3 ПРОГРАМНО-ТЕХНОЛОГІЧНЕ ЗАБЕЗПЕЧЕННЯ МОДУЛЯ

3.1 Програмна реалізація модуля

Програмний модуль виявлення образливих слів та ненависті у текстових повідомленнях був реалізовано за допомогою мови програмування Python з використанням бібліотек для обробки природної мови та глибинного навчання. Основні компоненти програмного модуля включають підготовку даних, розробку та навчання моделі, а також інтеграцію моделі в програмний модуль для обробки текстових повідомлень в реальному часі.

Структурна схема модуля (Рисунок 3.1) представляє клас "СистемаВиявленняМовиНенависті" та процеси, які він виконує. Клас містить методи, що відповідають за різні етапи обробки текстових даних для виявлення ненависті. Методи включають збирання даних, очищення тексту, токенізацію, векторизацію, тренування моделі, оцінку моделі, впровадження моделі, моніторинг та оновлення.

Перший етап - "ЗбірДаних", який виконує метод "збиратиДані()". Після цього слідує "ОчищенняТексту" з методом "очищатиТекст()", який підготовлює текстові дані для подальшої обробки. Третій етап - "Токенізація", де метод "токенізувати()" перетворює текст на послідовність токенів. Далі йде "Векторизація" з методом "векторизувати()", що перетворює токени на числові вектори.

Після підготовки даних виконується "ТренуванняМоделі", де метод "тренуватиМодель()" використовує дані для навчання моделі глибинного навчання. Наступний етап - "ОцінкаМоделі" з методом "оцінюватиМодель()", який перевіряє ефективність моделі на тестових даних. Потім йде "ВпровадженняМоделі", де метод "впроваджувати()" інтегрує модель у виробниче середовище.

Останній етап - "МоніторингТаОновлення", який включає методи "моніторити()" та "оновлювати()" для спостереження за продуктивністю моделі та її вдосконалення при необхідності.

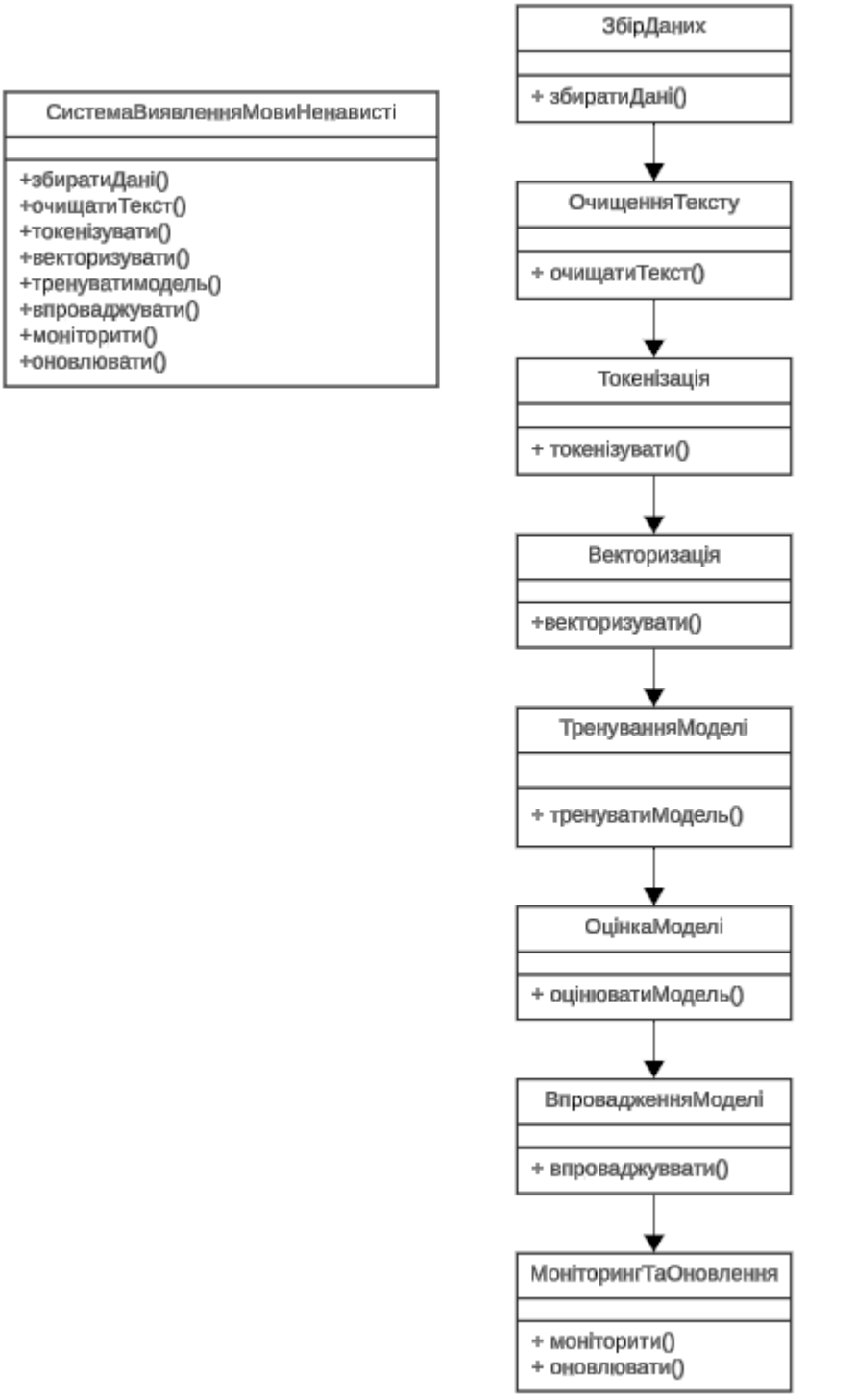

Рисунок 3.1 - Структурна схема програмного модуля

UML-діаграма класів (Рисунок 3.2) для програмного модуля для виявлення образливих слів та ненависті в текстових повідомленнях.

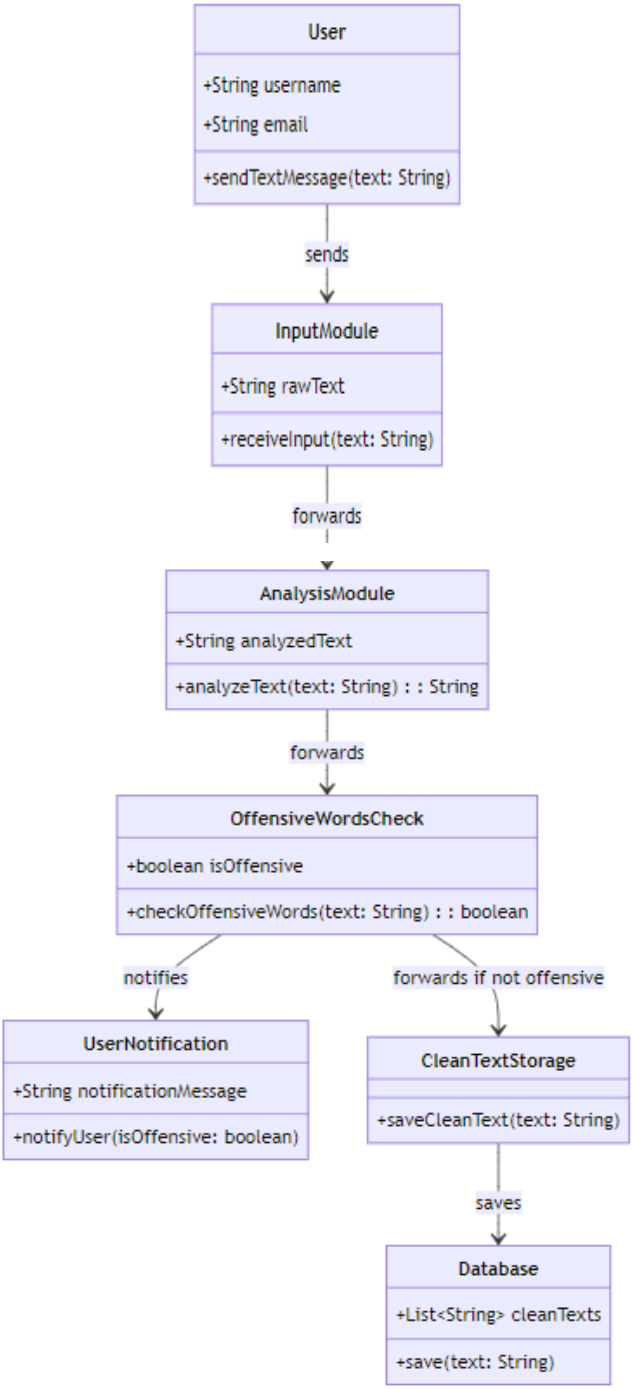

Рисунок 3.2 - UML-діаграма класів

Клас `User` представляє користувача, який надсилає текстові повідомлення для перевірки. Він містить атрибути `username` і `email`, а також метод `sendTextMessage`, який приймає текстове повідомлення.

Клас `InputModule` відповідає за прийом введених текстових повідомлень від користувача. Він має атрибут `rawText` і метод `receiveInput`, який приймає текст як аргумент.

Клас `PreprocessingModule` здійснює попередню обробку тексту, включаючи очищення від зайвих символів та нормалізацію тексту. Він має атрибут `processedText` і метод `preprocessText`, який повертає оброблений текст.

Клас `AnalysisModule` відповідає за семантичний аналіз тексту, використовуючи методи машинного навчання. Він має атрибут `analyzedText` і метод `analyzeText`, який повертає проаналізований текст.

Клас `OffensiveWordsCheck` здійснює перевірку наявності образливих слів у тексті. Він має атрибут `isOffensive` і метод `checkOffensiveWords`, який повертає булеве значення, що вказує на наявність образливих слів.

Клас `UserNotification` відповідає за оповіщення користувача про результати перевірки. Він має атрибут `notificationMessage` і метод `notifyUser`, який приймає булеве значення і оповіщає користувача.

Клас `CleanTextStorage` здійснює зберігання текстів, які не містять образливих слів. Він має метод `saveCleanText`, який зберігає текст.

Клас `Database` представляє базу даних, яка зберігає чисті тексти. Він має атрибут `cleanTexts`, що є списком текстів, та метод `save`, який додає текст до цього списку.

Діаграма станів (Рисунок 3.3) описує процес обробки текстових повідомлень для виявлення образливих слів та ненависті. Процес починається з стану "ВведенняДаних", де користувач вводить текстове повідомлення. Потім дані передаються в стан "ПопередняОбробка", де текст проходить очищення і нормалізацію.

З попередньо обробленими даними модуль переходить до стану "АналізТексту", де проводиться семантичний аналіз тексту. Після аналізу тексту, модуль переходить до стану "ПеревіркаОбразливихСлів", де перевіряється наявність образливих слів.

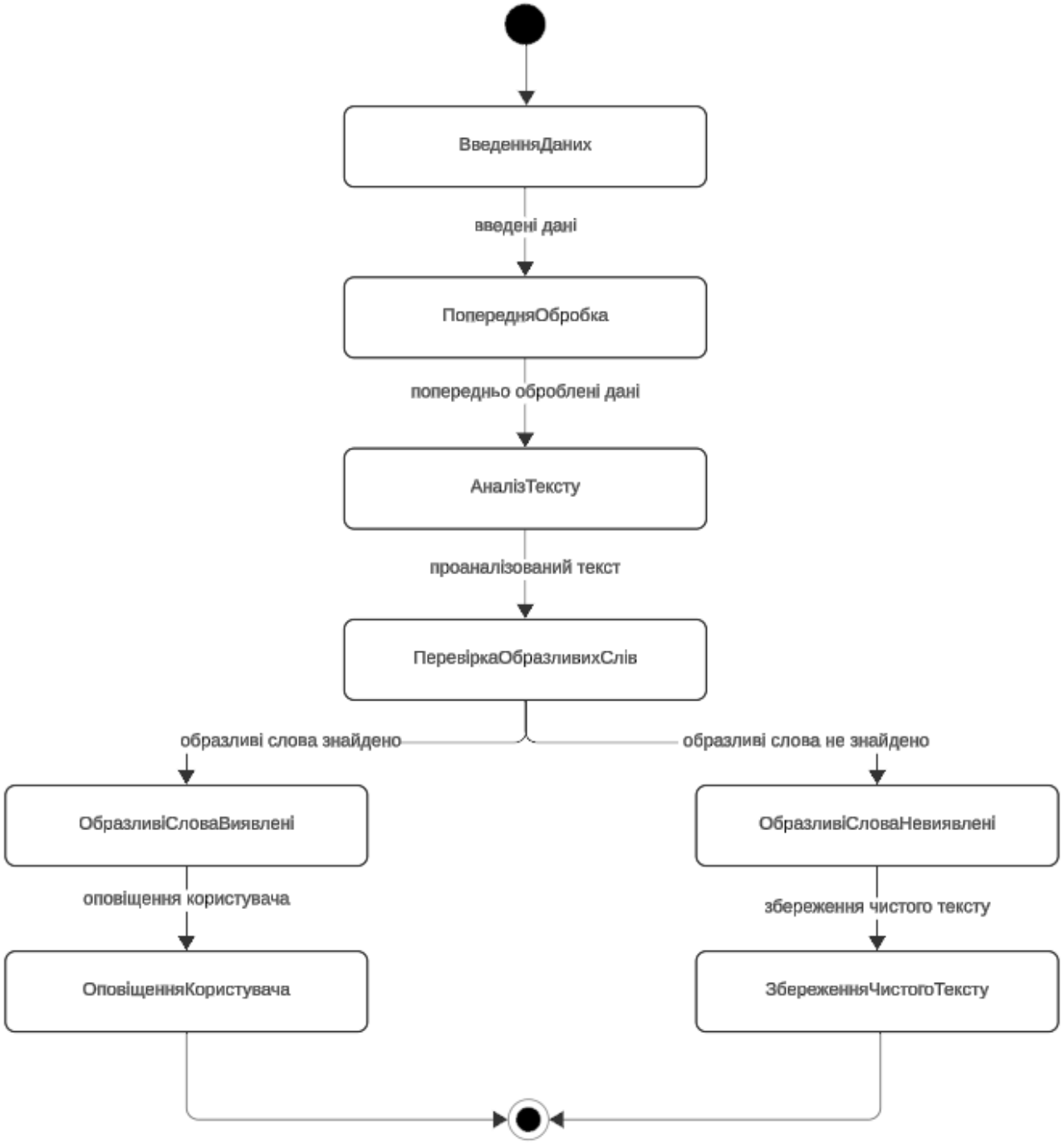

Рисунок 3.3 - UML-діаграма станів

Якщо образливі слова виявлені, модуль переходить до стану "ОбразливіСловаВиявлені", а потім до стану "ОповіщенняКористувача", де користувач оповіщається про результати перевірки. Якщо образливі слова не виявлені, модуль переходить до стану "ОбразливіСловаНеВиявлені", а потім до стану "ЗбереженняЧистогоТексту", де чистий текст зберігається в базі даних. Процес завершується поверненням в кінцевий стан.

Для реалізації програмного модуля вибір програмних засобів має вирішальне значення для досягнення точності, ефективності та зручності використання. Основним інструментом для розробки такого модуля є мова програмування Python завдяки її багатій екосистемі бібліотек для обробки тексту та машинного навчання. Використання бібліотек таких як NLTK і spaCy забезпечує ефективну попередню обробку тексту, включаючи токенізацію, стемінг і лематизацію (Рисунок 3.4).

	class	tweet
0	1	rt junebugg voiceofdstreetz hell yea save mone...
1	2	how the hell was david murphys hit not a home ...
2	1	rt feeelgreatness you dont know where your boy...
3	1	thats some hoe shit doe
4	2	i just want vanilla oreos

Рисунок 3.4 - Набір даних після видалення пунктуації

TensorFlow і PyTorch є вибором для розробки моделей глибокого навчання, які необхідні для складного аналізу тексту та класифікації. Бібліотека scikit-learn пропонує набір алгоритмів машинного навчання для базової обробки і моделювання даних.

Для візуалізації результатів і відображення важливих характеристик даних обираються Matplotlib і Seaborn. Вони дозволяють створювати графіки і діаграми, що допомагають у розумінні розподілу даних та результатів аналізу. Flask або Django служать основою для розробки веб-інтерфейсу, через який користувачі можуть взаємодіяти з модулем. Ці фреймворки підтримують розробку веб-додатків, надаючи інструменти для обробки запитів користувачів, управління базами даних і відображення результатів.

Загалом, обрані програмні засоби забезпечують повний цикл розробки, від обробки текстових даних до побудови та розгортання моделі для виявлення образливих слів та ненависті в текстових повідомленнях. Вони забезпечують

високу продуктивність, точність аналізу і зручність у використанні, що є критичними для успішної реалізації програмного модуля.

3.2 Тренування моделі

На етапі тренування моделі відбувається навчання нейронної мережі на підготовлених даних для виявлення мови ненависті. Спочатку ми створюємо архітектуру моделі, що включає шари для обробки послідовностей тексту за допомогою LSTM (Long Short-Term Memory) нейронів. Для запобігання перенавчанню, додаємо Dropout шари.

Після визначення архітектури моделі, її компіляція здійснюється з використанням функції втрат "binary_crossentropy" і оптимізатора "adam". Модель тренується на тренувальних даних протягом кількох епох, при цьому використовується пакетний розмір для оптимізації процесу навчання.

Під час тренування моделі здійснюється також валідація на тестових даних для оцінки її продуктивності та коригування параметрів. В кінці цього процесу, модель здатна класифікувати текстові повідомлення на основі набутих знань з високою точністю (Рисунок 3.5).

```
Epoch 1/50
257/257 [==============================] - 13s 17ms/step - loss: 1.9472 - accuracy: 0.7541 - val_loss: 1.0245 - val_accuracy: 0.5285
Epoch 2/50
257/257 [==============================] - 3s 13ms/step - loss: 0.3463 - accuracy: 0.9128 - val_loss: 0.5729 - val_accuracy: 0.8445
Epoch 3/50
257/257 [==============================] - 3s 13ms/step - loss: 0.2469 - accuracy: 0.9409 - val_loss: 0.4134 - val_accuracy: 0.8820
Epoch 4/50
257/257 [==============================] - 3s 13ms/step - loss: 0.1812 - accuracy: 0.9627 - val_loss: 0.4633 - val_accuracy: 0.8903
Epoch 5/50
257/257 [==============================] - 3s 13ms/step - loss: 0.1460 - accuracy: 0.9726 - val_loss: 0.4398 - val_accuracy: 0.9015
Epoch 6/50
257/257 [==============================] - 3s 13ms/step - loss: 0.0989 - accuracy: 0.9828 - val_loss: 0.4565 - val_accuracy: 0.8991
Epoch 7/50
257/257 [==============================] - 3s 13ms/step - loss: 0.0802 - accuracy: 0.9867 - val_loss: 0.4333 - val_accuracy: 0.8957
Epoch 8/50
257/257 [==============================] - 3s 13ms/step - loss: 0.0651 - accuracy: 0.9900 - val_loss: 0.4442 - val_accuracy: 0.9010
```

Рисунок 3.5 - Результати тренування моделі

Після цього тексти перетворюються у числові послідовності за допомогою токенізації. Використовуючи Keras Tokenizer, ми обмежуємо кількість слів у послідовностях до 20000 і перетворюємо їх на матрицю вхідних даних. Потім ми використовуємо функцію pad_sequences для вирівнювання всіх послідовностей до однакової довжини.

Модель, яка використовується для виявлення образливих слів та ненависті, включає наступні шари: шар вбудовування (Embedding) для представлення слів, шар просторового випадіння (SpatialDropout1D) для запобігання перенавчанню, рекурентний шар LSTM для обробки послідовних даних і щільний (Dense) шар з функцією активації softmax для класифікації. Модель компілюється з використанням функції втрат categorical_crossentropy і оптимізатора adam.

Для тренування моделі дані розподіляються на навчальну та тестову вибірки. Навчання моделі здійснюється за допомогою методу фітингу з вказанням кількості епох та розміру пакету даних. Після завершення тренування, модель перевіряється на тестовій вибірці, і результати оцінюються за допомогою матриці неточностей та показника точності.

3.3 Тестування роботи модуля

Етап тестування розробленої програми спрямований на оцінку її продуктивності та точності у виявленні мови ненависті в текстових повідомленнях. Після завершення тренування моделі, її роботу перевіряють на тестовому наборі даних, який не використовувався під час навчання. Це дозволяє об'єктивно оцінити здатність моделі узагальнювати та правильно класифікувати нові, невідомі їй раніше тексти.

Важливою частиною тестування є функція прогнозування, яка застосовує модель до нових текстових повідомлень і визначає, чи містять вони образливі висловлювання. Наприклад, після навчання моделі, її тестують на конкретних текстах, щоб перевірити коректність прогнозів. Відповідно, результат прогнозування порівнюється з реальними мітками тестового набору.

Тестування розробленої програми для виявлення образливих слів та ненависті в текстових повідомленнях складається з кількох етапів. Першим етапом є функціональне тестування, яке перевіряє правильність роботи алгоритмів класифікації. Для цього використовуються попередньо зібрані набори даних, які містять тексти з різним рівнем образливості. Програма аналізує ці тексти і порівнює результати з очікуваними значеннями.

Другим етапом є тестування на реальних даних, яке включає аналіз текстових повідомлень з соціальних мереж або інших публічних джерел. Це дозволяє перевірити, як програма справляється з виявленням ненависті в умовах, наближених до реальних.

На рисунку 3.6 показано два графіки, які ілюструють процес навчання моделі для розпізнавання мови ненависті за допомогою глибокого навчання. Перший графік відображає зміну значень втрати та валідаційної втрати протягом епох, а другий графік показує зміну точності та валідаційної точності за ті ж епохи.

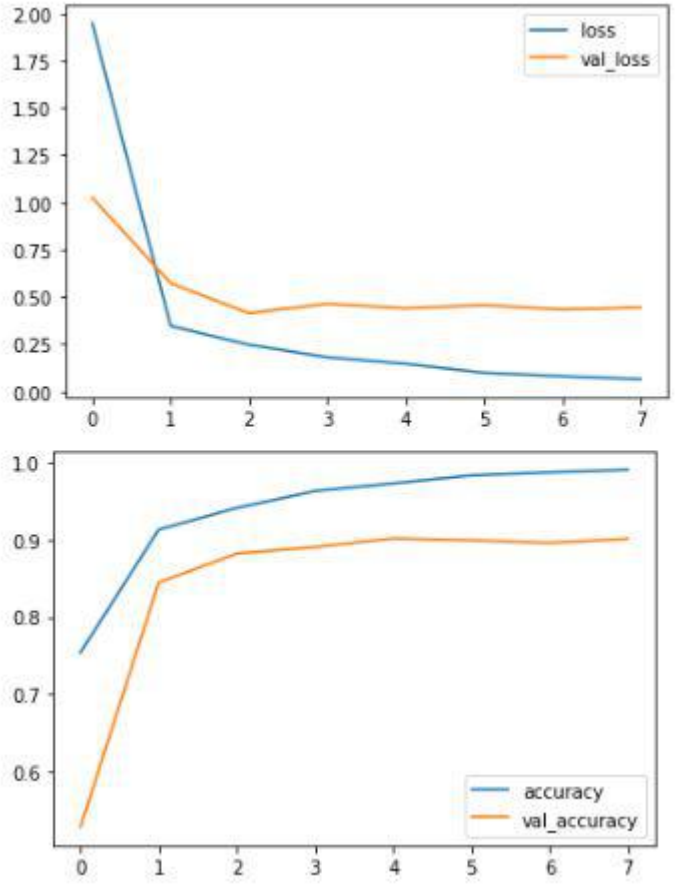

Рисунок 3.6 – Графік втрат і точності

Перший графік демонструє втрату на тренувальних даних (синя лінія) та втрату на валідаційних даних (помаранчева лінія). Значення втрати на тренувальних даних швидко зменшуються протягом перших кількох епох, а потім стабілізуються, що свідчить про те, що модель добре навчається на тренувальних

даних. Втрата на валідаційних даних також зменшується на початку, але потім залишається на приблизно постійному рівні, що може вказувати на певну стабільність або ж початок перенавчання моделі.

Другий графік показує точність на тренувальних даних (синя лінія) та точність на валідаційних даних (помаранчева лінія). Точність на тренувальних даних зростає протягом перших кількох епох і поступово виходить на плато, що свідчить про покращення моделі в розпізнаванні мови ненависті. Точність на валідаційних даних також зростає, хоча й трохи нижчими темпами, ніж тренувальна точність, що може вказувати на узгодженість між тренувальними та валідаційними даними.

Оцінивши графіки можна зробити висновки, що модель показує хороше навчання на тренувальних даних зі зменшенням втрати та збільшенням точності. Валідаційні показники також покращуються, але з менш вираженим трендом, що може сигналізувати про можливість перенавчання, яке слід контролювати. Загалом, ці графіки демонструють, що модель ефективно навчається розпізнавати мову ненависті, хоча подальше вдосконалення може знадобитися для покращення її узагальнюючої здатності.

Наступним етапом є тестування на продуктивність, яке оцінює швидкість та ефективність роботи програми при обробці великих обсягів даних. Це важливо для забезпечення швидкої та надійної роботи у випадках високого навантаження.

Заключним етапом є тестування на стійкість до помилок, яке включає перевірку програми на здатність правильно обробляти некоректні або пошкоджені дані, а також оцінку її надійності в умовах можливих збоїв. Це тестування забезпечує стабільність роботи програми та мінімізує ризики помилок у реальних умовах.

Тестування є критичним етапом, що гарантує високу якість та надійність розробленої програми для виявлення образливих слів та ненависті у текстових повідомленнях.

Модуль розпізнавання мови ненависті за допомогою глибокого навчання готовий до використання. Однак, оскільки він не містить власного модуля виводу,

його інтеграція та результати роботи залежатимуть від конкретної системи, в яку його буде вбудовано. Це означає, що розробники можуть інтегрувати цей модуль у різні додатки або платформи, використовуючи відповідні інтерфейси або бібліотеки, що забезпечать коректне відображення та використання результатів.

ВИСНОВКИ

Розробка програмного модуля для виявлення образливих слів та ненависті в текстових повідомленнях є актуальною та важливою задачею в сучасному світі. Вона спрямована на покращення якості онлайн-комунікацій та створення безпечного середовища для всіх користувачів.

Програмний модуль для виявлення образливих слів та ненависті в текстових повідомленнях демонструє значний потенціал у покращенні онлайн-комунікації та захисті користувачів від шкідливого контенту.

Метою роботи було створення ефективного інструменту для автоматичного аналізу текстових даних з метою ідентифікації образливого контенту та висловлювань, що розпалюють ненависть. Зокрема, дослідження було зосереджено на застосуванні сучасних методів обробки природної мови та машинного навчання для створення точного і надійного інструменту, який може автоматично аналізувати великі обсяги текстових даних.

Використання глибокого навчання, зокрема нейронних мереж, дозволило ефективно виявляти ненависть та образливість у текстах, не обмежуючись заздалегідь складеними списками слів.

Використання реальних даних для навчання моделей допомогло краще адаптуватися до різних варіантів мови та контекстів, забезпечуючи більшу точність виявлення.

В результаті даної роботи було створено програмний модуль, який здатний з високою точністю виявляти образливі слова та висловлювання ненависті в текстових повідомленнях з високою точністю, що свідчить про ефективність використаних алгоритмів і технологій обробки природної мови

Такий модуль може бути використаний для автоматизованої модерації контенту в соціальних мережах, на форумах, у системах електронної пошти та інших платформах, що сприятиме зменшенню кількості випадків кібербулінгу та поширення ненависті онлайн.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.
2. Stallings, W. (2018). Computer Security: Principles and Practice. Pearson.
3. Fortuna, P., & Nunes, S. (2018). A survey on automatic detection of hate speech in text. ACM Computing Surveys (CSUR), 51(4), 1-30.
4. Wulczyn, E., Thain, N., & Dixon, L. (2017). Ex Machina: Personal Attacks Seen at Scale. In Proceedings of the 26th International Conference on World Wide Web (pp. 1391-1399).
5. Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching Word Vectors with Subword Information. Transactions of the Association for Computational Linguistics, 5, 135-146.
6. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Vol. 1, pp. 4171-4186).
7. O'Docherty, M. (2005). Object-Oriented Analysis and Design: Understanding System Development with UML 2.0. John Wiley & Sons.
8. Jurafsky, D., & Martin, J. H. (2021). Speech and Language Processing. Prentice Hall.
9. Elmasri, R., & Navathe, S. B. (2016). Fundamentals of Database Systems. Pearson.
10. Batini, C., Ceri, S., & Navathe, S. B. (1992). Conceptual Database Design: An Entity-Relationship Approach. Benjamin-Cummings Publishing.
11. Chen, P. P. (1976). The Entity-Relationship Model—Toward a Unified View of Data. ACM Transactions on Database Systems (TODS), 1(1), 9-36.
12. Hoffer, J. A., Ramesh, V., & Topi, H. (2016). Modern Database Management. Pearson.
13. Stonebraker, M., & Hellerstein, J. M. (2005). Readings in Database Systems. MIT Press.

14. Bird, S., Klein, E., & Loper, E. (2009). Natural Language Processing with Python. O'Reilly Media.

15. Goldberg, Y. (2017). Neural Network Methods for Natural Language Processing. Morgan & Claypool.

16. Viega, J., & McGraw, G. (2001). Building Secure Software: How to Avoid Security Problems the Right Way. Addison-Wesley.

17. Petkovic, M., & Jonker, W. (Eds.). (2007). Security, Privacy, and Trust in Modern Data Management. Springer.

18. Комар М.П., Саченко А.О., Васильків Н.М., Гладій Г.М., Коваль В.С., Ліп'яніна-Гончаренко Х.В. Методичні рекомендації до виконання кваліфікаційної роботи з освітньо-професійної програми «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки» за першим (бакалаврським) рівнем вищої освіти. Тернопіль: ЗУНУ, 2024. 52 с.

19. Завойовська О. М. Програмний модуль для виявлення образливих слів та ненависті в текстових повідомленнях. Перспективні напрямки розвитку економіки, обліку, управління та права: теорія і практика: тези доп. Міжнарод. науково-практичної конф. (Ізмаїл, 18 травня 2024 р.). Ізмаїл: ЦФЕНД, 2024. С. 28-29. [Додаток А]

ДОДАТОК А

Копія публікації

1. Distance Learning Advantages and Disadvantages: Teaching Experience Analysis at the University with the Basis on Different Informational-Communicative Technologies. / Lobanova, Y.I., Silhavy, R. (eds) // Artificial Intelligence in Intelligent Systems., CSOC 2021. Lecture Notes in Networks and Systems, vol 229. Springer, Cham. https://doi.org/10.1007/978-3-030-77445-5_46

2. Aiman A Turani; Jawad H Alkhateeb; AbdulRahman A. Alsewari.: "Students Online Exam Proctoring: A Case Study Using 360 Degree Security Cameras", 2020 Emerging Technology in Computing, Communication and Electronics (ETCCE), 2020.

3. Hadian S. G. Asep,Yoanes Bandung.:"A Design of Continuous User Verification for Online Exam Proctoring on MLearning",In.: 2019 International Conference on Electrical Engineering and Informatics (ICEEI), 9-10 July 2019.

УДК 004.67

Завойовська О. М.
студентка 4 курсу факультету комп'ютерних
інформаційних технологій
Західноукраїнського національного університету

ПРОГРАМНИЙ МОДУЛЬ ДЛЯ ВИЯВЛЕННЯ ОБРАЗЛИВИХ СЛІВ ТА НЕНАВИСТІ В ТЕКСТОВИХ ПОВІДОМЛЕННЯХ

У сучасному інформаційному середовищі, що характеризується стрімким зростанням обсягів текстової інформації, питання виявлення та запобігання образливих слів та проявів ненависті в текстових повідомленнях стає все більш актуальним. Соціальні мережі, форуми, коментарі на сайтах новин, електронна пошта та інші платформи для обміну повідомленнями часто стають середовищем для поширення ненависницьких висловлювань, які можуть призвести до серйозних соціальних наслідків. Розробка програмного модуля для автоматичного виявлення таких проявів є важливим кроком у боротьбі з онлайн-насильством та кібербулінгом. [1]

Проблема мови ненависті та образ у мережі є глобальною і торкається всіх аспектів суспільного життя. Соціальні платформи, форуми та чати часто стають майданчиками для розповсюдження ненависті, що може призвести до дискримінації, цькування та інших негативних явищ. Тому розробка ефективних інструментів для виявлення та блокування таких повідомлень є критично важливою. [2]

Метою даної роботи є розробка та впровадження програмного модуля, який буде здатний ефективно ідентифікувати образливі слова та висловлювання ненависті у текстових повідомленнях. Зокрема, дослідження зосереджується на застосуванні сучасних методів обробки природної мови (NLP) та машинного навчання для створення точного і надійного інструменту, який зможе автоматично аналізувати великі обсяги текстових даних.

Для досягнення поставленої мети дослідження будуть використані наступні методи: аналіз літератури, розробка алгоритмів NLP, машинне навчання, тестування та валідація.

В результаті даної роботи було створено програмний модуль, який буде здатний з високою точністю виявляти образливі слова та висловлювання ненависті в текстових повідомленнях. Такий модуль може бути використаний для автоматизованої модерації контенту в соціальних мережах, на форумах, у системах електронної пошти та інших платформах, що сприятиме зменшенню кількості випадків кібербулінгу та поширення ненависті онлайн.

Розробка програмного модуля для виявлення образливих слів та ненависті в текстових повідомленнях є важливим кроком у забезпеченні безпеки та комфортного середовища для користувачів інтернету. Застосування сучасних технологій машинного навчання та обробки природної мови дозволяє створити ефективні інструменти, які можуть значно зменшити негативний вплив онлайн-насильства.

Список літератури

1. Waseem, Z., & Hovy, D. (2016). Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL Student Research Workshop* с. 88

2. Jurafsky, D., & Martin, J. H. (2019). *Speech and Language Processing*. Pearson. с. 76

УДК 004.9

Кучар О. М.
студент групи КН-42,
Західноукраїнський національний університет

ПРОГРАМНИЙ МОДУЛЬ НАВЧАННЯ ГРИ В ПОКЕР

Покер, як інтелектуальна гра, поєднує в собі елементи математики, психології та стратегії. Останні десятиліття він набув величезної популярності не лише як азартна гра, а й як спортивна дисципліна, що вимагає високих аналітичних навичок та глибокого розуміння теоретичних основ [1].

У зв'язку з цим, навчання грі в покер стає актуальною темою для дослідження, адже знання та вміння, здобуті під час навчання, можуть бути застосовані не лише в грі, а й у різних сферах життя. Багато основних покерних стратегій зосереджені на фундаментальних поняттях, таких як позиція та розмір ставки, проте новачки часто розглядають їх ізольовано [2].

Навчити гравця можна через систематичне і послідовне вивчення ключових аспектів гри, таких як правила, стратегії, математичні розрахунки, правильна оцінка карт на руках в гравця та на ігровому столі . Постійна практика і навчання на реальних або програмно симульованих ігрових ситуаціях дозволяють гравцеві закріплювати отримані знання та розвивати власний стиль гри.

ДОДАТОК Б

Код програмного модуля

Імпорт бібліотек і наборів даних:

```
%%capture
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
from sklearn.model_selection import train_test_split

# Text Pre-processing libraries
import nltk
import string
import warnings
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from wordcloud import WordCloud

# Tensorflow imports to build the model.
import tensorflow as tf
from tensorflow import keras
from keras import layers
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

nltk.download('stopwords')
nltk.download('omw-1.4')
nltk.download('wordnet')
warnings.filterwarnings('ignore')
```

Попередня обробка тексту:

```
# Lower case all the words of the tweet before any preprocessing
df['tweet'] = df['tweet'].str.lower()

# Removing punctuations present in the text
punctuations_list = string.punctuation
def remove_punctuations(text):
    temp = str.maketrans('', '', punctuations_list)
    return text.translate(temp)

df['tweet']= df['tweet'].apply(lambda x: remove_punctuations(x))
df.head()
```

Допоміжна функція для видалення

```
def remove_stopwords(text):
     stop_words = stopwords.words('english')

     imp_words = []

     # Storing the important words
     for word in str(text).split():

          if word not in stop_words:

               # Let's Lemmatize the word as well
               # before appending to the imp_words list.

               lemmatizer = WordNetLemmatizer()
               lemmatizer.lemmatize(word)

               imp_words.append(word)

     output = " ".join(imp_words)

     return output


df['tweet'] = df['tweet'].apply(lambda text: remove_stopwords(text))
df.head()
Токенізація:
# training the tokenizer
max_words = 5000
token = Tokenizer(num_words=max_words,
                  lower=True,
                  split=' ')
token.fit_on_texts(train_X)

#Generating token embeddings
Training_seq = token.texts_to_sequences(train_X)
Training_pad = pad_sequences(Training_seq,
                                    maxlen=50,
                                    padding='post',
                                    truncating='post')

Testing_seq = token.texts_to_sequences(test_X)
Testing_pad = pad_sequences(Testing_seq,
                                    maxlen=50,
                                    padding='post',
                                    truncating='post')
```

Розробка та оцінка моделі

```
model = keras.models.Sequential([
     layers.Embedding(max_words, 32, input_length=max_len),
     layers.Bidirectional(layers.LSTM(16)),
     layers.Dense(512, activation='relu', kernel_regularizer='l1'),
     layers.BatchNormalization(),
     layers.Dropout(0.3),
     layers.Dense(3, activation='softmax')
])

model.compile(loss='categorical_crossentropy',
               optimizer='adam',
               metrics=['accuracy'])

model.summary()

Зворотні виклики:
from keras.callbacks import EarlyStopping, ReduceLROnPlateau
es = EarlyStopping(patience=3,
                   monitor = 'val_accuracy',
                   restore_best_weights = True)

lr = ReduceLROnPlateau(patience = 2,
                       monitor = 'val_loss',
                       factor = 0.5,
                       verbose = 0)
```