МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БУДІВНИЦТВА І АРХІТЕКТУРИ

Кваліфікаційна наукова
праця на правах рукопису

СТЕЦИК ОЛЕКСІЙ АНДРІЙОВИЧ

УДК 004.4; 004.6; 004.82

ДИСЕРТАЦІЯ

ІНТЕЛЕТУАЛЬНА ВИСОКОНАВАНТАЖЕНА РОЗПОДІЛЕНА СИСТЕМА ОБРОБКИ ДАНИХ В СОЦІАЛЬНИХ МЕРЕЖАХ

126 – Інформаційні системи і технології
12 – Інформаційні технології

Подається на здобуття наукового ступеня **доктора філософії**

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело
О.А.Стецик

Науковий керівник **Теренчук Світлана Анатоліївна,** кандидат фізико-математичних наук, доцент кафедри інформаційних технологій проектування та прикладної математики Київського національного університету будівництва і архітектури

Київ-2024

АНОТАЦІЯ

Стецик О.А. Інтелектуальна високонавантажена розподілена система обробки даних в соціальних мережах. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 126 «Інформаційні системи та технології». – Київський національний університет будівництва і архітектури. – Київ, 2024.

Наукова новизна одержаних результатів. Наукова новизна дисертаційної роботи полягає у тому, що:

вперше розроблено:

- модель НМКТуСН, що є нейронною мережею для класифікації тексту у стрічці новин на основі модифікації гібриду згорткової нейронної мережі та двосторонньої мережі довгої короткочасної пам’яті, яка ефективно поєднує архітектурні рішення, оптимізовані гіперпараметри та натреновані наперед векторні вбудування слів за допомогою нейромереж з трансформер архітектурою, для того, щоб класифікувати текст ефективно та швидко;

- метод зменшення ефекту «бульбашки» через диверсифікацію стрічки новин, який грунтується на класифікації розробленою нейромережею новин за конкретними дискусійними категоріями, і подальшій кластеризації текстів в цих дискусійних категоріях; запропонований метод диверсифікації стрічки новин, сприяє більш тонкому ознайомленню з різними точками зору, щодо дискусійної категорії;

удосконалено:

- метод виявлення спаму і пропаганди, за рахунок тренування НМКТуСН, в напрямку отримання балансу між точністю і швидкістю класифікації;

- архітектуру системи соціальних мереж в напрямку наближення швидкості класифікацій текстів і генерації диверсифікованої стрічки новин до режиму реального часу, шляхом комбінації масштабованих технологій і натренованої НМКТуСН;

набули подальшого розвитку:

- концепція створення гібридних систем, що заснована на моделях штучного інтелекту та машинного навчання в напрямку обробки даних для отримання чистішого і різноманітнішого середовища в соціальних мережах.

Основний зміст дисертаційної роботи.

Дисертаційну роботу присвячену проблемам які виникають у високонавантажених розподілених системах соціальних мереж, таким як: спам, пропаганда, та ефект «бульбашки», та їх вирішенню у реальному часі та при великій кількості користувачів.

За результатами дослідження розроблено:

- детальну архітектуру високонавантаженої розподіленої системи соціальної мережі, яка в реальному часі маркує пости, як спам чи пропаганда, та вказує конкретну пропагандистську техніку, яка використовується;

- метод диверсифікації стрічки новин, який дозволяє зменшення ефекту «бульбашки».

У першому розділі «Аналіз сучасних інтелектуальних високонавантажених розподілених систем соціальних мереж» досліджено сучасні інтелектуальні системи та проблеми, що у них виникають; виконано аналіз існуючих моделей і методів боротьби із спамом та пропагандою; виконано аналіз причин виникнення ефекту «бульбашки» та моделей і методів боротьби з ним; проаналізовано архітектурні вразливі місця розподілених систем соціальних мереж; досліджено причини виникнення вразливостей, таких як: відмову серверів, раптовий наплив великої кількості користувачів, велику кількість одночасних запитів; проаналізовано методи виявлення та усунення несправностей у розподілених системах; обгрунтувано необхідність створення ефективних моделей і методів розпізнавання спаму і пропаганди та боротьби з ефектом «бульбашки».

У другому розділі «Моделі та методи виявлення спаму, пропаганди і зменшення ефекту «бульбашки» в соціальних мережах» описано переваги та недоліки застосування машинного навчання і нейромереж для класифікації спаму, пропаганди та зменшення ефекту «бульбашки»; запропоновано модель нейронна мережа для класифікації тексту у стрічці новин (НМКТуСН) на основі модифікації

гібриду згорткової нейронної мережі та мережі довгої короткочасної пам’яті, яка ефективно поєднує архітектурні рішення, оптимізовані гіперпараметри і натреновані наперед векторні вбудування слів на основі нейромереж з трансформер архітектурою, для ефективної та швидкої класифікації тексту; удосконалено метод класифікації спаму і пропаганди за допомогою використання моделі НМКТуСН; запропоновано метод боротьби з ефектом «бульбашки» на основі моделі НМКТуСН, та кластеризації текстів.

У третьому розділі “Створення моделі високонавантаженої розподіленої системи соціальної мережі” описано необхідні операції користувачів у соціальних мережах; удосконалено масштабовану модель для соціальної мережі; описано напрямки розширення даної моделі, при збільшенні кількості функціоналу; визначено вразливі місця запропонованої моделі; під час моделювання даної системи особлива увага приділяється швидкодії даної системи, та її пропускній здатності; інтегровано модель високонавантаженої системи соціальної мережі з методами класифікації постів на пропаганду та спам; інтегровано модель з методом диверсифікації стрічки новин для зменшення ефекту «бульбашки» в соціальних мережах; покращено систему в напрямку збільшення швидкодії за рахунок використання мережі доставки повідомлень та кешування, додано моніторинг швидкості запитів та наявність помилок.

У четвертому розділі: «Експериментальні дослідження отриманих моделей та методів**»** розроблено інструментальні засоби для класифікації текстів на спам і пропаганду, та кластеризації текстів; проведено порівняння результатів класифікації текстів на спам і пропаганду за допомогою різних моделей і методів машинного навчання та нейронних мереж на основі оцінок влучності, повноти, точності та f-міри; проведено експерименти для кластеризації текстів по дискусійних темах.

Ключові слова: штучний інтелект, машинне навчання, нейронні мережі, управління знаннями, великі дані, швидкодія, пропаганда, спам.

ABSTRACT

Stetsyk O.A. Intelligent high-load distributed data processing system in social networks.

Dissertation for the degree of Doctor of Philosophy in the speciality 126 "Information Systems and Technologies." - Kyiv National University of Construction and Architecture.

Scientific novelty of the results. The scientific novelty of the dissertation is that:

for the first time developed:

- a model NMfTCiNF, which is a neural network for text classification in a news feed based on a modification of a hybrid of a convolutional neural network and a bidirectional long short-term memory network, which effectively combines architectural solutions, optimised hyperparameters and pre-trained vector word embeddings using neural networks with transformer architecture in order to classify text efficiently and quickly;

- a method for reducing the "bubble" effect through news feed diversification, which is based on the classification of news by the developed neural network into specific discussion categories, and the subsequent clustering of texts in these discussion categories; the proposed method of diversifying the news feed contributes to a more subtle introduction to different points of view regarding a discussion category;

improved:

- the method of detecting spam and propaganda, by training NMfTCiNF, in the direction of obtaining a balance between accuracy and speed of classification;

- the architecture of the social networking system in the direction of bringing the speed of text classifications and generation of a diversified news feed closer to real-time, by combining scalable technologies and trained NMfTCiNF;

were further developed:

- the concept of creating hybrid systems based on artificial intelligence and machine learning models in the direction of data processing to obtain a cleaner and more diverse environment in social networks.

The main content of the thesis.

The dissertation is devoted to the problems that arise in highly loaded distributed social networking systems, such as spam, propaganda, and the bubble effect, and their solution in real time and with a large number of users.

Based on the results of the study, we have developed

- a detailed architecture of a highly loaded distributed social network system that labels posts as spam or propaganda in real time and indicates the specific propaganda technique used;

- a method of diversifying the news feed to reduce the bubble effect.

The first chapter, "Analysis of Modern Intelligent Highly Loaded Distributed Social Networking Systems", examines modern intelligent systems and the problems they face; analyses existing models and methods of combating spam and propaganda; analyses the causes of the 'bubble' effect and models and methods of combating it; analyses the architectural vulnerabilities of distributed social networking systems; investigates the causes of vulnerabilities, such as server failure, sudden influx of a large number of users, large number of simultaneous requests; analyses methods of detecting and eliminating faults in distributed systems; substantiates the need to create effective models and methods for recognising spam and propaganda and combating the "bubble" effect.

The second chapter, "Models and methods for detecting spam, propaganda and reducing the bubble effect in social networks," describes the advantages and disadvantages of using machine learning and neural networks to classify spam, propaganda, and reduce the bubble effect; a model of a neural network for text classification in a news feed (NMfTCiNF) based on a modification of a hybrid of a convolutional neural network and a long short-term memory network, which effectively combines architectural solutions, optimised hyperparameters and pre-trained vector word embeddings based on neural networks with a transformer architecture, for effective and fast text classification; improved the method of classifying spam and propaganda by using the NMfTCiNF model; proposed a method for combating the "bubble" effect based on the NMfTCiNF model and clustering texts.

In the third chapter, "Models and methods for detecting spam, propaganda and reducing the bubble effect in social networks", the necessary operations of users in social

networks are described; the scalable model for the social network is improved; the directions of expansion of this model are described, with an increase in the number of functions; the vulnerabilities of the proposed model are identified; when modelling this system, special attention is paid to the performance of this system and its throughput; the model of a highly loaded social network system is integrated with classification methods.

In the fourth chapter: "Experimental studies for the obtained models and methods" develops tools for classifying texts into spam and propaganda and clustering texts; compares the results of classifying texts into spam and propaganda using different models and methods of machine learning and neural networks based on the accuracy, completeness, precision and f-measure; conducts experiments for clustering texts on controversial topics.

Keywords: artificial intelligence, machine learning, neural networks, knowledge management, big data, performance, propaganda, spam.

СПИСОК ОПУБЛІКОВАНИХ НАУКОВИХ ПРАЦЬ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковано основні результати дисертації:

1. Стецик О.А., Теренчук С.А. (2021) Порівняльний аналіз архітектур нереляційних баз даних. Управління розвитком складних систем. № 47. С. 78 – 82. DOI: 10.32347/2412-9933.2021.47.78-82.

2. Yeremenko, B., Mazurenko, R., Stetsyk, O., Buhrov, A. (2023). Intelligent Management of Traffic Flows in Large Cities. Lecture Notes in Intelligent Transportation and Infrastructure, 2023, Part F1379, pp. 33–42 (Scopus). DOI 10.1007/978-3-031-25863-3_4.

3. Stetsyk, O., Terenchuk, S. (2024). Model development of the system for avoiding echo chambers in social networks. Management of Development of Complex Systems, 57, 77–82. DOI 10.32347/2412-9933.2024.57.77-82.

Наукові праці, які засвідчують апробацію матеріалів дисертації:

1. Stetsyk, O., Pasieka, P., Yeremenko, B. (2023). Designing an Effective System Architecture for Detecting Propaganda and Spam in Social Media News Feed. 2023 IEEE 4th KhPI Week on Advanced Technology October 2 – 6, 2023, Kharkiv, Ukraine (Scopus). DOI 10.1109/KhPIWeek61412.2023.10312932.

2. Mazurenko, R., Stetsyk, O. (2022). Intelligent Management of Traffic Flows in Large Cities. ACeSYRI – International Workshop on Modern Experience for PhD students and Young Researchers November 14-18, 2022, pp. 40-41, book of abstract, Zilina, Slovakia. https://ki.fri.uniza.sk/ACeSYRI2022/Abstracts.pdf

3. Poliakov, M., Stetsyk, O., Yeremenko B. (2024) Modeling of the System for the Electronic Estimation of Adolescents' Special Abilities. – IEEE 4th International Conference on Smart Information Systems and Technologies (SIST), 15 - 17 May, 2024, Astana, Kazakhstan. https://sist.astanait.edu.kz/wp-content/uploads/2024/05/IEEE-SIST-2024-Programme-final-with-links-1.pdf

ЗМІСТ

ВСТУП

Актуальність теми. Дисертаційну роботу присвячену проблемам які виникають у високонавантажених розподілених системах соціальних мереж, таким як: спам, пропаганда, та ефект «бульбашки», та їх вирішенню у реальному часі та при великій кількості користувачів.

За результатами дослідження розроблено:

- детальну архітектуру високонавантаженої розподіленої системи соціальної мережі, яка в реальному часі маркує пости, як спам чи пропаганда, та вказує конкретну пропагандистську техніку, яка використовується;

- метод диверсифікації стрічки новин, який дозволяє зменшувати ефект «бульбашки».

Мета дослідження полягає в розробці моделі швидкодіючої високонавантаженої інтелектуальної системи соціальної мережі, оптимізованої для забезпечення чистоти контенту від спаму і пропаганди та здатної зменшувати ефект "бульбашки", тим самим сприяючи ширшому розповсюдженню і різноманіттю ідей та поглядів.

Для досягнення вказаної мети необхідно вирішити такі завдання:

1. Проаналізувати архітектурні вразливі місця розподілених систем соціальних мереж, які впливають на швидкодію, доступність та масштабованість системи;

2. Дослідити існуючі методи боротьби із спамом та пропагандою та причини виникнення ефекту «бульбашки» та методи боротьби з ним;

3. Створити модель нейронної мережі для ефективної класифікації текстів;

4. Запропонувати метод на основі машинного навчання та нейромереж для виявлення спаму та пропаганди в режимі реального часу;

5. Запропонувати метод на основі машинного навчання та нейромереж для зменшення ефекту «бульбашки»;

6. Створити швидкодіючу масштабовану модель розподіленої системи соціальної мережі, та визначити напрямки вдосконалення системи;

7. Інтегрувати отримані методи в процес функіонування системи соціальної мережі таким чином, щоб система зберегла властивості масштабованості та швидкодії.

Об’єкт дослідження – високонавантажені розподілені системи обробки даних у соціальних мережах.

Предмет дослідження – інтелектуальні моделі, методи і технології вирішення проблем проблем пропаганди, спаму і ефекту «бульбашки» в системах соціальних мереж.

Методи дослідження, що використовувалися під час роботи:

- аналіз існуючої літератури;
- інженерне проектування дизайну системи;
- збір та аналіз даних з реальних систем соціальних мереж;
- програмна розробка, та експерименти з отриманими моделями і методами на різних корпусах даних;

Наукова новизна одержаних результатів. Наукова новизна дисертаційної роботи полягає у тому, що:

вперше розроблено:

- модель НМКТуСН, що є нейронною мережею для класифікації тексту у стрічці новин на основі модифікації гібриду згорткової нейронної мережі та двосторонньої мережі довгої короткочасної пам’яті, яка ефективно поєднує архітектурні рішення, оптимізовані гіперпараметри та натреновані наперед векторні вбудування слів за допомогою нейромереж з трансформер архітектурою, для того, щоб класифікувати текст ефективно та швидко;

- метод зменшення ефекту «бульбашки» через диверсифікацію стрічки новин, який грунтується на класифікації розробленою нейромережею новин за конкретними дискусійними категоріями, і подальшій кластеризації текстів в цих дискусійних категоріях; запропонований метод диверсифікації стрічки новин, сприяє більш тонкому ознайомленню з різними точками зору, щодо дискусійної категорії;

удосконалено:

- метод виявлення спаму і пропаганди, за рахунок тренування НМКТуСН, в напрямку отримання балансу між точністю і швидкістю класифікації;

- архітектуру системи соціальних мереж в напрямку наближення швидкості класифікацій текстів і генерації диверсифікованої стрічки новин до режиму реального часу, шляхом комбінації масштабованих технологій і натренованої НМКТуСН;

набули подальшого розвитку:

- концепція створення гібридних систем, що заснована на моделях штучного інтелекту та машинного навчання в напрямку обробки даних для отримання чистішого і різноманітнішого середовища в соціальних мережах.

Практичне значення отриманих результатів – полягає в можливості негайного впровадження отриманих в дисертаційній роботі, результатів для застосування їх до маркування спаму та пропаганда контенту в соціальних мережах, і визначенні конкретної технології, що використовується в тексті пропагандистами. Також в можливості застосування методу зменшення ефекту «бульбашки» в соціальних мережах та отриманні більш різноманітного контенту в стрічці новин.

Особистий внесок здобувача. Дисертація є самостійною науковою працею, в якій висвітлені власні ідеї та розробки автора, що дозволили досягти поставленої мети. З наукових праць, опублікованих у співавторстві, у дисертації використано лише ті ідеї та положення, які є результатом особистої роботи здобувача.

Апробація результатів дисертації. Основні положення та результати дисертаційної роботи висвітлені та обговоренні на наступних конференціях:

- 13th International Scientific Conference "Transbaltica 2022: Transportation Science and Technology", September 15-16, Vilnius, Lithuania, 2022;
- ACeSYRI – International Workshop on Modern Experience for PhD students and Young Researchers November 14-18, 2022, Zilina, Slovakia;
- 2023 IEEE 4th KhPI Week on Advanced Technology October 2 – 6;
- IEEE 4th International Conference on Smart Information Systems and Technologies (SIST), 15 - 17 May, 2024.

Структура та обсяг дисертації. Дисертаційна робота складається із вступу, чотирьох розділів, загальних висновків, списку використаних джерел та додатків. Загальний обсяг роботи становить 157 сторінок у тому числі, основна частина складає 119 сторінок. Основна частина, крім тексту, включає – 14 таблиць, та 32 рисунки.

Подяка. Висловлюю глибоку подяку науковому керівнику – кандидату фізико-математичних наук, доценту Теренчук Світлані Анатоліївні.

РОЗДІЛ 1. АНАЛІЗ СУЧАСНИХ ІНТЕЛЕКТУАЛЬНИХ ВИСОКОНАВАНТАЖЕНИХ РОЗПОДІЛЕНИХ СИСТЕМ СОЦІАЛЬНИХ МЕРЕЖ

У першому розділі досліджено сучасні інтелектуальні системи та проблеми, що у них виникають; виконано аналіз існуючих моделей і методів боротьби із спамом та пропагандою; виконано аналіз причин виникнення ефекту «бульбашки» та моделей і методів боротьби з ним; проаналізовано архітектурні вразливі місця розподілених систем соціальних мереж; досліджено причини виникнення вразливостей, таких як: відмову серверів, раптовий наплив великої кількості користувачів, велику кількість одночасних запитів; проаналізовано методи виявлення та усунення несправностей у розподілених системах; обгрунтувано необхідність створення ефективних моделей і методів розпізнавання спаму і пропаганди та боротьби з ефектом «бульбашки».

1.1 Інтелектуальні системи соціальних мереж

Перші соціальні мережі, що створені для зв’язку людей на основі стосунків у реальному світі, з’явилися лише 2 десятиліття тому, а саме: Myspace – у 2003 р, Facebook – у 2004 р., Twitter – у 2006р., WhatsApp – у 2009 р. і Telegram – у 2013 р. Сьогодні соціальні мережі стрімко розвиваються і, долаючи географічні кордони, об’єднують людей у всьому світі покращуючи зв’язок і спілкування.

Сучасні соціальні медіа дозволяють охопити такі функції [1]:

– мультимедіа (фото, відео, прямі трансляції);

– прямий канал спілкування (відеодзвінки, групові чати, приватні повідомлення);

– персоналізований алгоритм рекомендацій (показ лише певного контенту для індивіда);

– інтерактивні функції (вікторини, опитування, елементи гейміфікації та інтеграція електронної комерції).

Проте з цим функціоналом з’явилися і такі проблеми, як:

– спам (соцмережі можуть бути використані для поширення шахрайських посилань, фейкових новин) [2]

– пропаганда (соцмережі можуть бути використані для політичних маніпуляцій) [3];

– ефект «бульбашки» (соціальні мережі створюють середовище, в якому користувачі обмежені інформацію, що підтверджує їхні переконання, і це посилює поляризацію серед людей) [4].

– порушення конфіденційності (соціальні медіа збирають велику кількість даних користувачів, які можуть бути використані для таргетування реклами та крадіжки особистих даних) [5];

– переслідування (анонімність і відсутність відповідальності в соціальних мережах можуть призвести до знущань) [6].

– погіршення психічного стану (надмірне використання соціальних медіа може спричинити тривогу, відчуття неадекватності та депресію) [7] .

У цій роботі основна увага зосереджена на проблемі спаму і пропаганди у соціальних мережах та ефекту бульбашки.

Кількість користувачів соціальної мережі X (Twitter) становить 420 мільйонів, це ті користувачі, які заходять в мережу, хоча б один раз в місяць. Приблизно 220 мільйонів з них заходять в мережу, не менше разу в день [8].

Приблизно 6000 твітів публікується в середньому за 1 секунду, тобто приблизно 500 мільйонів твітів на день. Найбільша кількість твітів за одну секунду становить 143199. Кількість запитів користувачів за одну секунду в середньому – 300000 [9].

Кількість користувачів соціальної мережі Facebook становить 3.05 мільярдів, це ті користувачі, які заходять в мережу, хоча б один раз в місяць. Приблизно 2.1 мільярди з них заходять в мережу не менше разу в день. Фейсбук видалив приблизно 1.5 мільярдів фейкових акаунтів, за один річний квартиль у 2022 році [10].

Враховуючи велику кількість користувачів та велику кількість контенту, який вони створюють, архітектура систем високонавантажених соціальних мереж повинна бути високо масштабованою. Додаткових викликів до масштабованості систем додають користувачі «зірки», які мають велику кількість підписників, тому на їхні публікації, реагує велика кількість людей. Також під час популярних подій, як то: вибори, природні катастрофи, вірусні тренди відбувається раптові та непередбачувані сплески трафіку, відбуваються перевантаження системи. Тому, вирішуючи проблему спаму, пропаганди, та ефекту «бульбашки» в соціальних мереж в даній роботі увага приділяється не тільки точності рішення, а й його швидкості та масштабованості.

1.2 Контентні проблеми та їх рішення в соціальних мережах

1.2.1 Пропаганда та спам

Вплив соціальних медіа-платформ виходить за межі їхньої ролі нейтральних каналів комунікації і вони активно беруть участь у формуванні взаємодії з користувачами через алгоритми генерації новин. Ці платформи надають пріоритет контенту, що генерує високий рівень залучення користувачів (лайки, поширення та коментарі). Така розстановка пріоритетів часто надає перевагу контенту, що характеризується сенсаційністю, обуренням або дезінформацією, експлуатуючи людську психологічну схильність до підвищеної емоційної реакції та залученості. При цьому людьми часто поширюється і споживається контент, який не є правдивим, чи не відповідає суспільному добробуту, такий, як спам і пропаганда.

Одним із критичних проявів такої динаміки є поширення автоматизованих акаунтів і "бот-мереж", які імітують людську діяльність для посилення певних повідомлень або наративів. Таке штучне посилення створює викривлене сприйняття популярності або консенсусу навколо певних тем, на які люди зазвичай покладаються при оцінюванні достовірності інформації [11].

Анонімність, яку надає інтернет, усуваючи багато соціальних і етичних обмежень, які в іншому випадку не сприяють поширенню шкідливого контенту, підсилює проблему поширення пропаганди і спаму.

Дослідження Массачусетського технологічного інституту показало, що в Твітері оманливі новини поширюються на 70% частіше, ніж правдиві. При цьому для того, щоб досягти аудиторії в 1500 осіб, правдива інформація потребує в шість разів більше часу, ніж оманливі історії [12]. Таке швидке поширення і широке охоплення можуть суттєво посилити вплив пропаганди, сприяючи її вірусному поширенню і швидкому впливу на величезні аудиторії.

Не можна також ігнорувати економічні та політичні аспекти цього явища. Створення і поширення оманливого чи сенсаційного контенту часто мотивоване фінансовою вигодою, доходами від реклами або такими стратегічними цілями, як політичний вплив чи соціальний розбрат [13]. Комерціалізація уваги та інформації має глибокі наслідки, трансформуючи публічний дискурс і ставлячи під сумнів цілісність демократичних процесів.

Пропаганда здатна роз'єднувати громади, сприяти насильницькому екстремізму і риториці ненависті, а також підривати основи демократії, зменшуючи віру в демократичні практики. Саме тому вона часто використовується для впливу на громадську думку з метою підтримки певної політичної ідеології або точки зору [14].

У статстиці наданій у [15] вказано, що станом на 2017 р. у 28 країнах було використано соціальні мережі для поширення дезінформації про політику, а станому на 2020 рік, таких країн було 81.

Кумулятивним ефектом динаміки використання соціальних мереж для поширення пропаганди та дезінформації є значна ерозія довіри до традиційних джерел інформації, включаючи засоби масової інформації, наукові установи та демократичні інститути. Поширення теорій змови, дезінформація про заходи громадського здоров'я та спроби підірвати виборчі процеси є прикладами відчутних наслідків еволюції цифрової екосистеми.

Важливість протидії поширенню дезінформації та спаму на платформах соціальних мереж за допомогою методів машинного навчання та використання нейронних мереж привернула значну увагу науковців, що призвело до кількох помітних досліджень.

Автори [16] всебічно вивчили виявлення фейкових новин у соціальних мережах, що охоплює:

- характеристики фейкових новин, засновані на психологічних і соціальних теоріях;
- метрики оцінювання;
- набори даних для класифікації.

Автори [17] використали штучний інтелект, а саме різні методи машинного навчання для покращеного виявлення пропаганди. Спочатку було використано статистичний показник TF-IDF і метод мішок слів, для перетворення текстів у невпорядковані вектори дійсних чисел. Далі до текстів було застосовано машини опорних векторів (англ. SVM), та мультиноміальний наївний Баєсівський метод. У дослідженні отримано точність 70 відсотків та влучність 69 відсотків за допогою модифікації методу машин опорних векторів.

У дослідженні [18] для виявлення спаму в соціальних мережах використано різні методи машинного навчання: дерево рішень, випадковий ліс, наївний баєсівський класифікатор, логістична регресія, та нейронну мережу прямого поширення. Найкращий результат отримано за допомогою нейронної мережі прямого поширення.

Автори [19] розробили бінарну класифікацію контенту новин на фейкові новин і справжні та застосували машинни опорних векторів та наївний баєсівський алгоритм.

У той час, як у дослідженнях [17], [18], [19] запропоновано методи для класифікації текстів на спам, пропаганду, та справжні і фейкові новини, їх стрімке поширення у сучасних соціальних мережах, свідчить про необхідність створення методів класифікації деструктивного контенту, який би мав більшу точність і влучність, та міг би бути використаний для систем з великою масштабованістю.

1.2.2 Ефект «бульбашки»

Когнітивні упередження людського мозку відіграють значну роль у цифрової екосистемі. Упередженість підтвердження (схильність шукати і надавати пріоритет інформації, яка підтверджує вже існуючі переконання) посилюється алгоритмами соціальних мереж, що призводить до формування ідеологічно однорідних спільнот - соціальних «бульбашок». Соціальні «бульбашки» сприяють формуванню почуття приналежності та підтвердження через такі механізми соціального зворотного зв'язку, як "лайки" і "поділитися". Для генерації стрічки новин у соціальних мережах Ікс (Твіттер) та Фейсбук використовуються методи на основі глибинних нейронних мереж [20], [21]. Як вхідні дані використовується такі дані користувача, як пости і коментарі, що він поширює, які пости йому подобаються і не подобаються, які публікації він коментує та зберігає.

Метод генерації стрічки новин є однією з причин утворення соціальних «бульбашок», оскільки метод пріоритезує контент подібний, до того, який користувач бачив раніше. Метод генерації стрічки новин допомагає привернути увагу користувачів, що є однією з основних бізнес-цінностей соціальних мереж. Тобто з позиції бізнесу основною метою створення стрічки новин є максимізація часу, який користувач витрачає на поточну соціальну мережу [22].

Порівняння впливу ефекту «бульбашки» в різних соціальних мережах, таких як Facebook, Twitter, Reddit, було зроблено в [23]. Автори цієї статті порівняли 100 мільйонів текстів, які містили дискусії на суперечливі теми, такі як контроль над зброєю, соціальна допомога, аборти, вакцинація. У результаті було виявлено, що ефект «бульбашки» сильніше проявляється в мережі Фейсбук чим Редділт.

Автори [24] припустили, що причина більшого впливу ефекту «бульбашки» на соціальну мережу Фейсбук полягає в тому, що користувачі платформи Reddit мають вищий ступінь контролю над стрічкою новин.

У дослідженні [25] автори виявили, що користувачі Ікс (Твіттер) зазвичай бачать політичну думку, яка підтримує їхню власну. І користувачі, які хочуть поділитися різними думками щодо політичного спектру, сплатять ціну

двопартійності, тобто будуть розкритиковані користувачами мережі, які підтримують одну політичну партію. Крім того це дослідження показало, що ефект «бульбашки» не з’являється, якщо тема не є суперечливою.

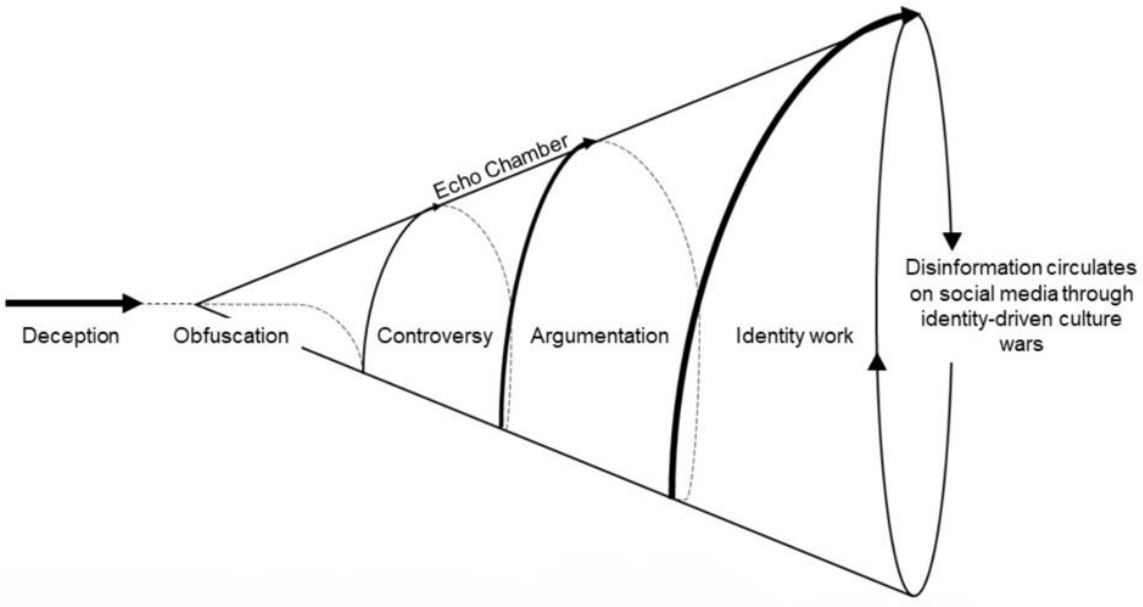


Рисунок 1.1 Механізм поширення новин у соціальних «бульбашках».

Передрук з [26]

У роботі [26] описано два етапи утворення соціальної «бульбашки» через політичні переконання. На першому етапі цього процесу, який називається "посів", пропагандисти стратегічно вводять в оману користувачів, маскуючи свою легітимність, наприклад використовуючи фейкові новини. На другому етапі, уже самі користувачі залучаються до створення суперечливих наративів. Тут включається не лише неправда, але й правда, напівправда та ціннісні судження. Тобто пропагандисти на цьому етапі працюють з ідентичностями людей на основі ціннісних суджень.

Автори дослідження [27], яке зосереджує увагу на поширенні чуток в соціальних «бульбашках», виявили, що перші 10% чуток в «бульбашці» сприяють поширенню 24% чуток у всіх соціальних мережах. Автори цього дослідження також виявили, що 36% ретвітів цих чуток зроблено членами тієї ж соціальної «бульбашки», в якій з’явилася початкова новина.

У роботах [28], [29] досліджено причини появи ехо-бульбашок:

– метод генерації контенту може віддавати пріоритет контенту, який підтверджує існуючі переконання та обмежує вплив різних точок зору, що посилює нетерпимість до протилежних переконань, тобто метод не є диверсифікованим по дискусійних темах;

– кількість з’єднань між кластерами користувачів мала, а кількість з’єднань всередині кластера велика.

– чати можуть досить швидко поширювати дезінформацію;

– приватні групи можуть підтримувати лише одну точку зору для деяких тем;

– відсутність звички перевіряти якість контенту та фактів, особливо коментарів від ботів та діпфейків від ботів;

У статті [30] автори за допомогою штучного інтелекту, а саме методів машиного навчання у соціальних мережах визначають політичну приналежність до республіканців чи демократів. Автори дослідили, що більша гомофілія проявляється серед прихильників демократичної партії. Це свідчить про неоднорідну структуру кластерів соціальних «бульбашок», навіть коли обговорення стосується однієї дискусійної теми.

У статті [31] автори представляють модель, спрямовану на визначення ідеологічних позицій користувачів соціальних мереж та контенту, яким вони діляться. Ця модель, яка досягає високого рівня точності, не покладаючись на прямі ярлики упередженості, є основою для генерування персоналізованих рекомендацій.

Ці рекомендації стають більш ідеологічно різноманітними завдяки інноваційному алгоритму, що базується на випадкових блуканнях [32]. Тести авторів, проведені на великих масивах даних Ікс (Твіттера), показують, що їхній підхід дійсно може розширити спектр ідеологічних політичних поглядів у наданих рекомендаціях.

Враховуючи попередні досліження, для змешення ефекту «бульбашки» актуальною є потреба в створенні ефективного масштабованого методу генерації

диверсифікованого контенту по дискусійних темах, для користувачів з різних соціальних «бульбашок».

1.3 Аналіз архітектурних проблем і обмежень в розподілених системах

1.3.1 Основи розподілених систем

Розподілена система – це мережа незалежних комп’ютерів, які працюють разом для досягнення спільної мети, обмінюючись ресурсами та інформацією через координований зв’язок [33].

Шляхом розподілу відповідальності на різні сервери, система зменшує ризики, асоційовані з єдиною точкою відмови, забезпечуючи вищу масштабованість та стійкість у нестабільних умовах [34].

Основні вимоги до розподілених систем: масштабованість, доступність, узгодженість даних, та малий рівень затримки [35].

Масштабованість є ключовою властивістю, що відображає спроможність системи динамічно обробляти зростаючі навантаження шляхом коригування своїх ресурсів [36]. Існують два основні способи масштабування: Горизонтальне масштабування: розширення мережі шляхом додавання більшої кількості вузлів для розподілу навантаження між більшою кількістю ресурсів. Вертикальне масштабування: покращення можливостей існуючих вузлів за допомогою потужніших апаратних компонентів (наприклад, додаткових процесорів, оперативної пам'яті або зберігання). Масштабованість є важливою для забезпечення продуктивності та надійності системи, коли попит і кількість запитів до неї зростає.

У розподілених системах доступність означає гарантію того, що система залишається працездатною та зможе давати відповіді на запити навіть під високим навантаженням, або коли її окремі компоненти виходять з ладу. Це часто передбачає впровадження «зайвих» компонентів дублікатів, таких як запуск кількох екземплярів критичних служб, та інтеграцію механізмів толерантності до

відмов, на рівнях як програмного, так і апаратного забезпечення [37]. Дизайн системи із високою доступністю мінімізує простої та перебої у наданні послуг.

Підтримання узгодженості даних у розподіленій мережі є складною задачею. Моделі узгодженості варіюються від сильної узгодженості, де кожний запит на вичитку відображає найсвіжіші зміни, до кінцевої узгодженості, яка допускає тимчасові розбіжності між копіями даних, але забезпечує їхнє остаточне збігання. Вибір відповідної моделі узгодженості залежить від конкретних вимог до застосунку та потреб у продуктивності [38]. Для розподілених систем соціальних мереж умови на узгодженість не настільки строгі, як наприклад для застосунків, які обслуговують банківську сферу, чи сферу торгів на біржі.

Часова затримка між тим, як користувача зробив запит до системи та отриманням даних, є критичним фактором у розподілених системах. Мережеве спілкування вносить додаткову затримку, що робить важливим впровадження технік оптимізації для забезпечення отриманні відповіді від систему до користувача [39].

1.3.2 Аналіз архітектурних проблем на основі теореми Брюера

У сфері розподілених обчислень теорема CAP (англ.: consistency, availability, partition tolerance), є наріжним принципом, що формує архітектуру систем. Ця теорема стверджує, що в рамках розподіленої системи керування даними неможливо одночасно забезпечити три основні властивості: узгодженість, доступність і толерантність до розділів.

Узгодженість вимагає, щоб будь-яка операція читання отримувала останній запис або сигналізувала про помилку, таким чином зберігаючи узгоджену перспективу даних на всіх вузлах.

Доступність вимагає, щоб на кожен запит відповідали без помилок, що підкреслює працездатність системи та швидкість реагування.

Толерантність до розділів позначає здатність системи підтримувати правильну роботу, незважаючи на перебої у зв’язку вузла через збої мережі.

Це твердження, спочатку висунуте Еріком Брюером у 2000 році та згодом підтверджене Сетом Гілбертом і Ненсі Лінч у 2002 році, закріпило теорему CAP як ключову концепцію в проектуванні розподілених систем. Розуміння теореми не вимагає двійкового вибору з трьох опцій, але заохочує до обдуманого прийняття рішень і вибору пріоритетів. Так, фінансовим установам варто надавати пріоритет узгодженості (CP), щоб гарантувати точну обробку транзакцій, приймаючи можливі затримки під час розділення мережі, щоб зберегти точну інформацію про обліковий запис. І навпаки, для розподіленої системи соціальної мережі перевагу варто надавати доступності (AP), дозволяючи доступ до дещо застарілої інформації під час розділення, щоб забезпечити безперебійне її обслуговування [40].

Визнання того, що досягнення всіх трьох гарантій одночасно, як правило, неможливо спонукає до стратегічних адаптацій на основі вимог системи. Сучасні системи баз даних використовують різноманітні методології, такі як реплікація, протоколи консенсусу та прийняття рішень на основі кворуму, щоб керувати компромісами, які наклаються теоремою CAP [41].

Вплив теореми CAP очевидний у її корисності для прискіпливої оцінки цілі проектування системної архітектори, при налаштовуанні архітектури системи відповідно до унікальних вимог їхніх програм. Визнаючи фундаментальні компроміси між узгодженістю, доступністю та толерантністю до розділів, архітектори мають можливість робити розумний вибір щодо архітектури, алгоритмів та інструментів для підвищення ефективності та надійності системи в розподілених налаштуваннях [42]

Тому для моделювання архітектури розподіленої високонавантаженої системи соціальної мережі потрібно враховувати ці три аспекти:

1) Узгодженість. В ідеальному сценарії соціальна мережа гарантує, що оновлення користувача будуть негайно видимі для всіх авторизованих користувачів. Однак у великих розподілених системах сувора узгодженість є важкою через затримку мережі та час, необхідний для оновлення даних на всіх серверах. Щоб вирішити цю проблему, у соціальній мережі можна використовувати остаточну послідовність, яка гарантує, що оновлення

поширюватимуться системою протягом розумного періоду часу, навіть якщо не миттєво [43]

2) Доступність. Соціальні мережі вимагають постійної доступності. Висока доступність вимагає реплікації даних у кількох місцях. У разі збою сервера або центру обробки даних система може безперешкодно використовувати копії даних з іншого місця. Однак ця стратегія може призвести до тимчасових невідповідностей, якщо користувачі переглядають дані до того, як усі репліки будуть оновлені [44].

3) Толерантність до розділів. У постійно мінливому середовищі інтернету мережевих розділів (збоїв зв'язку між частинами системи) не уникнути. Стійка до розділів соціальна мережа витончено справляється з такими збоями, залишаючись працездатною, навіть якщо деякі дані на мить недоступні або несинхронізовані [45]

Компроміси CAP у соціальній мережі не є монолітними. Залежно від конкретної функції баланс може змінюватися. Наприклад, компонент прямого обміну повідомленнями, швидше за все, надає пріоритет узгодженості ніж доступності, щоб гарантувати доставку повідомлень у правильній послідовності [46].

Теорема CAP є важливим аспектом при розробці та підтримці соціальних мереж. Вибір, зроблений щодо узгодженості, доступності та толерантності до розділів, глибоко формує взаємодію з користувачем і базову технічну архітектуру системи [47]. Ретельно орієнтуючись на ці компроміси, соціальні мережі можуть забезпечити переконливий баланс оперативності, надійності та цілісності даних.

Необхідність чіткої узгодженості виникає в сценаріях, які вимагають негайної й однорідної видимості даних, наприклад у фінансових транзакціях, де першорядним для кожної операції є відображення найновіших даних на всіх вузлах [48].

Незважаючи на переваги у забезпеченні точності даних, сильна узгодженість створює проблеми, зокрема щодо пропускної здатності системи та швидкості реагування. Необхідність широкої міжвузлової координації, яка необхідна для

досягнення високої узгодженості, може ненавмисно призвести до ескалації затримки та обмежень масштабованості системи [49].

І, навпаки, кінцева узгодженість пропонує прагматичний підхід у середовищах, де доступність системи та швидка можливість запису мають пріоритет над миттєвою однорідністю даних. Ця модель ураховує тимчасові неузгодженості, гарантуючи, що з часом і за відсутності нових оновлень усі вузли будудь мати єдиний стан даних [50]. Таким чином, кінцева узгодженість забезпечує стратегічну перевагу в програмах, де допустимі невеликі затримки в синхронізації даних.

Проміжною між чіткою узгодженістю і кінцевою узгодженості є причинно-наслідкова узгодженість, яка гарантує, що оновлення з причинно-наслідковою залежністю поширюються та спостерігаються в логічно узгодженій послідовності в системі. Ця модель покращує передбачуваність у змінах стану даних без необхідності суворої синхронізації, якої вимагає сувора узгодженість [51].

Управління цими моделями узгодженості в розподілених системах фундаментально підтримується консенсусними протоколами, яскравими прикладами яких є Paxos і Raft. Ці протоколи відіграють важливу роль в організації узгодження між вузлами, тим самим сприяючи уніфікованому процесу прийняття рішень, необхідному для підтримки узгодженості даних, особливо в системах, які дотримуються принципів суворої узгодженості [49].

Навмисний вибір рівнів узгодженості та консенсусних протоколів має значні наслідки для атрибутів розподіленої системи, включаючи, але не обмежуючись доступністю, затримкою та складністю архітектури [52]. Щоб знайти баланс між досягненням негайної узгодженості даних і оптимізацією продуктивності системи, необхідний стратегічний підхід до вибору відповідної моделі узгодженості.

Загалом, складна динаміка між рівнями узгодженості та консенсусними протоколами підкреслює складність проектування розподілених систем, які мають бути стійкими і ефективними. Як з’ясовано в науковому дискурсі, тонке розуміння цих елементів є критично важливим для розробки розподілених інфраструктур, через балансування узгодженості, доступності та толерантності до розділення [53].

Для базових операцій розподілених систем соціальних мереж достатньо використання кінцевої узгодженості. Але є операції користувача, як чат з іншими користувачами, які потребують причинно-наслідкової узгодженості.

1.3.3 Несправності та часткові відмови

У складному середовищі розподілених систем, що розвивається, виникнення несправностей і часткових відмов створює значні проблеми. Несправності, що характеризуються відхиленнями від очікуваної роботи системи, можуть виникати з різних причин, починаючи від помилок, що спричинені тимчасовими збоями, і закінчуючи постійними несправностями, які потребують значного втручання [54], [55].

Профілактичні заходи є одним ключовим аспектом управління несправностями в розподілених системах в напрямку запобігання та раннього виявлення. Профілактичне технічне обслуговування, яке включає регулярні оновлення програмного забезпечення та застосування виправлень, відіграє важливу роль у захисті систем від відомих вразливостей і потенційних загроз безпеці.

На передньому краї інновацій стратегії прогнозованого технічного обслуговування використовують потужність передової аналітики та алгоритмів машинного навчання для виявлення закономірностей, що вказують на неминучі несправності. Ця можливість прогнозування дозволяє системним адміністраторам вживати запобіжні дії, значно підвищуючи стійкість системи та знижуючи ймовірність непередбачених збоїв.

Останні дослідницькі зусилля ще більше розширили горизонти відмовостійкого керування. Ці дослідження підкреслюють необхідність адаптивних стратегій, які можуть динамічно пристосовуватися до різних умов несправності, забезпечуючи тим самим стійку продуктивність системи за несприятливих сценаріїв [56].

Загалом, постійна розробка складних відмовостійких механізмів у поєднанні з проактивними та прогнозними стратегіями є основою для підвищення надійності та продуктивності розподілених систем. Завдяки інтеграції розширеної відмовостійкості, заходів відновлення та превентивних підходів галузь продовжує прокладати шляхи до більш стійких і ефективних розподілених середовищ. Постійні дослідження та інновації в області відмовостійкого керування, дизайну розподілених систем і прогнозованого обслуговування спрямовані на вирішення складних проблем, пов’язаних з несправностями, провіщають майбутнє, де розподілені системи зможуть безперешкодно долати мінливості операційних збоїв.

Стратегії роботи системи у якій є несправності та часткові відмови:

- надлишок: стратегічне включення кількох екземплярів критично важливих системних елементів – будь то обладнання, програмне забезпечення чи дані — пом’якшує вплив одноточкових збоїв. Надлишок може бути просторовим (одночасна реплікація), або часовим (повторні спроби в межах часового вікна) [57].

- реплікація: підтримка синхронізованих копій даних або служб, розподілених між вузлами, забезпечує доступність у разі збою вузла. Стратегії реплікації повинні ретельно вирішувати проблеми узгодженості та синхронізації, щоб уникнути проблем із цілісністю даних [58].

- контрольні точки та відкат системи: практика періодичного запису стану системи надає можливість повернутися до відомої функціональної конфігурації після збою. Механізми контрольних точок повинні збалансувати накладні витрати на захоплення стану та потенційну вартість втраченої роботи під час відновлення [59].

Стратегії усунення несправностей:

- Виявлення збоїв: своєчасне виявлення стану збою потребує надійних механізмів моніторингу, які відстежують показники справності та продуктивності системи, а ефективне виявлення часто залежить від кореляції даних між компонентами для визначення аномалій [60].

- Діагностика: точне виділення першопричини несправності має важливе значення для вибору відповідних заходів для виправлення. Процеси діагностики

можуть включати аналіз журналів, кореляцію подій і навіть використання інструментів налагодження в складних сценаріях [61].

- Методи відновлення: конкретна стратегія відновлення залежить від характеру несправності, а потенційні дії включають перезапуск несправних компонентів, перехід до резервних систем, відновлення цілісності даних із резервних копій або, у більш складних випадках, зміну конфігурації топології системи [62].

Щоб доповнити реактивні заходи щодо відмовостійкості та відновлення, проактивні стратегії пом’якшення несправностей зосереджені на превентивному усуненні потенційних збоїв системи.

Проактивні стратегії зменшення кількості несправностей:

- регулярне технічне обслуговування та своєчасне оновлення: дотримуючись суворих графіків технічного обслуговування та своєчасно застосовуючи оновлення та виправлення, можна значно зменшити ризик збоїв системи через відомі вразливості або застарілі компоненти [63].

- балансування навантаження та ефективний розподіл ресурсів: стратегічний розподіл обчислювальних завдань і ефективне керування системними ресурсами допомагають запобігти перевантаженню будь-якого окремого компонента. Такий підхід мінімізує ймовірність збоїв через надмірне навантаження на частини системи [64].

- розширене виявлення аномалій і прогнозованих моделювання: використання машинного навчання та статистичного аналізу телеметричних даних системи дозволяє завчасно виявляти нерегулярні закономірності або тенденції, які можуть вказувати на загрозливі збої. Використання прогнозованих моделей полегшує раннє втручання, значно покращуючи здатність системи протистояти потенційним збоям [65].

Висновки до розділу 1

1. Аналіз проблеми поширення спаму та пропаганди в соціальних мережах та її рішень показав доцільність удосконалення методів розпізнавання спаму та пропаганди в соціальних мережах. Важливо, щоб метод для розпізнавання спаму та пропаганди, працював з високою точністю та міг обробити велику кількість текстів за короткий період часу.

2. Аналіз причин виникнення соціальних «бульбашок» показав, що вони виникають коли користувачі мають різні думки, щодо дискусійної теми. Відповідно з часом спільноти в соціальних мережах поляризуються, і кількість зв’язків між людьми з різних спільнот зменшується. Важливим в напрямі пом’якшення проблеми соціальних «бульбашок» є створення методу диверсифікації стрічки новин, щодо дискусійних тем, який би дозволяв більш тонке ознайомлення з різними думками, щодо дискусійної теми.

3. Аналіз архітектурних проблем і обмежень у високонавантажених розподілених системах показав, що для створення системи соціальної мережі, яка б застосовували методи класифікації текстів та диверсифікації стрічки новин в режимі близькому до реального часу, потрібно будувати систему в напрямі високої доступності та обмежитися кінцевою узгодженістю. Для побудови високодоступної системи необхідно використовувати комбінацію масштабованих технологій.

РОЗДІЛ 2. МОДЕЛІ ТА МЕТОДИ ВИЯВЛЕННЯ СПАМУ, ПРОПАГАНДИ І ЗМЕНШЕННЯ ЕФЕКТУ «БУЛЬБАШКИ» В СОЦІАЛЬНИХ МЕРЕЖАХ

У другому розділі описано переваги та недоліки застосування машинного навчання та нейромереж для класифікації спаму, пропаганди та зменшення ефекту «бульбашки»; запропоновано модель нейронна мережа для класифікації тексту у стрічці новин (НМКТуСН) на основі модифікації гібриду згорткової нейронної мережі та мережі довгої короткочасної пам'яті, яка ефективно поєднує архітектурні рішення, оптимізовані гіперпараметри і натреновані наперед векторні вбудування слів на основі нейромереж з трансформер архітектурою, для ефективної та швидкої класифікації тексту; удосконалено метод класифікації спаму і пропаганди за допомогою використання моделі НМКТуСН; запропоновано метод боротьби з ефектом «бульбашки» на основі моделі НМКТуСН, та кластеризації текстів.

2.1 Попередня обробка текстів та їх подання у векторному форматі

Для того, щоб тренувати моделі машинного навчання та нейронні мережі на текстах, перш за все потрібно розбити тексти на токени, які несуть інформацію, контекст, теми, значення, наміри. Частину слів, які не несуть семантичного значення з тексту прибирається, частина замінюється на числові значення, чи вектори чисел (вбудовані вектори). Для обробки слів у текстах використовується нормалізація, лематизація, стемінг, та токенізація.

Стемінг – це процес приведення відмінюваних (або іноді похідних) слів до їхньої словотворчої основи, основи або кореня - як правило, до письмової форми слова. Англійські слова «cat», «catlike», «catty» за допомогою стемінгу перетворяться на «cat». Стем від слова не завжди є реальним словом, наприклад стем для групи англійських слів «argue», «argues», «argued», «argus» буде «argu» [66].

Алгоритми стемінгу базуються на відсіченні суфіксів та закінчень. В правила стемінгу англійської мови входить: якщо слово має одне з закінчень «ed», «ly», «ing» то потрібно видалити дане закінчення. Є і інші модифікації стемінг алгоритмів, які перетворюють англійські слова таким чином, щоб стем слова був реальним словом. В такому випадку в правила алгоритмів включаються не тільки видалення суфіксів, але і їх заміна, як от, заміна «ies» на «y». В такому випадку «friendlies» заміниться на реальне англійське слово «friendly» , замість стему «friendl», що не є реальним словом англійської мови.

Лематизація – це процес групування відмінюваних форм слова, щоб їх можна було аналізувати, як єдине ціле, ідентифіковане за лемою слова, або словниковою формою [67]. Лематизація це більш комплексний процес чим стемінг. Для лематизації потрібно правильно визначити частину мови, та значення слова в реченні, і можливо навіть його ширше значення в контексті цілого документу. Лема для слова «walking» це «walk» і в цьому випадку алгоритми стемінгу і лематизації нададуть однаковий результат. Лема для анлійського слова «better» це «good» і алгоритм стемінгу в даному випадку не спрацює. Для цього в алгоритм лематизації потрібно включити пошук по словнику.

Для нормалізації тексту спочатку потрібно всі слова у тексті привести до маленької літери, оскільки регістр дуже часто не замінює значення слів у тексті. Наступним етапом є видалення розділових знаків, та спеціальних символів. Далі необхідно видалити стоп-слова, тобто слова, які часто зустрічаються в тексті і несуть в собі мало інформації про текст, як от, англійські слова: «which», «on», «the», «a», «at». Їх потрібно відфільтрувати, щоб методи машинного навчання та моделі нейронних мереж концентрувалися, на більш значущих частинах тексту. Наступним є обробка дат, тобто їх перетворення в один формат. Є і інші етапи нормалізації, як обробка юнікоду, тобто перевірка, що всі однакові символи, закодовані однаковими юнікод символами, та інші. Їх добавляють в залежності від потреб конкретних задач.

Токенізація тексту включає в себе розбиття тексту на слова, або н-грами. Н-грам це декілька послідовних слів в тексті. Маючи текст «Сьогодні дуже хороший

сонячний день» 2-грам цього тесту - це «хороший сонячний», 3-граму цього тексту «сьогодні дуже хороший». Після того, як текст розбили на токени, токени заміняються числовими значеннями. У технології «мішок слів» створюється словник з обмеженої кількості слів, або н-грамів (наприклад 1000), і замінється кожен токен, на відповідний номер цього токену в словнику. Таким чином замість тексту в цьому випадку, отримається послідовність, або множину чисел у векторі.

Зазвичай для тренування моделей машинного навчання далі тексти сприймаються, як невпорядкована множина чисел, а нейронні мережі сприймають текст, як послідовності чисел. Це відбувається тому, що методи машинного навчання погано працюють із послідовностями.

Якщо потрібно використовувати послідовності для машинного навчання, в цьому випадку використовують н-грами. Мінус використання н-грам в тому, що вони збільшують кількість вимірів, для тренування цих методів штучного інтелекту навчання.

Є декілька способів представити слова та тексти у числових та векторних поданнях.

Мішок слів. Слова записуються у словник обмеженого розміру (наприклад 1000) і нумеруються по порядку. Кожен текст вибірки, з якою працює метод машинного навчання в такому випадку буде кодуватися вектором розміру 1 * 1000, де в кожній клітинці стоїть, або нуль, або одиниця, в залежності, чи міститься слово словника під номером комірки, чи не міститься.

Однією з модифікацій цього векторного представлення є, вказування не 0 чи 1 в залежності того, чи слово зустрічається в тексті, а вказування скільки разів слово зустрічається серед усіх текстових наборів.

Ще одна модифікація – це використання частоти символу (англ. TF – term frequency) є вказування відношення кількості входжень даного слова у даному тексті до загальної кількості слів в даному тексті.

$$\mathrm{TF} = \frac{n_i}{\sum_k n_k}, \tag{2.1}$$

де n_i – це кількість входжень слова, що розглядається, ітерація суми в знаменнику відбувається по всіх словах.

Ще одна модифікація (англ. TF-IDF – term frequency, inverse document frequency) – враховує не тільки частоту слова в текстовій вибірці, а й інверсію частоти, з якою слово зустрічається в усіх тектах корпусу даних.

Формула для обчислення цієї інверсії (IDF):

$$\mathrm{IDF} = \log \frac{|D|}{|(w_i \text{ in } d_i)|}, \tag{2.2}$$

$$\text{TF-IDF} = \text{TF} * \text{IDF}, \tag{2.3}$$

де чисельник це кількість $|D|$ текстів корпусу даних, а w_i in d_i це кількість текстів корпусу даних, в якому зустрічається і-те слово словника.

Нехай маємо корпус даних і з двох текстів: «Сьогодні сонячна погода», «Завтра також буде сонячна погода». Пронумеруємо слова цього речення таким чином: сьогодні – 0, сонячна – 1, погода – 2, завтра 3, також – 4, буде 5.

Представлення у числовому форматі даних двох речень показано на таб. 2.1.

Таблиця 2.1.

Наявність слів в реченнях

1	1	1	0	0	0
0	1	1	1	1	1

Далі врахуємо модифікацію з кількістю слів в тексті (таб. 2.2):

Таблиця 2.2.

Кількість слів в реченнях

1	2	2	0	0	0
0	2	2	1	1	1

Сонячна і погода зустрічаються два рази, всі решта значення без змін.

Далі (таб. 2.3) перерахуємо, враховуючи, частоту (англ. TF).

Таблиця 2.3.

Частота слів в реченнях

0.25	0.5	0.5	0	0	0
0	0.5	0.5	0.25	0.25	0.25

І остання – це модифікація TF-IDF (таб. 2.4).

Таблиця 2.4.

Статистичний показник TF-IDF для речень

0.25	0	0	0	0	0
0	0	0	0.25	0.25	0.25

Ці векторні представлення передаються на вхід методам машинного навчання. Нейронні мережі можуть приймати, такі вектори на вхід так само. Але більш ефективною стратегією є передавати на вхід нейронним мережам не множину чисел, а їх послідовність.

Більш комплексними методами попередньої обробки текстів є латентний семантичний аналіз (на основі сингулярного розкладу матриці), латентне розміщення Діріхле, та вбудовані вектори, які формуються за рахунок нейронних мереж.

Розглянемо деякі моделі вбудованих векторів, побудованих за допомогою нейронних мереж, які несуть зміст слова.

Всі минулі моделі враховували лише лінійні залежності між словами. Нейронні мережі можуть захоплювати не тільки лінійні залежності.

Текстові вбудовування, які перетворюють слова і фрази в числові вектори, є трансформаційним інструментом в обробці природної мови (NLP). Ці

багатовимірні представлення фіксують складні семантичні та синтаксичні зв'язки, полегшуючи надійний аналіз мови за допомогою моделей машинного навчання.

Word2vec модель. Кожне слово подається у вигляді 100-500 вимірного вектора дійсних чисел. Коли слова натреновані, їх можна порівнювати за допомогою різних метрик. Однією з таких метрик є скалярний добуток нормалізованих векторів. Якщо він близький до одиниці, значить слова дуже подібні за семантикою. В 2013 році компанія Гугл натренувала таку модель на 100 мільярдах слів. Ці моделі вловлюють більше залежностей, за латентний семантичний аналіз. Наприклад, якщо додати Король відняти Чоловік додати Жінка, то результат вийде дуже близьким до слова Королева. Тобто з словами можна проводити певні семантичні операції.

Тренування Word2Vec пропонує відбувається на основі алгоритмічних варіантів: модель безперервного мішка слів (CBOW) та модель пропуску графів. CBOW має на меті передбачити цільове слово на основі навколишнього контексту, тоді як модель Skip-Gram інвертує цей процес.

Підхід Skip-Gram, чітко зосереджуючись на передбачуваності слова в контексті, часто дає більш якісні вбудовування. Приклад відстаней між словами можна бачити на рис. 2.1.

ФастТекст (англ. FastText) розширює підхід Word2Vec за рахунок включення інформації про підслова. ФастТекст розроблений компанією Фейсбук. ФастТекст на відмінну від Word2Vec працює не з словами, а н-грамами символів. Це забезпечує механізм для представлення рідкісних або позасловникових слів, пропонуючи більш тонке представлення морфологічної структури.

GloVe використовує глобальну статистику вживання слів з корпусу, застосовуючи методи матричної факторизації для створення семантично багатих векторів слів.

Здатність цих методів вбудовування фіксувати тонкі семантичні зв'язки робить їх особливо цінними для таких тонких завдань, як виявлення пропаганди. Пропаганда часто покладається на неявне значення, конотацію та маніпулювання словесними асоціаціями.

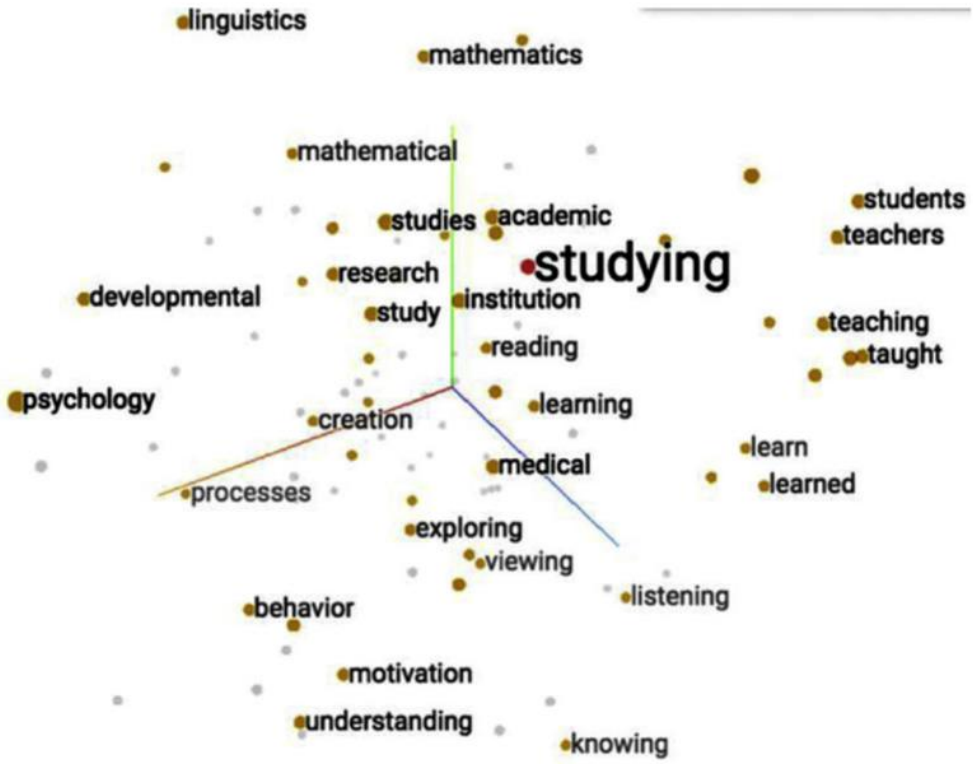


Рис. 2.1. Метод основних компонент для зображення векторного представлення слів моделі Word2Vec. Передрук з [68]

Представляючи слова як щільні, контекстно-залежні вектори, методи вбудовування дозволяють моделям машинного навчання виявляти приховані патерни, характерні для пропагандистської мови.

Вибір найбільш підходящої моделі вбудовування вимагає врахування конкретного завдання. Крім того, такі методи, як налаштування гіперпараметрів, відіграють вирішальну роль у покращенні роботи моделі. Word2Vec часто демонструє найкращі результати в задачах семантичної схожості; GloVe демонструє сильні сторони в задачах аналогії слів; а здатність FastText працювати з термінами OOV робить його особливо цінним для доменів зі спеціалізованою лексикою або такою, що швидко еволюціонує.

У сфері рекурентних нейронних мереж (англ. RNN), зокрема таких архітектур, як мережі з довгою короткочасною пам'яттю (англ. LSTM) і вентильними рекурентними вузлами (англ. GRU), текстові вбудування відіграють

важливу роль. Їх значення полягає в тому, що вони сприяють ефективній обробці послідовних даних, особливо природної мови.

Притаманна природній мові мінливість довжини послідовностей створює певні труднощі для нейроних мереж. Вбудовування забезпечують послідовне представлення слів, полегшуючи обробку текстових послідовностей різної довжини. Такий метод, як падинг, гарантує, що нейроні мережі можуть однаково працювати з партіями послідовностей, незалежно від їхньої індивідуальної довжини.

Використання попередньо навчених вбудовувань слів, отриманих з великих текстових масивів, вносить в архітектуру нейроних мереж багатство лінгвістичних знань. Ці попередньо навчені вектори діють як основа, прискорюючи продуктивність моделі та покращуючи узагальнення нових даних, особливо в сценаріях з обмеженим обсягом навчальних даних.

Для спеціалізованих застосувань у таких галузях, як медицина або право, специфічні для домену вбудовування виявляються безцінними. Ці вбудовування, навчені на корпусах, специфічних для відповідної галузі, забезпечують ШНМ лінгвістичним розумінням, необхідним для вирішення завдань, що характеризуються спеціалізованою термінологією та нюансованими поняттями.

Створення вбудовування зображень, паралельно зі створенням текстових вбудовувань за допомогою таких технологій, як Word2Vec, GloVe і FastText, є наріжним каменем у галузі комп'ютерного зору. Переважно завдяки досягненням у галузі глибокого навчання, ці методології створюють щільні векторні репрезентації, які інкапсулюють квінтесенцію візуальних атрибутів зображень.

CNN вважаються найкращою архітектурою для отримання зображень, що вбудовуються в текст. Їх ефективність пояснюється здатністю методично аналізувати зображення за допомогою послідовності згорткових фільтрів, таким чином виділяючи складні ознаки і створюючи ієрархічні представлення. Вкладення, отримані на виході кінцевих шарів мережі, стисло підсумовують внутрішній зміст зображень.

Використання попередньо навчених моделей, які пройшли інтенсивне навчання на великих наборах даних, таких як ImageNet, значно прискорює процес генерації вбудовувань. Ці моделі, що пройшли навчання на безлічі зображень, слугують готовими до використання екстракторами ознак, які можна легко адаптувати до нових наборів даних. Варті уваги попередньо навчені моделі охоплюють:

- VGG: Розроблені групою візуальної графіки Оксфордського університету, моделі VGG-16 і VGG-19 широко відомі своєю ефективністю в задачах вбудовування зображень.
- ResNet: Відома своїми інноваційними "залишковими" зв'язками, ResNet полегшує навчання мереж безпрецедентної глибини, завдяки чому стала найкращою моделлю для вбудовування зображень.
- Inception (і InceptionResNet): Характеризується складною структурою, ця модель прагне перевершити показники продуктивності, встановлені попередніми архітектурами.

Візуальні трансформатори представляють революційне застосування архітектури трансформаторів, орієнтованої на увагу, яка спочатку була розроблена для задач обробки природної мови, для аналізу зображень. Концептуалізуючи сегменти зображення як послідовні елементи, подібні до слів, ViT використовують механізми самоуваги, щоб генерувати вкраплення виняткової виразності.

CLIP (Contrastive Language-Image Pre-training) від OpenAI - це новаторська модель, яка розшифровує візуальні концепції за допомогою описів природною мовою. Вона чудово справляється зі створенням вбудовувань, які узгоджуються як із зображеннями, так і з текстом, що робить її безцінною для додатків, які потребують інтеграції візуальних і текстових даних.

Застосування цих методологій полегшує синтез вбудовувань, які виділяють складну візуальну інформацію, що міститься в зображеннях, підтримуючи широкий спектр застосувань комп'ютерного зору, таких як класифікація зображень, пошук і всебічний аналіз.

2.2 Застосуваня нейронних мереж та методів машиного навчання для класифікації текстів у соціальних мережах

2.2.1 Наївний класифікатор Байєса

Наївний класифікатор Байєса є прикладом імовірнісної методології в штучному інетелекті, а саме машинному навчання, спеціально розробленої для задач класифікації. Його теоретичною основою є теорема Байєса, яка робить спрощене припущення, що окремі ознаки є умовно незалежними від змінної класу. Хоча це припущення може здатися надто спрощеним (звідси і термін "наївний"), емпіричні дані демонструють практичну корисність наївних байєсівських класифікаторів у широкому спектрі прикладних контекстів.

Теорема Байєса дає математичне формулювання для обчислення ймовірності події (A) на основі попередніх знань про умови, потенційно пов'язані з її настанням (B).

Це можна виразити так:

$$P(A|B) = P(B|A) * P(A) / P(B) \quad (2.5)$$

де:

- P(A|B) – це умовна ймовірність класу (A) при заданій події B (апостеріорна ймовірність).
- P(B | A) – це умовна ймовірність класу (B) при заданій події (A);
- P(A) – попередня ймовірність події A;
- P(B) – попередня ймовірність події B.

У контексті наївного Байєса ця теорема дозволяє оцінити ймовірність того, що точка даних належить до певного класу, враховуючи пов'язаний з нею набір ознак [69]. На етапі навчання класифікатора визначаються ймовірності належності точок даних до різних класів на основі їхніх ознак. При отриманні нових даних класифікатор обчислює апостеріорні ймовірності для кожного класу, а потім відносить точку даних до класу, який дає максимальну апостеріорну ймовірність.

Завдяки обчислювальній ефективності, концептуальній простоті та успіху в різних галузях наївний байєсівський класифікатор широко використовується.

Серед помітних прикладів [70]:

- Виявлення спаму: Розрізнення спаму та легітимної електронної пошти шляхом аналізу шаблонів використання слів.
- Аналіз настроїв: Класифікація афективного тону, вираженого в текстових даних, як позитивного, негативного або нейтрального.
- Класифікація документів: Алгоритмічна класифікація документів за попередньо визначеними тематичними категоріями.
- Медична діагностика: Визначення ймовірності різних медичних станів на основі даних про пацієнта та спостережуваних симптомів.

Сильні сторони наївного Байєса:

- Потребує відносно невеликого набору навчальних даних для оцінки необхідних параметрів для прогнозування.
- Демонструє швидку класифікацію порівняно з більш складними алгоритмами.
- Логіка, що лежить в основі, і процес прийняття рішень легко доступні для аналізу.

Слабкі сторони наївного Байєса:

- Припущення про незалежність ознак може бути порушене в реальних наборах даних, що потенційно може вплинути на точність моделі.
- До вихідних ймовірностей слід ставитися з певним застереженням, оскільки наївний Байєс не завжди є надійним оцінювачем ймовірностей.
- Класифікатор може присвоїти нульову ймовірність ознакам, які не зустрічалися на етапі навчання, що може призвести до помилкової класифікації.

Наївний Байєс може бути ефективним у виявленні спаму завдяки своїй швидкості та здатності працювати з невеликими наборами даних. Однак його спрощувальні припущення та проблема нульової частоти обмежують здатність точно виявляти пропаганду, яка часто залежить від контексту і тонкощів мови [71].

2.2.2 Дерево рішень

Дерева рішень одним із методів машинного навчання, дозволяючи класифікації, регресію та інших форм задач прогнозування. Завдяки своїй характерній деревоподібній структурі ці моделі дозволяють легше зрозуміти і візуально представити шляхи прийняття рішень. У цій статті досліджується внутрішня робота дерев рішень, включаючи їх алгоритмічні основи та широкий спектр застосувань.

Дерева рішень відображають різні шляхи прийняття рішень разом з їх можливими результатами, включаючи ймовірності подій, пов'язані з ними витрати і загальні вигоди. Їх структура, схожа на блок-схему, включає внутрішні вузли для тестування атрибутів, гілки, які показують результати цих тестів, і вузли листків, які показують остаточні рішення після оцінки атрибутів [72]. Шляхи від кореня до листка представляють правила класифікації або прийняття рішень.

По суті, алгоритми дерев рішень намагаються визначити ознаку, яка найкраще розділяє набір даних на окремі групи. Розбиття даних на групи на основі цих критеріїв триває доти, доки не буде досягнуто точки зупинки, яка може настати, коли дерево досягне певної глибини, або коли у вузлі буде мінімальна кількість зразків, достатня для розбиття даних на групи.

Існує кілька стратегій побудови дерев рішень за допомогою таких алгоритмів, як ID3, C4.5 і CART (Дерева класифікації та регресії). Ці методи відрізняються тим, як вони вибирають атрибути для розбиття, обробляють безперервні дані, працюють з відсутньою інформацією та застосовують стратегії обрізання, щоб уникнути надмірної пристосованості.

Навчання на основі дерев рішень теж включає методи, які є інкрементними, розподіленими і не покладаються виключно на жадібні алгоритми. Деякі підходи виконують ретельний пошук найкращої структури дерева, беручи до уваги обмеження, які покращують надійність і справедливість дерева [73].

Завдяки своїй простоті, легкості інтерпретації та ефективності, дерева рішень використовуються в багатьох сферах:

- В охороні здоров'я вони допомагають аналізувати дані про пацієнтів, діагностувати захворювання і рекомендувати лікування.
- У фінансах вони використовуються для кредитного скорингу, управління ризиками та виявлення шахрайства.
- У маркетингу та управлінні взаємовідносинами з клієнтами вони допомагають сегментувати клієнтів, прогнозувати їхню поведінку та розробляти цільові стратегії.
- У виробництві та контролі якості вони передбачають поломки обладнання, вдосконалюють процеси та забезпечують якість.
- В науці про навколишнє середовище вони прогнозують зміни в навколишньому середовищі та оцінюють екологічні наслідки.

Дерева рішень можуть страждати від перенавчання, створюючи надто складні моделі, які відображають аномалії навчальних даних. Такі методи, як обрізка та ансамблеві стратегії, такі як випадковий ліс (англ. Random Forests) та градієнтне прискорення (англ. Gradient Boosting), допомагають пом'якшити цю проблему. Однак, коли дерева стають складнішими, їхня інтерпретованість може зменшитися.

Дерева рішень цінуються за їхню простоту, інтерпретованість і універсальність, що закріплює їхню роль як життєво важливих інструментів для науковців з даних та експертів з машинного навчання. Їх широке застосування підкреслює їх здатність вирішувати проблеми класифікації та регресії. Оскільки машинне навчання продовжує розвиватися, дерева рішень залишатимуться фундаментальними, слугуючи як окремі моделі або як ключові елементи більш складних ансамблевих методів.

Застосування дерев рішень для виявлення спаму та пропаганди в соціальних мережах показало багатообіцяючі результати завдяки їхній здатності надавати інтерпретовані моделі [74]. Ця ефективність підкреслюється їхнім застосуванням у різних реальних сценаріях, включаючи виявлення спаму в електронній пошті та ідентифікацію пропаганди під час великих глобальних подій, таких як пандемія COVID-19 .

Хоча дерева рішень виявляють спам і пропаганду, існують перешкоди для їхнього використання в складинх реальних ситуаціях. Основною проблемою є перенавчання, коли модель чудово працює на навчальних даних, але не справляється з новими, непередбачуваними даними [75]. Для подолання цієї проблеми потрібне ретельне налаштування, а для забезпечення здатності моделі до узагальнення може знадобитися використання ансамблевих стратегій, таких як випадкові ліси.

2.2.3 Випадковий ліс

Алгоритм випадкового лісу (англ. Random Forest) - це передовий і багатогранний метод керованого навчання у штучному інтелекті, який цінується за свою майстерність у вирішенні як класифікаційних, так і регресійних задач. Цей метод ґрунтується на побудові декількох дерев рішень під час фази навчання. Ключовим фактором ефективності методу Random Forest є його опора на ансамблеве навчання – стратегію, яка об'єднує прогнози з декількох моделей для підвищення загальної точності, надійності та узагальнюваності прогнозу.

В основі методології випадкового лісу лежить принцип бутстрап-агрегування, або пакування. Цей процес передбачає створення декількох підмножин початкового навчального набору даних із заміною, щоб навчити кожне складове дерево в лісі. Такий підхід не тільки посилює здатність моделі до узагальнення, але й допомагає зменшити дисперсію та запобігти надмірному пристосуванню – двом важливим проблемам, притаманним дереву рішень. Ще одним підвищенням продуктивності алгоритму є введення випадковості в процес відбору ознак під час розбиття вузлів, що сприяє різноманітності в ансамблі моделі, яка часто призводить до кращих показників продуктивності [76].

Алгоритм випадкового лісу застосовується в широкому спектрі галузей, включаючи, зокрема, банківську справу та фінанси для оцінки ризиків і виявлення шахрайства, охорону здоров'я для прогнозування захворювань і діагностики пацієнтів, електронну комерцію для створення персоналізованих рекомендацій і

прогнозування продажів, фінансові ринки для прогнозування цін на акції та біоінформатику для аналізу експресії генів і виявлення маркерів хвороб [77].

Серед сильних сторін алгоритму – його адаптивність до задач класифікації та регресії, ефективність в обробці великих масивів даних та стійкість до неповних даних. Крім того, алгоритм характеризується мінімальними вимогами до попередньої обробки даних і здатністю ефективно працювати як з категоріальними, так і з числовими даними. Тим не менш, складність алгоритму та дещо непрозора природа ансамблевої моделі створюють проблеми для інтерпретації та аналізу, особливо у порівнянні з простішими моделями, такими як одиночні дерева рішень.

Таким чином, алгоритм випадкового лісу є потужним інструментом в інструментарії науки про дані, що забезпечує гармонійний баланс між точністю, зручністю для користувача та адаптивністю. Незважаючи на перешкоди, пов'язані з його складністю та інтерпретованістю, переваги застосування методу випадкових лісів у широкому спектрі завдань прогнозованого моделювання часто переважають ці недоліки, що підкреслює його цінність у сучасних дослідженнях в галузі аналізу даних.

У сфері виявлення спаму в соціальних мережах Random Forest виявився надзвичайно досить надійним. Дослідження, що порівнюють Random Forest з іншими методами класифікації, як от порівняння за наївним Байєсівським класифікатором, підкреслюють його точність [78].

Щоб застосувати Random Forest для виявлення спаму і пропаганди, алгоритм повинен бути навчений на великих маркованих наборах даних, що містять приклади спаму, пропаганди і справжнього контенту. Використовуючи ансамбль дерев рішень, модель підвищує точність класифікації завдяки об'єднанню прогнозів окремих дерев. Цей підхід виявляється особливо корисним у середовищі соціальних мереж, де дані можуть бути багатогранними і потенційно оманливими [79].

Якість і масштаб навчальних даних суттєво впливають на продуктивність моделі. Дослідження показують, що набори даних обмеженого розміру можуть перешкоджати здатності моделі добре узагальнювати дані з різних платформ

соціальних мереж, що є потенційною сферою для подальших досліджень і вдосконалення.

2.2.4 Машини опорних векторів

Машини опорних векторів (англ. support vector machine – SVM) виділяються як складна і добре обґрунтована група методів керованого навчання, відома своєю ефективністю у вирішенні завдань класифікації, регресії та виявлення аномалій. В основі SVM лежить стратегія побудови гіперплощини в N-вимірному просторі для ознак, яка ідеально розділяє різні категорії. Основна мета полягає у визначенні гіперплощини, яка забезпечує найбільший запас або найвіддаленішу відстань між найближчими точками даних (так званими опорними векторами), що належать до кожної категорії [80].

Пошук оптимальної гіперплощини здійснюється за допомогою оптимізаційної задачі квадратичного програмування. Це завдання має на меті збільшити різницю між категоріями, одночасно зменшуючи кількість помилкових класифікацій точок даних. Для ситуацій, коли дані не можуть бути лінійно розділені, SVM використовують "трюк ядра". Цей підхід використовує функції ядра для тонкого проектування вхідних даних у простір більшої розмірності, де можливе лінійне розділення, оминаючи при цьому необхідність прямого обчислення координат у цьому розширеному просторі. Популярні функції ядра включають

- Лінійне: Підходить для наборів даних, які можна лінійно розділити.
- Поліноміальне: Призначена для захоплення нелінійних граничних рішень певного ступеня.
- Радіальна базова функція (RBF): Пристосована для широкого спектру нелінійних задач класифікації.

Надійність і теоретична база SVM роблять їх незамінними в широкому спектрі наукових галузей [81]:

- Біомедицина: SVM відіграють вирішальну роль у діагностиці захворювань, прогнозуванні прогнозів (наприклад, класифікація раку) та полегшенні комп'ютерного відкриття ліків.
- Комп'ютерний зір та розпізнавання мови: Необхідні для розпізнавання зображень і об'єктів, ідентифікації осіб і технологій розпізнавання мови.
- Обробка природної мови: SVM допомагають класифікувати тексти, аналізувати настрої та автоматизувати класифікацію документів.

Незважаючи на свою потужність, SVM вимагають ретельного налаштування параметрів. Вибір функції ядра та її параметрів, а також параметра регуляризації, який балансує між розміром маржі та помилкою класифікації, сильно впливає на результат. Оптимізація гіперпараметрів, як правило, шляхом перехресної перевірки, має вирішальне значення для підвищення продуктивності SVM. Крім того, навчання SVM на великих масивах даних може вимагати значних обчислювальних затрат, що потребує ефективних методів оптимізації.

Як керовані моделі машинного навчання, SVM характеризуються здатністю генерувати гіперплощини в багатовимірних просторах, що робить їх придатними для широкого спектру застосувань, включаючи завдання класифікації та регресії.

У контексті виявлення спаму в соціальних мережах, SVM зарекомендували себе, як хороший метод класифікації. Чудова продуктивність SVM в цій області, особливо при використанні ядра радіально-базисної функції (RBF) і здатність SVM орієнтуватися в складних структурах даних мають вирішальне значення для їхнього успіху у виявленні спаму.

SVM також використовуються для виявлення пропаганди в соціальних мережах. Хоча альтернативні методи можуть перевершувати SVM у певних сценаріях виявлення пропаганди, SVM зберігають свою актуальність як невід'ємна частина комплексного арсеналу машинного навчання [82]. Емпіричні дані свідчать про те, що інтеграція інженерії ознак з класифікацією на основі SVM може призвести до значного прогресу в ідентифікації текстової пропаганди.

Важливо розуміти, що ефективність SVM залежить від вибору оптимальної функції ядра та ретельного калібрування пов'язаних з нею параметрів. Наявність

шуму в наборі даних може вплинути на точність класифікації. Таким чином, ретельний вибір і розробка ознак є критично важливими для ефективності SVM-орієнтованих систем виявлення спаму і пропаганди [83]. Підходи, що поєднують гібридні стратегії інженерії ознак, можуть ще більше підвищити ефективність класифікації.

Машини опорних векторів є чудовим інструментом для виявлення спаму та пропаганди в соціальних мережах. Їх перевага полягає в здатності орієнтуватися в просторах високої розмірності та гнучкості, яку надає вибір функцій ядра. Однак, щоб розкрити весь їхній потенціал, необхідно приділити значну увагу налаштуванню параметрів, вибору функцій та управлінню шумами. Поєднання SVM з комплексною інженерією ознак і додатковими методами машинного навчання виявляється найбільш життєздатним шляхом до успіху в цих складних сферах.

2.2.5 Згорткові нейронні мережі

Згорткові нейронні мережі (ЗНМ – англ. convolutional neural networks – CNN) являють собою спеціалізовану підгрупу в більш широкому спектрі глибоких нейронних мереж, відому своєю здатністю обробляти дані, що характеризуються решітчастою структурою, з особливим акцентом на аналізі зображень. Ці мережі добре зарекомендували себе в галузі розпізнавання зображень, класифікації та інших подібних сферах завдяки своїй здатності автономно вивчати ієрархії просторових ознак на основі вхідних даних.

Основні механізми роботи згорткових нейронних мереж.

Суть роботи ЗНМ полягає в послідовності шарів, які перетворюють вхідне зображення на розпізнаваний вихідний результат.

Основні шари, що відіграють центральну роль у цьому перетворенні, включають [84]:

- Згорткові шари: як основні компоненти ЗНМ, ці шари використовують різні фільтри, які застосовуються до зображення. Кожен фільтр спеціально

налаштований на виявленя певних візуальних об'єктів. Це призводить до створення карт об'єктів, які фіксують розташування цих об'єктів на зображенні;

- Функції активації: після шарів згортки застосовуються нелінійні функції активації, такі як зрізаний лінійний вузол (ReLU). Ці функції мають вирішальне значення для того, щоб мережа могла фіксувати та вивчати складні патерни, таким чином дозволяючи їй моделювати складні взаємозв'язки в даних;
- Об'єднання шарів: у цих шарах зазвичай застосовуються такі операції, як максимальне об'єднання, щоб зменшити розміри карт об'єктів. Таке зменшення не тільки зменшує кількість параметрів і обчислювальні вимоги, але й підвищує стійкість мережі до незначних змін у вхідних даних, таких як зсуви і повороти;
- Повністю з'єднані шари: архітектура ЗНМ часто доповнюється повністю зв'язаними шарами. Ці шари необхідні для об'єднання високорівневих ознак, витягнутих попередніми шарами, для виконання остаточного завдання класифікації на основі вивчених даних.

ЗНМ ознаменували зміну парадигми комп'ютерного зору і знайшли застосування в широкому спектрі галузей [85]:

- Розпізнавання зображень і відео: Передові системи класифікації зображень і відео, ЗНМ дозволяють застосовувати їх у різних сферах - від маркування зображень до відеоспостереження і взаємодії на основі жестів.
- Медичний аналіз зображень: У медичній галузі ЗНИ відіграють ключову роль у виявленні та діагностиці захворювань за допомогою медичних зображень, таких як рентгенівські знімки та МРТ, допомагаючи виявляти пухлини, відстежувати прогресування хвороби і полегшувати багато інших завдань, пов'язаних з охороною здоров'я.
- Автономні системи: Критично важливі для роботи безпілотних транспортних засобів, ЗНМ допомагають інтерпретувати дорожні знаки,

виявляти пішоходів і перешкоди, а також загалом забезпечують безпечну навігацію.

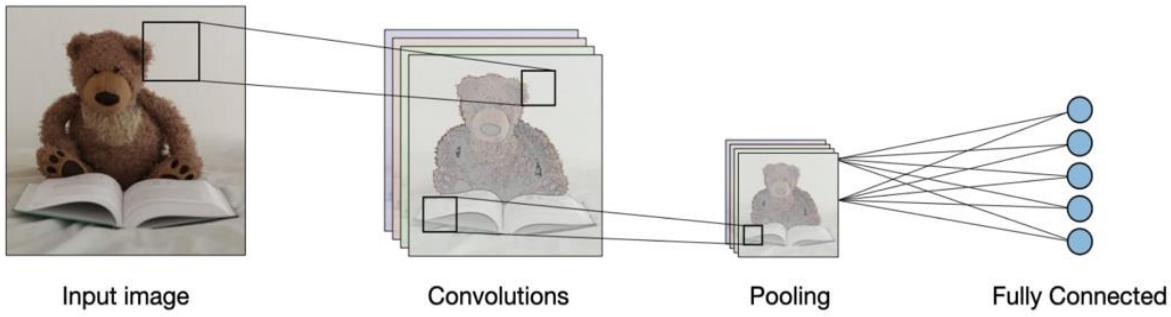


Рис. 2.2. Архітектура традиційної згорткової нейронної мережі. Передрук з [86]

У сучасному ландшафті цифрових комунікацій застосування згорткових нейронних мереж виходить за межі їхньої традиційної сфери обробки зображень і поширюється на аналіз тексту та контенту. Це розширення особливо помітне у сфері соціальних мереж, де ЗНМ використовуються для виявлення спаму і пропаганди.

Поєднання ЗНМ з мережами довготривалої короткочасної пам'яті (ДКЧП) для виявлення спаму в соціальних мережах є прикладом інтегративної архітектури, яка використовує сильні сторони обох технологій. ЗНМ відмінно виділяють локальні та позиційно незмінні ознаки з текстових даних, тоді як ДКЧП вправно фіксують контекстуальні нюанси та часову динаміку, що сприяє надійному аналізу в умовах лінгвістичного розмаїття.

Незважаючи на ці досягнення, ця галузь стикається зі значними проблемами, зокрема потребою у великих маркованих наборах даних для навчання моделей, обчислювальними вимогами архітектур глибокого навчання та складністю інтерпретації моделей. Майбутні дослідження будуть спрямовані на вирішення цих проблем, зосереджуючись на підвищенні ефективності моделей, зменшенні залежності від великих наборів даних і поліпшенні інтерпретації [87].

2.2.6 Рекурентні нейронні мережі

Рекурентні нейронні мережі (РНМ – англ. recurrent neural networks – RNN) це унікальна підгрупа штучних нейронних мереж, відома своєю здатністю обробляти послідовні дані. РНМ використовують внутрішній стан для розпізнавання закономірностей у таких послідовностях, як текст, генетичні послідовності, почерк і дані часових рядів, і відрізняються від традиційних мереж прямого поширення тим, що використовують попередні обчислення для впливу на поточні результати. Ця здатність враховувати послідовний контекст робить РНМ особливо ефективними для завдань, які вимагають розуміння часових залежностей.

РНМ включають петлі зворотного зв'язку, які діють як обчислювальна пам'ять, зберігаючи інформацію з попередніх входів. Вони працюють у циклі, де результат кожного кроку залежить від попередніх результатів, що дозволяє мережі добре справлятися із завданнями, що включають послідовності

Незважаючи на свої сильні сторони, РНМ стикаються з труднощами, що пов'язані зі зникаючими та вибухаючими градієнтами, особливо коли мають справу з довгими послідовностями. Впровадження моделей довготривалої короткочасної пам'яті – ДКЧП (англ. Long-short term memory -LSTM) - та закритих рекурентних блоків (анл. Gated recurrent unit – GRU) допомогло подолати ці проблеми, забезпечивши механізми для кращого контролю над потоком інформації [88].

RNN знаходять застосування в широкому спектрі областей, включаючи моделювання мови, розпізнавання мови, машинний переклад і прогнозування часових рядів. Рекурентні нейронні мережі добре справляються з обробкою та аналізом послідовних даних, що притаманно текстовій та мовленнєвій комунікації. Ця характеристика робить їх особливо ефективними у виявленні та зменшенні впливу спаму та пропаганди, які часто проявляються через соціальні мережі та електронні канали зв'язку.

Гібридна система глибокого навчання, яка поєднує згорткові нейронні мережі з рекурентними нейронними мережами, продемонструвала високу точність

у 94,28% у розпізнаванні арабськомовного спаму в Twitter, що визнано складним завданням [89].

Окрім класифікації спаму, рекурентні нейронні мережі відіграють важливу роль у виявленні пропаганди та дезінформації. Її здатність контекстуалізувати інформацію в часі є ключовою для цього застосування. Розгортання архітектур довготривалої короткочасної пам'яті (LSTM) і закритих рекурентних блоків (GRU) в ініціативах з раннього виявлення чуток дало багатообіцяючі результати, підкресливши їхню майстерність у відстеженні розвитку дезінформаційних кампаній. Тим не менш, важливо розуміти, що продуктивність RNN залежить від якості та обсягу навчальних даних. Крім того, ці мережі повинні постійно адаптуватися до нових стратегій, що застосовуються спамерами і пропагандистами [90].

На рис. 2.3 поназано приклад того, як працює рекурентна нейронна мережа.

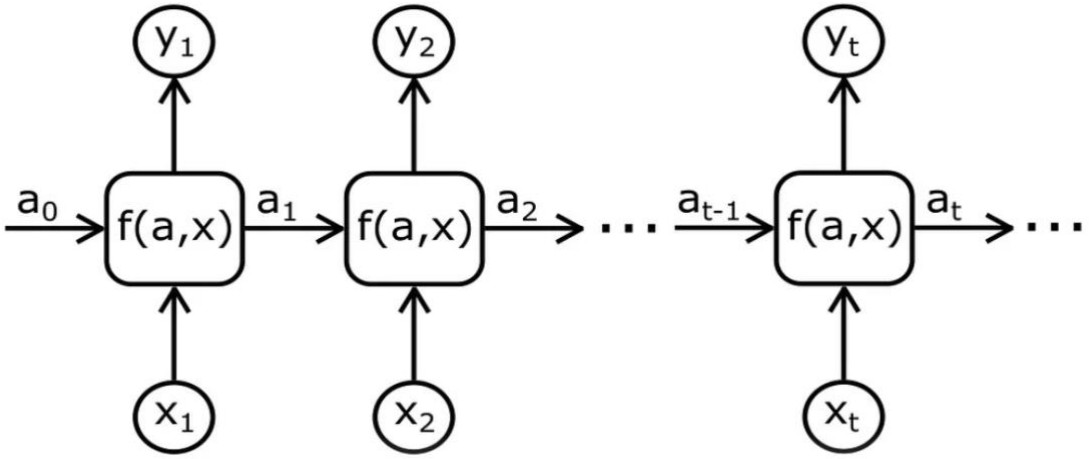


Рисунок 2.3. Архітектура рекурентної нейронної мережі. Передрук з [91]

Від стандартної нейронної мережі прямого поширення, вона відрізняється тим, що зберігає «пам’ять» про минулі слова. Рисунок можна інтерпритувати таким чином, що x_i це токени (наприклад слова, чи н-грами у числовому представленні), y_i це результат, який видає нейрона мережа для слова x_i. На вхід нейронній мережі подається не тільки x_i, а ще і деяка інформація про стан системи a_i.

Формально це записується наступним чином:

$$a_i = g_1 (w_{aa} a_{i-1} + w_{ax} x_i + b_a) \quad (2.4)$$

$$y_i = g_2 (w_{ya} a_{i-1} + b_y) \quad (2.5)$$

де: w_{aa}, w_{ax}, w_{ya} – значення на початку виконання рекурентної нейроної мережі, що ініціалізуються випадковими числами; A_0 , b_a, b_y – значення, що ініціалізуються випадковими числами, або нулями; G – це функція активації.

Популярні функції активації показано на рис. 2.4.

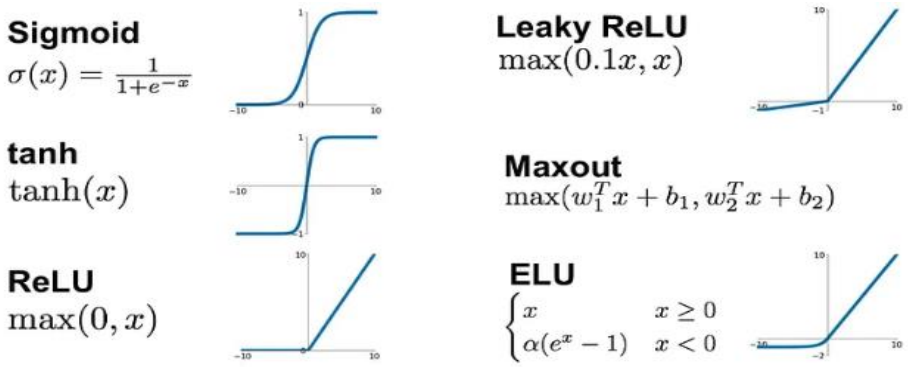


Рисунок 2.4. Функції активації. Передрук з [92]

Якщо взяти за функцію активації лінійну функцію, то нейронна мережа матиме змогу лише створювати лінійну комбінацію вхідних змін, і алгоритм навчання нейронної мережі насправді буде подібний до алгоритму навчання лінійної регресії. Тому функція активації повинна бути нелінійною.

Формули:

$$a_i = g_1 (w_{aa} a_{i-1} + w_{ax} x_i + b_a) \quad (2.6)$$

$$y_i = g_2 (w_{ya} a_{i-1} + b_y) \quad (2.7)$$

або:

$$a_i = g_1 (w_a [a_{i-1}, x_i] + b_a) \quad (2.8)$$

$$y_i = g_2 (w_{ya} a_{i-1} + b_y) \quad (2.9)$$

Це можливо, оскільки w_{aa}, w_{ax} мають однакову кількість рядків, яка рівна довжині вектора x_i.

Через зникаючий градієнт рекурентні глибинні нейронні мережі мають проблему з тим, що можуть «забувати» слова у довгих текстах чи реченнях. Через цю проблему було запропоновано дві модифікації рекурентних нейронних мереж – ВРВ вентильний рекурентний вузол (англ. GRU – Gated recurrent units) і ДКЧП довга короткочасна пам’ять (англ. LSTM – long short-term memory).

На рис.2.5.надано формули для ВРВ (англ. GRU) і ДКЧП (англ. LSTM).

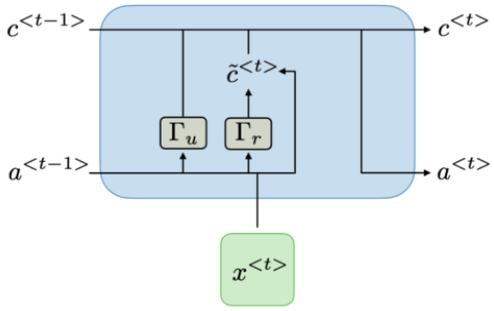
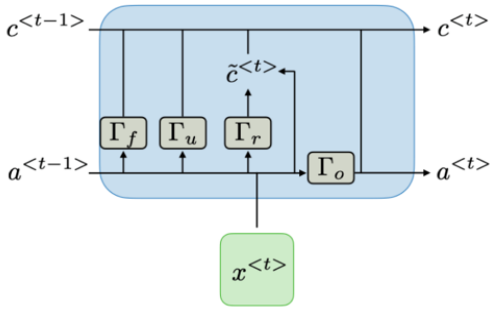
Characterization	**Gated Recurrent Unit** (GRU)	**Long Short-Term Memory** (LSTM)
$\tilde{c}^{<t>}$	$\tanh(W_c[\Gamma_r \star a^{<t-1>}, x^{<t>}] + b_c)$	$\tanh(W_c[\Gamma_r \star a^{<t-1>}, x^{<t>}] + b_c)$
$c^{<t>}$	$\Gamma_u \star \tilde{c}^{<t>} + (1 - \Gamma_u) \star c^{<t-1>}$	$\Gamma_u \star \tilde{c}^{<t>} + \Gamma_f \star c^{<t-1>}$
$a^{<t>}$	$c^{<t>}$	$\Gamma_o \star c^{<t>}$
Dependencies		

Рисунок 2.5. Формули для LSTM та GRU. Передрук з [93]

«Ворота» Γ_u, та Γ_r обчислюються формулами:

$$\Gamma_i = g(W_i[c_{t-1}, x_t] + b_i) \quad , \tag{2.10}$$

де і може приймати значення u та r, а g – це сигмоїд функція.

Модифікація РНН – ВРВ та ДКЧП були створені, бо особливо в довгих речення виникала проблема зникаючого градієнту. «Ворота» в ВРВ та ДКЧП дозволяю «пам’ятати» попередні значення послідовності.

2.2.7 Нейронні мережі з трансформер архітектурою

Моделі з трансформер архітектурою (рис. 2.6) представляють категорію нейромережевих архітектур, спеціально розроблених для обробки послідовних даних, включаючи текст та інформацію часових рядів.

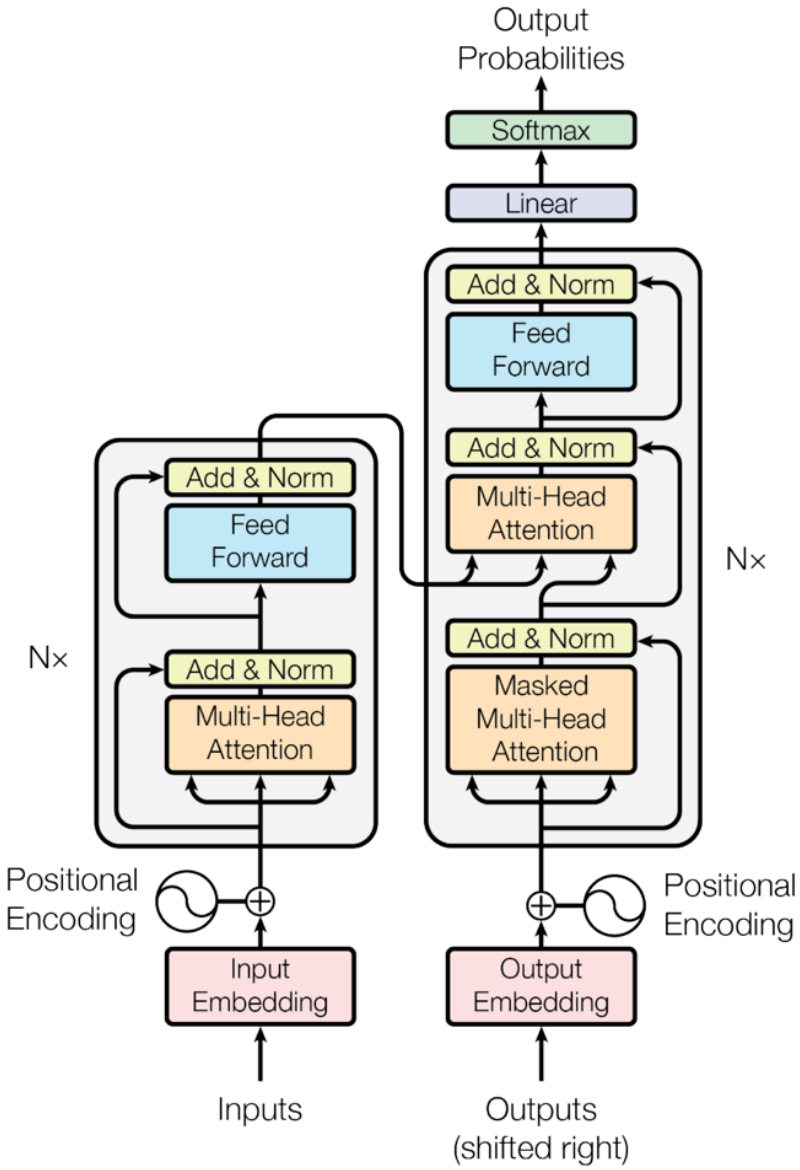


Рисунок 2.6. Нейронна мережа з трансформер архітектурою. Репринт з [94]

Ці моделі були представлені у фундаментальній статті "Attention Is All You Need" команди Google Brain 2017 року, що ознаменувала значний прогрес у галузі машинного навчання [94].

В основі трансформаційних моделей лежить механізм уваги, який динамічно розподіляє різний ступінь значущості між різними частинами вхідних даних під час генерації кожного сегмента вихідних даних. Цей механізм здатен вловлювати довгострокові залежності більш ефективно, ніж попередні моделі, які зазвичай використовувалися для подібних завдань.

Відмінні характеристики трансформаторних моделей можна підсумувати наступним чином:

- повна залежність від механізмів уваги для розпізнавання взаємозв'язків у вхідних даних, що свідомо уникає традиційних методів, таких як рекурентність і згортка;
- здатність обробляти вхідні послідовності паралельно, що підвищує ефективність навчання порівняно з послідовними обчисленнями;
- структурована конструкція, що складається з кодувальника і декодувальника: кодувальник перетворює вхідну послідовність у складне абстрактне представлення, тоді як декодувальник систематично генерує вихідну послідовність на основі цього представлення;
- реалізація багатоголової уваги, що дозволяє моделі одночасно звертати увагу на різні аспекти вхідних даних з різних підпросторів представлення протягом всієї послідовності.

Видатні приклади нейронних мереж з трансформер архітектурою, такі як BERT, GPT, T5 і ViT, відіграли ключову роль у просуванні вперед останніх досягнень у великомасштабних мовних моделях і мультимодальному навчанні, що підкреслює трансформаційний вплив цієї архітектури на сучасні спроби машинного навчання.

2.3 Створення моделі класифікації текстів та методу розпізнавання спаму та пропаганди

У даній роботі створена модель нейронної мережі на основі комбінації нейронних мереж ЗНМ, ДКЧП та перетворення слів у векторні представлення.

Для тренування та тестування запропонованої нейронної мережі для детекції спаму будемо використовувати корпус текстів SMS Spam Collection Dataset [95].

Приклад текстів перекладеної на українську мову з цього корпусу текстів:

1) ви ПЕРЕМОЖЕЦЬ!!! Як цінний клієнт мережі, ви були обрані для отримання призової винагороди у розмірі 20000 гривень! Щоб отримати приз, зателефонуйте за номером 09061701461. Код заявки KL341. Код дійсний лише 12 годин;
2) Вітаю вас з цим святом. Насолоджуйтесь і бажаю вам багато щасливих моментів, де б ви не були;

Перший текст є спамом, другий текст не є спамом.

Діаграма, що показує розподіл текстів (приблизно 4800 текстів не є спамом і 800 є спамом) в даному корпусі представлена на рис. 2.7.

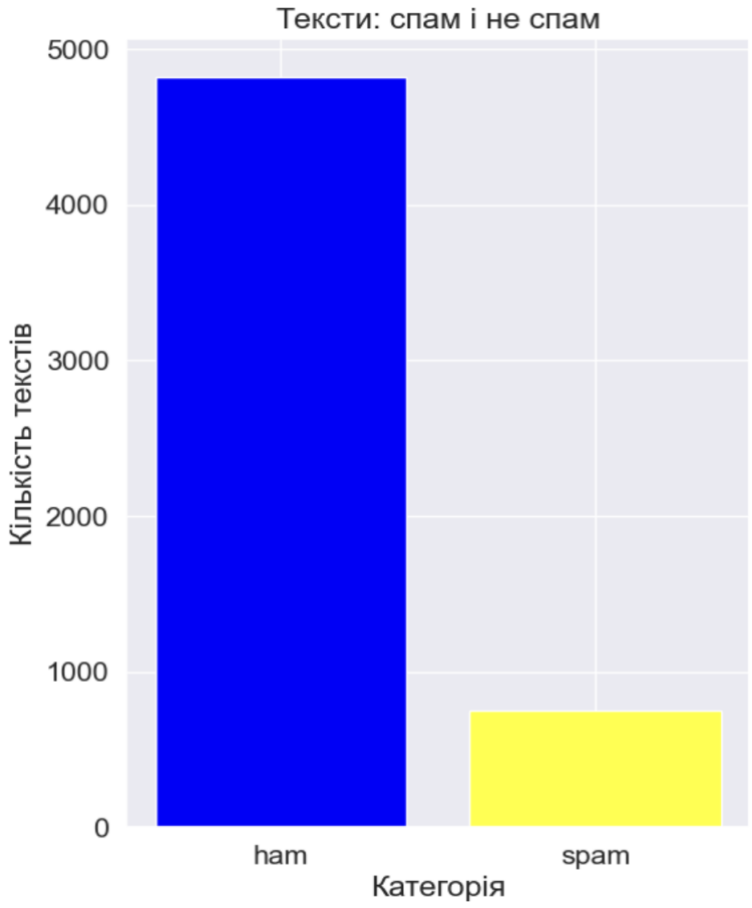


Рисунок 2.7. Спам і не спам тексти у вибірці

Для тренування модифікації нейронної мережі ДКЧП для детекції пропаганди будемо використовувати датасети HiQualProp (A Human-Annotated Dataset for Detecting Online Propaganda) [96] та матеріали із сайту EUvsDisinfo.eu, який веде “оперативна робоча група зі стратегічних комунікацій”, ОРГСК, підрозділ Європейської служби зовнішньополітичної діяльності, що створений у квітні 2015 р. на підставі ухваленого на зустрічі Європейського союзу 19-20 березня 2015р. рішення про необхідність «протистояти постійним дезінформаційним кампаніям з боку Росії» [97].

Набір HiQualProp створений збираючи різні пости в Твітері (X), щодо війни України з Росією. Промаркована, як бінарна класифікація, так і пропагандистські технології. Тобто для кожного тексту промарковано пропаганда, чи не пропаганда, а також, який метод пропаганди використовується.

Вказуються такі типи пропаганди: «гасла», «наклеювання ярликів», «апеляція до страху чи упередження», «навантажена мова», «сумнів», «махання прапором», «стадний інстинкт», «редукція порівняння з Гітлером», «чорно-біла помилка», «апеляція до авторитету», «думкозупиняючі кліше», «а як щодо».

Також промаркована класифікація за стратегією пропаганди: «проти України», «за проросійський уряд», «проти західних країн», «проти інших країн».

Декілька прикладів класифікації технології пропаганди:

1) Технологія, а як щодо? (контрзвинувачення з можливим переведенням теми): "Чудова ідея. Росія повинна забратися з України. Але як щодо існуючих наслідків, від яких зараз страждають етнічні росіяни? Як зупинити "зачистку" українським урядом етнічних росіян-українців? Як зупинити розміщення ядерних боєголовок НАТО на російських кордонах і наведення їх на росіян?"

2) Технологія, апелювання до авторитета (експертний сайт в даному випадку): "Україна хоче відключити Росію від основних інтернет-систем. Експерти кажуть, що це лише зашкодить російському народу і мало вплине на уряд країни. https://t.co/lFSymigNld."

3) Технологія, порівняння з еталоном негативного: "Колись Володимир Зеленський був популярним коміком. Зараз він балакає про всілякі дикі речі, в яких,

як він стверджує, винна Росія, так само, так сам, як робив Гітлер. Кокаїн - серйозна проблема в Києві."

4) Технологія, кліше (намагання завершити думку): "Росія оточена американськими базами. Розпустити НАТО. Покласти край нескінченній війні. Я знаю тільки ці дві речі, які мушу сказати."

5) Технологія, апеляція до страху: "Кремль попередив, що світ може опинитися на межі великої продовольчої кризи через незаконні обмеження, накладені на Росію західними країнами, і рішення, прийняті українською владою.

У кластері пропагандистських текстів (рис.2.8), понад 25000 регулярних текстів, та близько 5000 стандартних текстів.

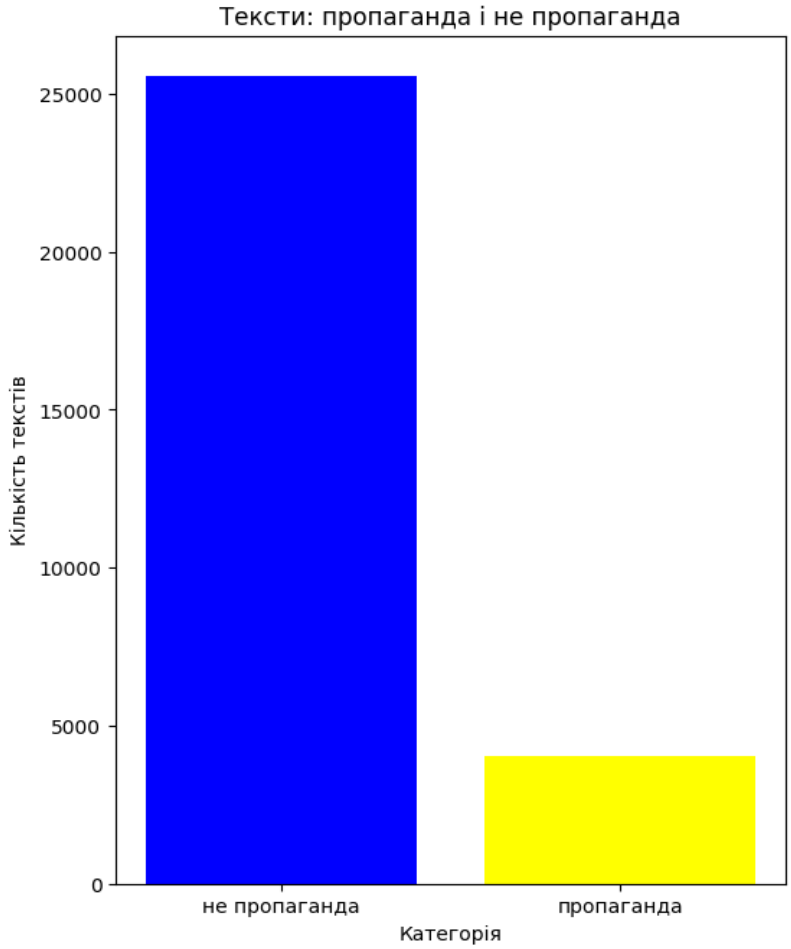


Рисунок 2.8. Тексти пропаганда і не пропаганда

Для того, щоб запропонувати алгоритм нейронної мережі, для початку потрібно зробити передобробку текстів для спаму та пропаганди (рис. 2.9).

Для обробки текстів будемо використовувати попередню обробку. Застосуємо стемінг (процес приведення відмінюваних слів до їхньої словотворчої основи), лематизацію (це процес групування відмінюваних форм слова, щоб їх можна було аналізувати, як єдине ціле), нормалізацію текстів.

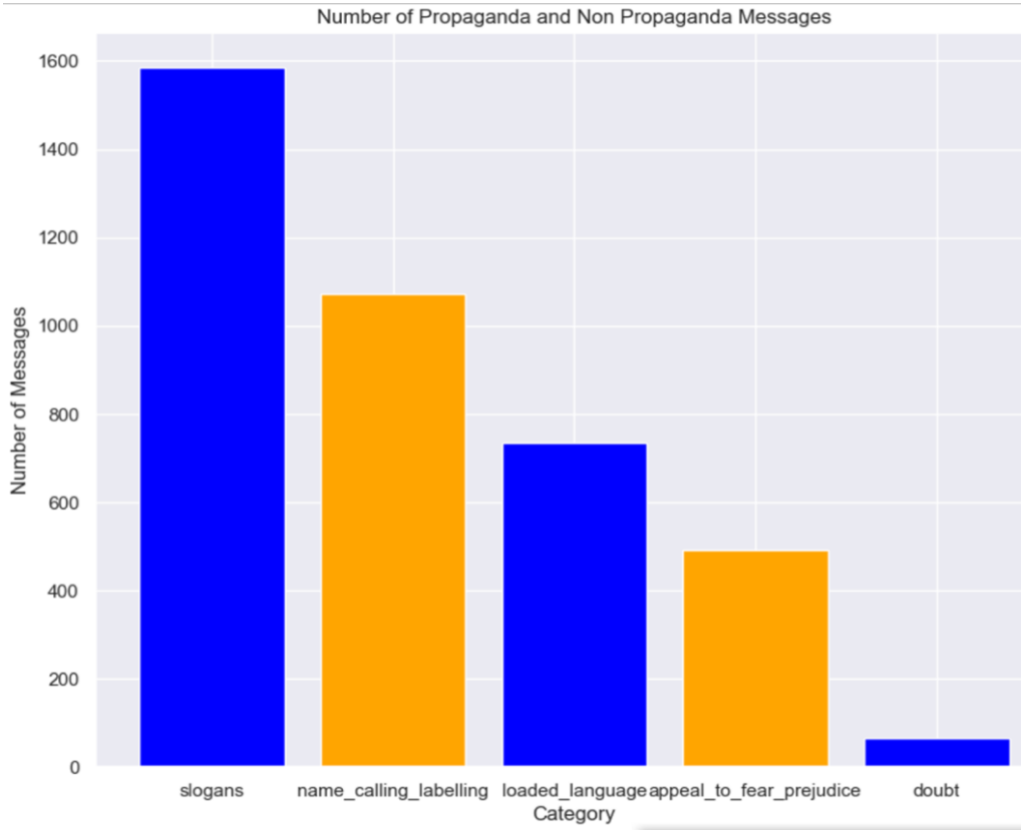


Рисунок 2.9. Діаграма кількості текстів для різних типів пропаганди

На рис. 2.10 запропонована модель НМКТуСН – нейронна мережа класифікації текстів у стрічці новин.

На вхід цієї нейронній мережі подаються наперед оброблені тексти у вигляді послідовнестей чисел і також бінарна класифікація до кожного тексту. Це може буьт одиничка, якщо текст спам і нуль, якщо текст не є спамом.

Далі, кожне число цієї послідовності за допомогою технології «word embedding» перетворюється на вектор чисел. Вектор чисел несе певну семантику слова, за яке відповідає число послідовності. Для перетворення слів та текстів у векторні представлення будемо використовувати наперед натреновану нейронну мережу з трансформер архітектурою «Distill Roberta».

Нейронна мережа складається з комбінації різних елементів – спочатку перетворення вектору в текст, потім перетворення за допомогою ЗНМ, подальшого використання двосторонньої ДКЧП (рис. 2.11), і функції активації сигмоїд (у випадку бінарної класифікації) наприкінці.

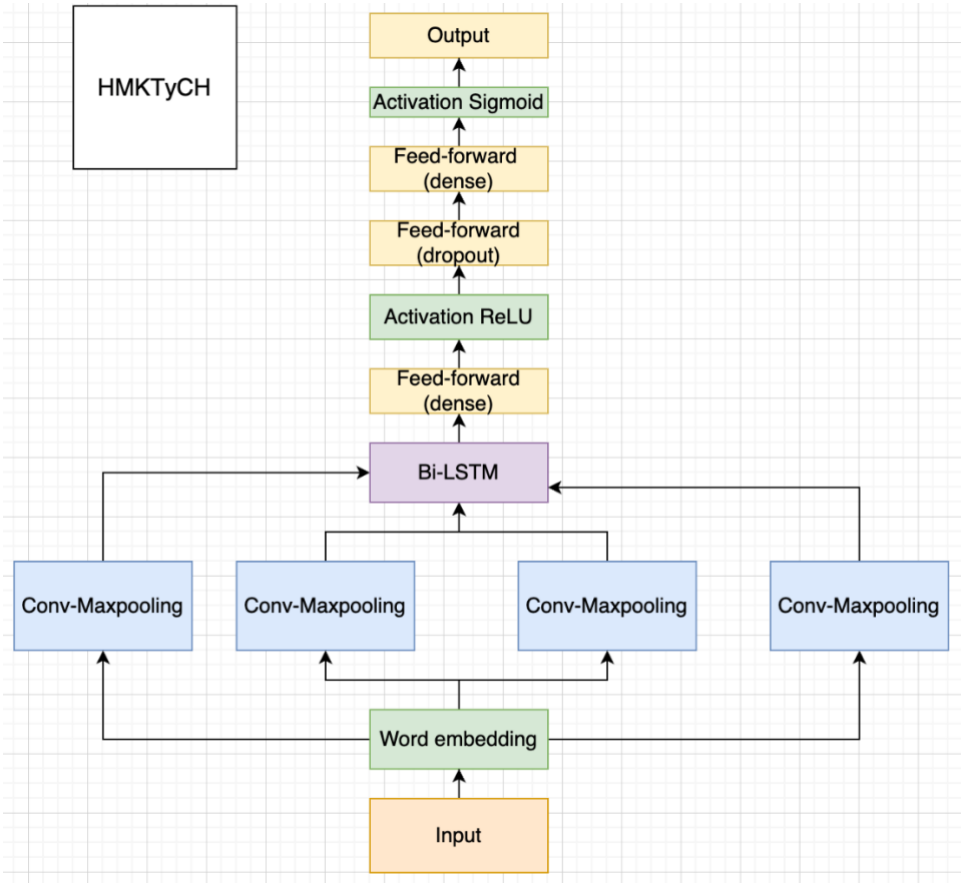


Рисунок 2.10. Модель НМКТуСН

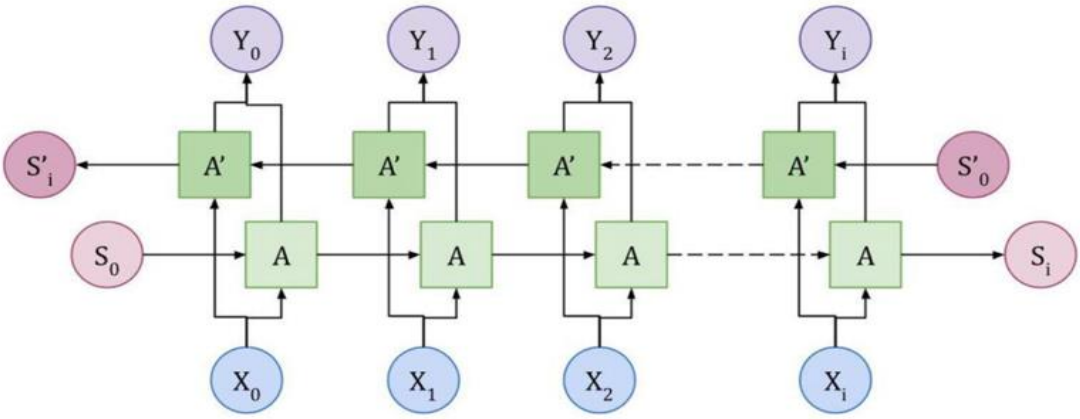


Рисунок. 2.11 Робота двосторонньої ДКЧП. Репринт з [98]

На вхід для НМКТуСН подається матриця розміром кількість текстів * максимальна кількість слів в тексті * розмір вектора вбудувань. Перетворення слів у вектори чисел виконується за допомогою класифікатора Distill Roberta, який кожне слово перетворює у нормалізований вектор із 768 дійсних чисел.

На рис. 2.12 показано конфігурацію для класифікатора текстів спаму, а саме максимальна кількість слів в текстах для класифікації текстів спаму – 50.

Layer (type)	Output Shape	Param #	Connected to
input_layer_10 (InputLayer)	(None, 50, 768)	0	-
Branch1 (Sequential)	(None, 12, 256)	590,080	input_layer_10[0…
└ conv1d_8 (Conv1D)	(None, 50, 256)	590,080	-
└ max_pooling1d_8 (MaxPooling1D)	(None, 12, 256)	0	-
Branch2 (Sequential)	(None, 12, 256)	983,296	input_layer_10[0…
└ conv1d_9 (Conv1D)	(None, 50, 256)	983,296	-
└ max_pooling1d_9 (MaxPooling1D)	(None, 12, 256)	0	-
Branch3 (Sequential)	(None, 12, 256)	1,376,512	input_layer_10[0…
└ conv1d_10 (Conv1D)	(None, 50, 256)	1,376,512	-
└ max_pooling1d_10 (MaxPooling1D)	(None, 12, 256)	0	-
Branch4 (Sequential)	(None, 12, 256)	1,966,336	input_layer_10[0…
└ conv1d_11 (Conv1D)	(None, 50, 256)	1,966,336	-
└ max_pooling1d_11 (MaxPooling1D)	(None, 12, 256)	0	-
ConcatenateLayers (Concatenate)	(None, 12, 1024)	0	Branch1[0][0], Branch2[0][0], Branch3[0][0], Branch4[0][0]
BiLSTMLayer (Bidirectional)	(None, 256)	1,180,672	ConcatenateLayer…
FullyConnectedLayer (Dense)	(None, 64)	16,448	BiLSTMLayer[0][0]
DropoutLayer (Dropout)	(None, 64)	0	FullyConnectedLa…
FullConnectedOutpu… (Dense)	(None, 1)	65	DropoutLayer[0][…

Рисунок 2.12 Детальна архітектура моделі НМКТуСН

Для вибору важливих ознак, та зменшення розмірності, використовується Conv-Maxpooling шари, на основі яких будується ЗНМ. Використовується Conv1d шари, з різними розмірами ядра – 3, 5, 7, 10. Потім відбувається «пулінг» MaxPooling1D, з максимізаційним агрегуванням і розміром шару 4. Кількість фільтрів, яка застосовується – 256, для всіх Conv1d шарів. Кожен із шарів згорткової нейронної мережі на виході для кожного тексту видає тензор розмірності (12 * 256).

Кількість параметрів для зготкової нейромережі обчислюється за формулою:

$$parameter\_cnt = (kernel_{size} * embedding_{size} * filters_{cnt}) + filters\_cnt \quad (2.10)$$

В даному випадку 3 * 768 * 256 + 256 = 590080.

Наступний етап на рис. 2.12 це конкатенація всіх шарів згорткової нейромежі. Таким чином, отримується тензор розмірності (12 * 1024) для кожного вхідного тексту. В наступному шарі застосовується двостороння модель ДКЧП.

Двостороння мережа ДКЧП обробляє текст, як у прямому так і в зворотньому напрямі. Це дозволяє моделі фіксувати контекст як з минулого, так і з майбутнього станів послідовності, забезпечуючи більш повне розуміння тексту. Це особливо корисно для складних синтаксичних структур, або коли значення слова сильно залежить від оточуючих слів. В даному випадку, на кожному кроці мережі ДКЧП буде подаватися вектор довжиною 1024 дійсних чисел.

Кількість параметрів для ДКПЧ обраховується за формулою:

$$\text{params} = 4 * ((\text{input dim} + \text{output dim} + 1) \times \text{output dim}) \quad (2.11)$$

Маємо: 4* ((1024+128+1))*128 = 590336. Враховучючи те, що мережа є двосторонньою, то кількість параметрів 590336 * 2 = 1180672.

Наступний шар це мережа прямого поширення з розмірністю вихідного шару 64, і функцією активації «ReLU». Кількість параметрів для даного шару обчислюється за формулою:

parameters_cnt = (input features * output features) + output features (2.12)

Для моделі на рис. 2.12 це (256 * 64) + 64 = 16448.

Далі йде шар відсіву. Основна ціль використання шару відсіву, це запобігання перенавчання нейронної мережі. Тобто, щоб не було такого випадку, коли нейронна мережа на тренувальних даних показує хороші результати, а на тестових погані.

І останній етап це проектування за допомогою мережі прямого поширення в одну комірку, яка прийматиме значення від нуля до одиниці, тому використовується функція активації сигмоїд. Тобто нейронна мережа видаватиме ймовірність того, що текст є наприклад спамом, чи не спамом. І, якщо ймовірність того, що текст є спамом більша, або рівна, 0.5 то, вважається, що текст є спамом. У випадку багатокласової класифікації, тексту, що є пропагандою, буде використовуватися «one hot encoding» та повнозв'язний шар мережі прямого поширення з фукнцією активації softmax.

Сумарно НМКТуСН має 6 мільйонів 113 тисяч 409 параметрів.

Для того, щоб обчислити результат Conv шару, використовується операція згортки.

В математичному аналізі операція згортки визначається таким чином:

$$(f * g)(t) = \int_{-\infty}^{+\infty} f(t-s)g(s)\,ds, \quad (2.13)$$

де f – це функція фільтр.

В дискретному випадку, з функцією f – обмеженею областю значень (область значень від 1 до кількості елементів ядра), ця формула перетворюється в:

$$(f * g)(t) = \sum_{s=1}^{s = kernel\ size} f\ (t-s) * g\ \ (s) \quad (2.14)$$

Після застосування різних фільтрів і операції згортки, на отриманих даних використовується функція активації.

Формула для максимізаційне агрегування:

$$\text{maxpooling}(f(t)) = \max_{s=0}^{s=\text{pooling size}-1} f(t+s) \quad (2.15)$$

Формула для поширення даних в одну сторону в двосторонній нейронній мережі ДКЧП:

$$s_t = \tanh(W_c [\Gamma_r * a_{t-1}, x_t] + b_c) \quad (2.16)$$

$$c_t = \Gamma_u * s_t + \Gamma_f * c_{t-1} \quad (2.17)$$

$$a_o = \Gamma_o * c_t \quad (2.18)$$

$$y_i = \text{sigm}(W_{ya} a_{i-1} + b_y) \quad (2.19)$$

x_i – значення, які подаються на кожному кроці (вбудовані вектори); W_c, W_{ya} – значення на початку виконання нейронної мережі ДКЧП ініціалізуються випадковими числами; tanh, sigm – функції активації; Γ_r – ворота релевантності, Γ_f – ворота забуття, Γ_u – ворота оновлення, Γ_o – вихідні ворота.

На основі методу модифікованого ДКЧП буде визначатися ймовірність того, чи пост є спамом чи пропагандою, та у випадку високої ймовірності чи є пост пропагандою, буде вказувати, якого типу пропаганди є даний пост. Тобто буде вказуватися, наприклад, що пост з ймовірністю 80 відсотків є пропагандою, і найімовірніше в даному пості використовується технологія «апелювання до страху».

На основі історії постів користувача, постів, які він лайкає та коментує, буде формуватися новий вектор характеристик. Цей вектор буде подаватися на вхід нейронній мережі прямого поширення похибки.

Мережа прямого поширення похибки буде будувати результати на основі результатів НМКТуСН, а також на основі історії користувача. Інтуїція для цього вдосконалення, є такою, що користувач, який поширював пропаганду певного типу, буде робити це знову. Нейронна мережа буде мати інформацію про це, помітить дані зв’язки і точніше оцінить новий пост.

2.4 Застосування методів машинного навчання для кластеризації текстів у соціальних мережах

2.4.1 К-середніх

Алгоритм кластеризації К-середніх є наріжним каменем у сфері інтелектуального аналізу даних та машинного навчання, відомий своєю простотою та ефективністю у поділі набору даних на окремі підмножини, що не перетинаються, або кластери. Суть цього методу полягає в його здатності мінімізувати дисперсію всередині кожного кластера, тим самим гарантуючи, що елементи в кожній групі максимально схожі один на одного, одночасно максимізуючи відмінності між різними кластерами [99].

Алгоритм К-середніх складається з низки послідовних кроків:

1) Ініціалізація: Процес починається з вибору k початкових центроїдів, що може бути зроблено випадковим чином, або за допомогою евристичного підходу;
2) Присвоєння: Згодом кожній точці даних присвоюється найближчий до неї центроїд, в результаті чого формується k кластерів;
3) Оновлення: Центроїди перераховуються як середнє значення всіх точок у кожному кластері;
4) Ітерація: Ці кроки призначення та оновлення повторюються ітераційно, поки центроїди не стабілізуються з мінімальними змінами, що свідчить про збіжність алгоритму.

Метою алгоритму є мінімізація суми квадратів відстаней між точками даних і відповідними центроїдами кластерів, що по суті зменшує суму квадратів всередині кластера (внутрішньокластерна сума квадратів, англ. WCSS) і збільшує суму квадратів між кластерами (BCSS). Така оптимізація забезпечує формування компактних, добре диференційованих кластерів [100].

Кластеризація К-середніх використовується в широкому спектрі галузей, що підкреслює її адаптивність та ефективність у категоризації немаркованих даних:

1) Сегментація ринку: Дозволяє компаніям розподіляти клієнтів на кластери на основі різних атрибутів, таких як купівельні звички, демографічні показники

чи вподобання, що сприяє розробці більш цілеспрямованих маркетингових стратегій.

2) Сегментація та стиснення зображень: У сфері комп'ютерного зору K-середніх допомагають сегментувати зображення на значущі кластери або стискати зображення, зменшуючи палітру до "k" центроїдів.
3) Кластеризація документів: Групування документів зі схожим тематичним змістом, тим самим спрощуючи пошук і організацію інформації у великих текстових масивах даних.
4) Виявлення аномалій: K-середніх допомагають виявляти аномальні дані, створюючи кластери нормальної поведінки, що є цінним у таких контекстах, як виявлення шахрайства або мережева безпека.

Незважаючи на широке розповсюдження, K-середніх не позбавлені проблем, таких як чутливість до початкового вибору центроїда, проблеми з кластеризацією даних, що мають несферичний розподіл або різний розмір кластерів, а також необхідність заздалегідь визначати кількість кластерів k.

Для усунення цих недоліків було запропоновано вдосконалення та альтернативні методи ініціалізації, що підвищують стійкість та ефективність збіжності алгоритму.

Застосування кластеризації K-середніх полегшує категоризацію учасників соціальних мереж відповідно до їхньої активності. Цей процес категоризації вміло розмежовує користувачів з вузькоспеціалізованим фокусом ("сфокусовані користувачі") і тих, хто демонструє ширший спектр інтересів ("різноманітні користувачі"). Варто зазначити, що різкі та концентровані зміни в тематичній активності користувача можуть свідчити про маніпулятивні спроби. Інтеграція латентного розподілу Діріхле (LDA) для розширеного моделювання тем ще більше вдосконалює цей аналітичний процес, потенційно проливаючи світло на організовані зусилля, спрямовані на просування певних наративів [101].

За допомогою кластеризації за методом K-середніх можна виявити користувачів, які характеризуються рідкісними або аномальними соціальними

зв'язками, що, ймовірно, свідчить про сфабриковані акаунти, призначені для поширення пропагандистських повідомлень [102].

Кластеризація за методом K-середніх допомагає виявити профілі, які демонструють поведінку, що різко відрізняється від загальноприйнятих шаблонів. Такі аномалії, незалежно від того, чи проявляються вони у вигляді надмірно частих постів або нетрадиційних мережевих утворень, можуть свідчити про роботу автоматизованих ботів або організацію дезінформаційних кампаній.

Інтеграція аналізу настроїв з кластеризацією за методом K-середніх полегшує дослідження емоційного підтексту, що пронизує дописи та діалоги. Кластеризація користувачів або контенту на основі вираження настроїв дає змогу виявити тенденції, що вказують на намагання спровокувати інтенсивну емоційну реакцію, або вплинути на суспільні настрої.

Кластеризація K-середніх може класифікувати користувачів відповідно до даних про взаємодію (наприклад, лайки, пости, коментарі). Це полегшує виявлення тісно пов'язаних спільнот користувачів, які демонструють тенденції соціальних бульбашок. Такі тенденції характеризуються переважною взаємодією всередині групи та обміном контентом без значного впливу протилежних точок зору.

Кластеризація за методом K-середніх дозволяє аналізувати структури соціальних мереж, виявляючи групи користувачів, які демонструють вищий рівень внутрішнього зв'язку, ніж зв'язку з ширшою мережею. Ці щільні кластери можуть означати соціальні бульбашки, особливо коли внутрішньокластерні моделі комунікації виявляють однорідність і поляризацію контенту.

Виявивши соціальні бульбашки, кластеризація за методом K-середніх може підтримати зусилля, спрямовані на диверсифікацію контенту. Розуміння ознак кластерів дозволяє алгоритмам соціальних мереж представляти альтернативні точки зору в стрічках користувачів. Це може призвести до точного визначення кластерів з протилежними точками зору та перехресного просування обраного контенту. Платформи можуть використовувати тих людей, які мають зв'язок з кількома кластерами, або які демонструють більш різноманітні звички споживання контенту [103].

Висновки, отримані в результаті кластеризації за методом K-середніх, можуть вдосконалити алгоритми персоналізації. Визначення домінуючої соціальної бульбашки користувача та його сприйнятливості до різноманітності дозволяє створювати рекомендації, які забезпечують баланс між залученням і поступовим ознайомленням з ширшим спектром точок зору.

2.4.2 DBScan

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) – це популярний алгоритм машинного навчання, що використовується для кластерного аналізу.

Основна ідея алгоритму DBSCAN це групування точок, на основі їхньої щільності, розрізняючи кластери за щільними областями, які перемежовуються областями з меншою щільністю. Алгоритм приймає на вхід два основних параметри:

eps (ε): Цей параметр являє собою максимальну відстань між двома точками, які вважаються сусідами.

minPts: Позначає мінімальну кількість точок, необхідну для утворення щільної області.

Алгоритм відносить кожну точку до одного з трьох типів:

- *основні точки*: Це точки, які мають принаймні minPts кількість точок у радіусі eps.
- *граничні точки*: Ці точки не є основними, але є сусідами принаймні однієї основної точки.
- *точки шум, або точки винятки*: Ці точки не є основними, або граничними точками.

DBSCAN починає пошук з довільної невідвіданої точки і знаходить всі точки, досяжні з цієї точки, на основі eps і minPts. Якщо точка є опорною, створюється новий кластер. Потім DBSCAN ітеративно збирає точки досяжні з даної і додає їх до кластера. Цей процес продовжується до тих пір, поки кластер не буде повністю

знайдено. Потім обробляється нова невідвідана точка, щоб знайти наступний кластер.

2.5 Створення методу диверсифікації новин на основі кластеризації текстів з дискусійних тем, для зменшення ефекту «бульбашки»

Як зясовано в першому розділі, одна з основних причин виникнення соціальних бульбашок це те, що користувачі не бачать думки користувачів з інших бульбашок. Або, якщо їм таки і попадаються новини з інших соціальних бульбашок в своїй стрічці, то ця позиція сильно відрізняється від позиції даного користувача по темі.

Основна ідея методу (рис.2.13), що пропонується користувачам соціальних мереж новини з інших кластерів, які висвітлюють думку близьку до думки користувача даного кластеру.

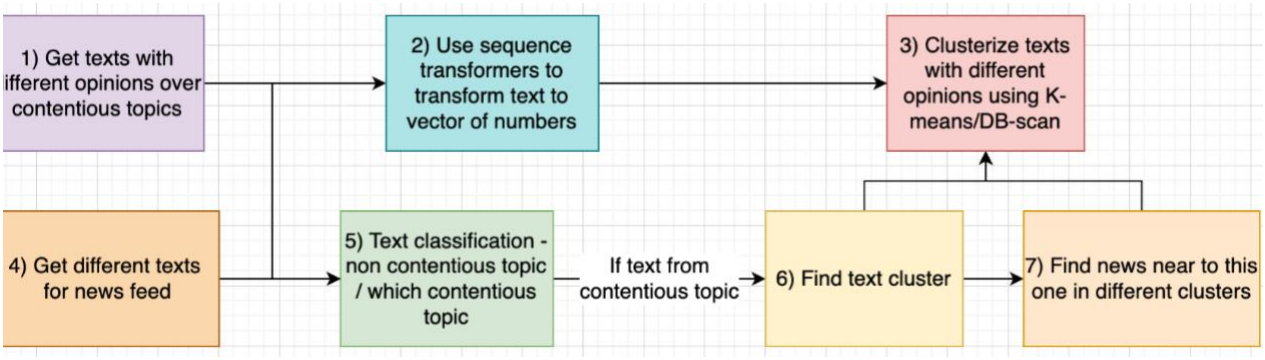


Рисунок 2.13 Кроки необхідні для пошуку «близькі» новини з інших кластерів

Як і у випадку з класифікацією текстів, буде використана попередня обробка текстів, яка перетворює тексти в вектори чисел. Для цього буде використана нейронна мережа з трасформер архітектурою.

Перший крок, це отримання вибірки з дискусійної теми. Для цього взято корпус даних «UKP Sentential Argument Mining Corpus» [104], що включає 25492 речення з восьми суперечливих тем. Кожне речення, було анотоване, як аргумент за, аргумент проти, або не стосується даної теми. Вісім суперечливих тем – аборти,

клонування, смертна кара, контроль над володінням зброєю, мінімальна заробітна плата, ядерна енергія, шкільні форми.

Інший корпус даних, який використовується в даній роботі використовує дані з сайту args.me, який базується на статті «Building an Argument Search Engine for the Web» [105]. Тексти щодо дискусійних тем, також додані з цього вебсайту до загальної вибірки.

Другий крок – ці корпуси текстів трансформовані у числові вектори за допомогою трансформеру послідовностей paraphrase-distilroberta-base-v1. Таким чином це дозволяє перевіряти подібності між текстами на основі значення скалярного добутку нормалізованих векторів.

Треій крок це використання кластеризація на основі K-середніх для того, щоб розд9ілити дані вектори на різні кластери.

В результаті виконання цього кроку для дискусійної теми буде отримано кластери (рис. 2.14) з різними думками, щодо даної теми.

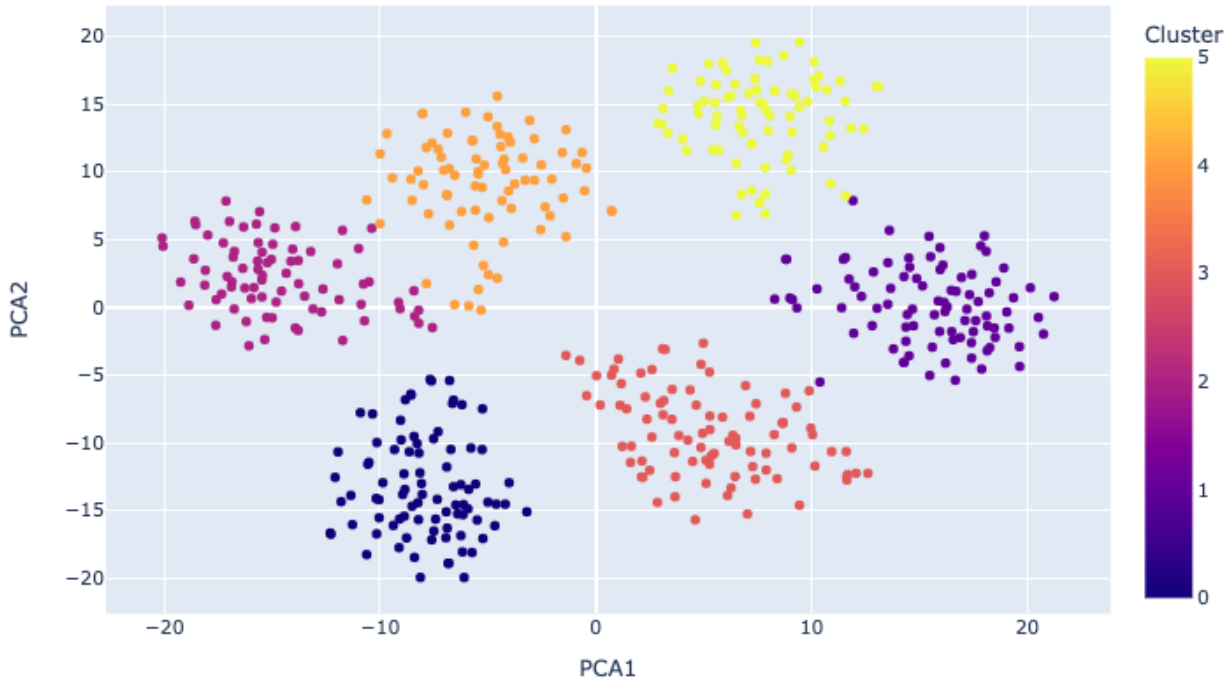


Рисунок 2.14. Результат використання алгоритму K-середніх для кластеризації текстів, в яких користувачі вказують свою думку, щодо носіння шкільної форми.

Метод K-середніх [106] використовується для кластеризації даних за спектром поглядів на різні теми. Для визначення оптимальної кількості кластерів (думок) використовується метод «ліктя» [107] для методу K-середніх.

Четвертий і п’ятий кроки необхідні для того, щоб натренувати модель НМКТуСН, щоб вона вміла класифікувати тексти на дискусійні і не дискусійні, і у випадку, якщо текст належить до дискусійної категорії, визначала, яка саме дискусійна категорія відповідає даному тексту. В четвертому кроці використовується корпус даних MN-DS: A Multilabeled News Dataset for News Articles Hierarchical Classification [108] із текстами дискусійних тем, описаних на першому кроці.

Після закінчення п’ятого кроку, стає відомо чи є текст на дискусійну тематику, і якщо текст на дискусійну тематику, то яка саме його тема рис. 2.15.

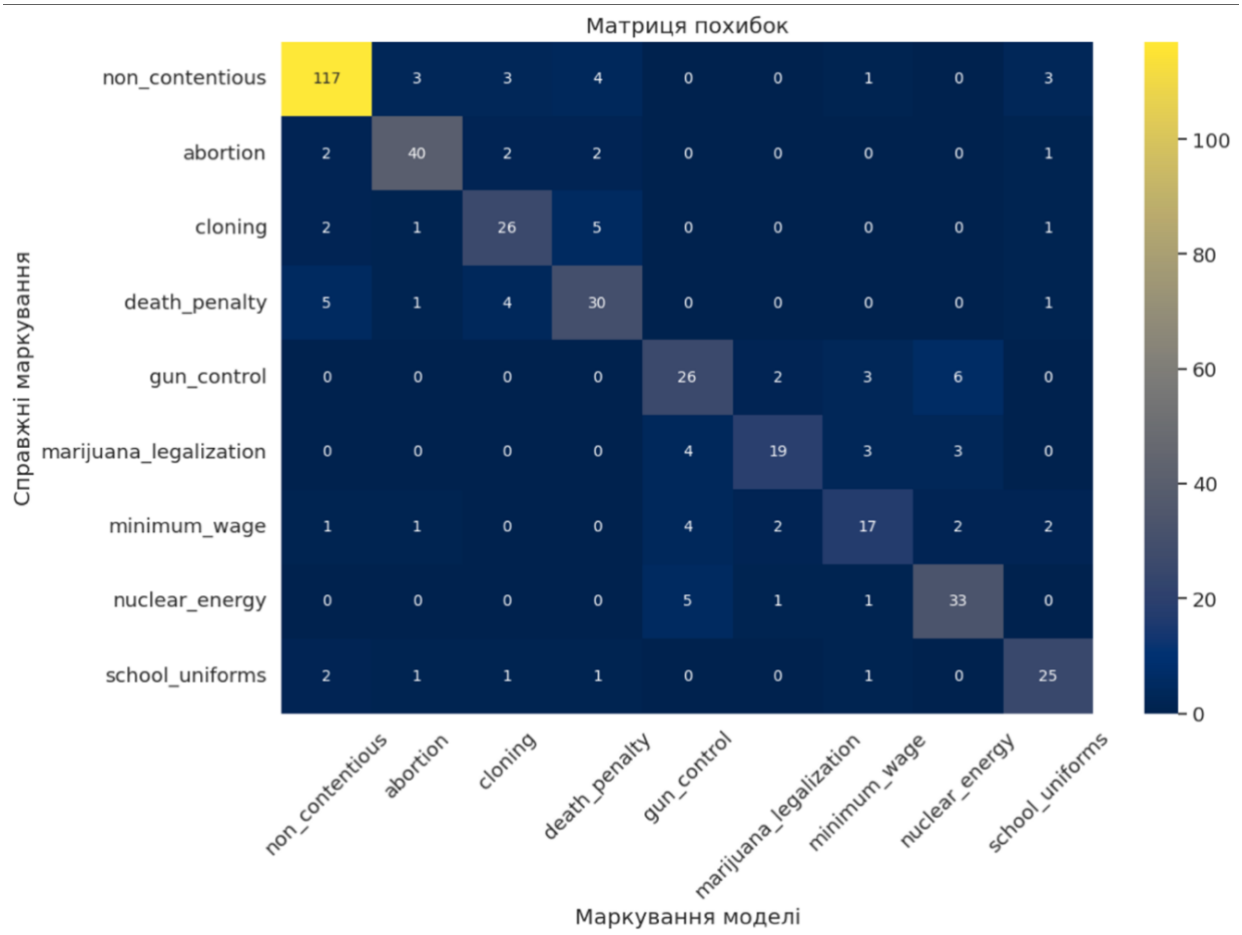


Рисунок 2.15 Класифікація тексту на дискусійні категорії за допомогою мережі НМКТуСН

На рис 2.16 показано модифікації НМКТуСН, яка використовується. Останній шар, складається з 9 нейронів, використовується функція активації softmax, для визначення ймовірності приналежності тексту до кластеру.

Layer (type)	Output Shape	Param #	Connected to
input_layer_15 (InputLayer)	(None, 80, 768)	0	–
Branch1 (Sequential)	(None, 20, 256)	590,080	input_layer_15[0…
Branch2 (Sequential)	(None, 20, 256)	983,296	input_layer_15[0…
Branch3 (Sequential)	(None, 20, 256)	1,376,512	input_layer_15[0…
Branch4 (Sequential)	(None, 20, 256)	1,966,336	input_layer_15[0…
ConcatenateLayers (Concatenate)	(None, 20, 1024)	0	Branch1[0][0], Branch2[0][0], Branch3[0][0], Branch4[0][0]
BiLSTMLayer (Bidirectional)	(None, 256)	1,180,672	ConcatenateLayer…
FullyConnectedLayer (Dense)	(None, 64)	16,448	BiLSTMLayer[0][0]
DropoutLayer (Dropout)	(None, 64)	0	FullyConnectedLa…
FullConnectedOutpu… (Dense)	(None, 9)	585	DropoutLayer[0][…

Рисунок 2.16 Модифікація НМКТуСН для класифікації на дискусійні теми.

На рис. 2.15 non_contentious, значить, що модель класифікує даний текст, як недискусійний, в такому випадку до 6-го кроку перехід зроблено не буде. В інакшому випадку, якщо текст, було класифіковано, як один з: abortion, cloning, death_penalty, gun_control, marijuana_legislation, minimum_wage, nuclear_energy, school_uniforms, для даного тексту буде знайдено відповідний йому кластер серед кластерів дискусійної категорії.

На 6-му кроці, знаходиться кластер, який відповідає даному текстові. І на 7-му кроці знаходиться тексти з інших кластерів, які є на невеликій відстані від даного тексту. Відстань вимірюється скалярним добутком нормалізованих векторів.

Таким чином, управління знаннями відбувається наступним чином: маючи якийсь тест на дискусійну тематику, описаний вище метод, дозволяє шукати близькі тексти з інших кластерів. В соціальній мережі слідкуючи за темами і думками, які подобаються користувачеві, можна оцінити, яку думку має користувач, щодо даної теми. Це можна зробити агрегуванням по вбудованим векторам текстів даної теми, які сподобалися даному користувачеві. Таким чином маючи множину новин, яку можна запропонувати користувачеві, даний метод дає змогу вибирати такі новини, які знаходяться в інших кластерах по дискусійних темах.

Для того, щоб генерувати стрічку новин у соціальній мережі, необхідно зібрати різні типи даних, як-от: публікації та коментарі, які раніше подобалися користувачеві; зв’язок з іншими користувачами, тобто підписки; кількість лайків, репостів, коментарів під іншим профілем користувача; теми, які цікаві поточному користувачеві.

Для цього окрім таблиць в базі даних, які необхідні для функціонування системи соціальної мережі потрібні додаткові.

Також потрібно додати необхідні поля в існуючих таблицях (табл. 2.3 – 2.8).

Таблиця 2.5

Пости

posts	
post_id	*bigint*
user_id	*bigint*
hashtag	*str*
data_link	*str*
likes	*bigint*
shares	*bigint*

Для того, щоб ефективно генерувати стрічку новин необхідно зберігати кількість вподобайок та репостів постів.

Таблиця 2.6.

Коментарі

comments	
comment_id	*bigint*
post_id	*bigint*
user_id	*bigint*
likes	*bigint*
created_at	*date*
updated_at	*date*
deleted_at	*date*
data_link	*str*

Ці самі дані потрібно зберігати і для коментарів.

Таблиця 2.7.

Історія коментарів користувача

user_comment_history	
user_id	*bigint*
comment_id	*bigint*
comment_data_link	*str*

Також у окремій таблиці потрібно зберігати історії коментарів користувача, та статистику вподбайок, поширень та коментарів під постом іншого користувача.

Ці дані, а також інформація про приналежність до кластерів дозволяють ефективно генерувати диверсифіковану стрічку новин на основі нейронних мереж.

Таблиця 2.8.

Статистика відносин користувача

relationship	
following_id	*bigint*
followed_id	*bigint*
created_at	*date*
avg_likes	*float*
overall_likes	*bigint*
avg_shares	*float*
overall_shares	*bigint*
avg_comments	*float*
overall_comments	*bigint*

Висновки до розділу 2

1. В розділі створено модель нейронна мережа для класифікації тексту у стрічці новин (НМКТуСН) на основі модифікації гібриду згорткової нейронної мережі та мережі довгої короткочасної пам’яті. Нейронна мережа також використовує наперед натреновані векторне представлення слів. На основі НМКТуСН розроблено метод класифікації текстів на спам та пропаганду. Також, у випадку, якщо текст є спамом, то класифікатор додатково класифікує, яка саме пропагандистька технологія використовується у текстів.

2. Використання різних способів попередньої обробки текстів дозволяє використовувати різні методи машинного навчання і моделі нейроних мереж для порівняння результатів класифікації з результатами класифікації за допомогою НМКТуСН.

3. Запропоновано метод пом’якшення впливу ефекту «бульбашки» через диверсифікацію стрічки новин на основі класифікації текстів за допомогою моделі НМКТуСН, подальшої кластеризації текстів на основі комбінації алгоритму К-середніх та перетворення текстів у вектори чисел за допомогою нейронної мережі з трасформер архітектурою.

РОЗДІЛ 3. СТВОРЕННЯ МОДЕЛІ ВИСОКОНАВАНТАЖЕНОЇ РОЗПОДІЛЕНОЇ СИСТЕМИ СОЦІАЛЬНОЇ МЕРЕЖІ

У третьому розділі описано необхідні операції користувачів у соціальних мережах; удосконалено масштабовану модель для соціальної мережі; описано напрямки розширення даної моделі, при збільшенні кількості функціоналу; визначено вразливі місця запропонованої моделі; під час моделювання даної системи особлива увага приділяється швидкодії даної системи, та їй пропускній здатності; інтегровано модель високонавантаженої системи соціальної мережі з методами класифікації постів на пропаганду та спам; інтегровано модель з методом диверсифікації стрічки новин для зменшення ефекту «бульбашки» в соціальних мережах; покращено систему в напрямку збільшення швидкодії за рахунок використання мережі доставки повідомлень та кешування, добавлено моніторинг швидкості запитів та наявність помилок.

3.1 Основні операції користувачів у соціальній мережі.

Створення архітектури стрічки новин у соціальних мережах пов’язана з багатьма проблемами. Перш за все, система має бути масштабованою, стрибки кількості нових користувачів, нових дописів та коментарів повинні добре оброблятися. По-друге генерація стрічки новин повинна відбуватися в режимі реального часу з низькою затримкою, щоб користувачі не чекали надто довго (більше 200 мілісекунд) свою стрічку новин. По-третє, для кожного користувача, особливо з великою кількістю зв’язків, необхідно розставляти пріоритети новинам з його зв’язків.

Основні дії, які користувач може виконувати в системі соціальної мережі:

1) Зареєструватися та авторизуватися в системі, створити обліковий запис, надати основні ідентифікаційних дані. Такі елементи, як фотографії профілю, короткі біографії та поля інтересів, служать для встановлення онлайн-персони, яка сигналізує іншим, ким є користувач і ким хоче, щоб його бачили.

2) Створити пост. Соціальні мережі процвітають завдяки вмісту, створеному їхніми користувачами. У людей є різноманітні інструменти для обміну думками, досвідом і творчістю. Текстові дописи, фотографії, відео та посилання плавно наповнюють сторінки користувачів. Цей вміст служить мовою соціальної мережі, стимулюючи як індивідуальне самовираження, так і спілкування.

3) Підписуватися та відписуватися від інших користувачів. Цей зв’язок не є двостороннім. Тобто список людей на кого підписаний користувач і хто підписаний на даного користувача є різним.

4) Писати коментарі під постами інших користувачів. Здатність реагувати на контент є основою роботи в соціальній мережі. «Подобається» залишається широко поширеним означником інтересу, оскільки багато платформ розширюють спектр емоційних реакцій для більш насичених відповідей. Коментарі пропонують простір для обговорення, тоді як функція «поділитися» дозволяє користувачам виділяти вміст, який вони вважають переконливим, сприяючи його поширенню в своїх колах.

5) Гортати свою сторінку з персоналізованими новинами – стрічка новин створюється на основі публікацій інших користувачів, за якими стежить поточний користувач.

3.2 Вибір технологій для архітектури соціальної мережі

3.2.1 Бази даних

В цей час у високонавантажених розподілених системах використовуються різні типи нереляційних баз даних, включаючи бази даних з широкими стовпчиками, бази даних "ключ-значення" та документно-орієнтовані бази даних [109].

Бази даних з широкою колонкою мають схожість з реляційними базами даних у використанні стовпців, рядків і таблиць. Однак ці бази даних відрізняються тим, що стовпці можуть відрізнятися за назвою та кількістю в різних рядках. Вони

зазвичай використовуються для бізнес-аналітики та аналізу процесів, використовуючи паралельні обчислення без спільної пам'яті. Яскравими прикладами є Касандра (Cassandra) та АчБейз (HBase).

Бази даних ключ-значення, натхненні Dynamo від Amazon, призначені для платформ з численними сервісами, де доступ до даних здійснюється переважно за допомогою ключа. Неефективність реляційних баз даних щодо масштабованості та доступності робить бази даних на основі ключів більш підходящим варіантом [110]. Інші приклади баз даних типу "ключ-значення" включають Redis та Memcached.

Документно-орієнтовані бази даних представляють собою більш просунуту форму сховищ ключ-значення, здатну обробляти пари ключ-документ [111]. Ці бази даних стають все більш популярними завдяки своїй гнучкій структурі, яка усуває необхідність у жорстких процесах міграції даних. Прикладами таких баз даних є Монго (MongoDB) та Коуч (CouchDB). Ці бази даних особливо корисні при розробці прототипів та початкових версій продукту, оскільки розробнику не потрібно вказувати чітку структуру таблиць в базі.

База даних Касандра, спочатку розроблена Facebook у 2008 році, є системою розподілених баз даних, яка інтегрує ключові функції як бази даних Dynamo від Amazon, так і моделі Bigtable від Google.

Модель даних схожа на сховища ключів і значень. За своєю суттю Касандра функціонує подібно до баз даних типу "ключ-значення", але відрізняється тим, що зберігає значення у вигляді широкої колонки. Структура цих стовпців є гнучкою і може бути визначена динамічно під час виконання, що дозволяє створювати схему, яка може розвиватися відповідно до вимог програми.

Децентралізована архітектура. Децентралізований дизайн бази даних Касандра підвищує її масштабованість, доступність та надійність. Ця архітектура робить її особливо придатною для середовищ з високим навантаженням на запис і великими потоками даних, оскільки вона може ефективно розподіляти дані між декількома серверами [112].

Сегментація даних за допомогою віртуальних вузлів і стійке хешування. База даних Касандра використовує алгоритм стійкого хешування, зокрема хеш-функцію

Murmur3, для рівномірного розподілу даних по кластеру. Ключі даних, хешовані за допомогою Murmur3, призначаються віртуальним вузлам, які є логічними сегментами кластера. Кожен фізичний вузол у кластері керує кількома віртуальними вузлами, що спрощує такі операції, як масштабування. Зміни в кластері, такі як додавання або видалення серверів, вимагають лише перепризначення віртуальних вузлів, таким чином підтримуючи баланс навантаження без значного переміщення даних [113].

Реплікація всередині центрів обробки даних. Сервери логічно розташовані в кільце в межах кожного центру обробки даних, щоб полегшити реплікацію даних. Заздалегідь визначений коефіцієнт реплікації диктує, скільки копій кожного фрагмента даних створюється, забезпечуючи надмірність і відмовостійкість. Алгоритм пліток. Цей одноранговий протокол зв'язку використовується для ефективного поширення інформації про стан кожного вузла в кластері. Він допомагає підтримувати узгоджене уявлення про топологію кластера, полегшуючи процес реплікації та загальний моніторинг стану системи.

Динамічне балансування навантаження. Використання віртуальних вузлів і послідовного хешування також сприяє динамічному балансуванню навантаження між серверами. При зміні робочого навантаження або конфігурації кластера база даних Касандра може адаптивно перерозподіляти дані і балансувати навантаження без значного ручного втручання.

Така комбінація функцій робить базу даних Касандра ефективним вибором для додатків, які потребують високої доступності, швидкого масштабування та надійного розподілу даних.

Монго - це нереляційна база даних з відкритим вихідним кодом, яка була розроблена за допомогою C++ у 2009 році. Вона орієнтована на документи, тобто зберігає дані у гнучких документах у форматі JSON, де пов'язана інформація зберігається разом для полегшення доступу до неї за допомогою запитів. Ці документи організовані в колекції, які є групами документів Монго, які можна розглядати як еквівалент таблиць в реляційній базі даних.

Однією з ключових особливостей Монго є підтримка паралельних операцій, що робить її дуже придатною для додатків, які вимагають високої доступності. Відмовостійкість і висока доступність даних досягається завдяки механізму асинхронної реплікації "господар - слуга". Це дозволяє поширювати оновлення між репліками із затримкою - за замовчуванням дані записуються на диск кожні 60 секунд, але цей інтервал можна налаштувати [114].

Управління пам'яттю в Монго оптимізовано для ефективності. Коли створюються нові файли, Монго перезаписує старі ділянки пам'яті на диску, звільняючи місце і підтримуючи продуктивність. Однак Монго накладає обмеження на обсяг пам'яті для кожного документа; наприклад, максимальний розмір документа BSON становить 16 мегабайт.

Щоб полегшити швидкий пошук даних, MongoDB використовує індекси, які функціонують подібно до індексів у реляційних базах даних. Кожен документ повинен мати унікальний ідентифікатор _id, який MongoDB використовує для створення первинного індексу. Адміністратори також можуть визначати вторинні індекси для інших полів, щоб покращити продуктивність операцій запитів. Ці індекси, як правило, структуровані у вигляді B-дерева, яке є більш ефективним для операцій читання, ніж для операцій запису. Це особливо корисно, оскільки в реплікованій структурі MongoDB первинний вузол обробляє як читання, так і запис, в той час як вторинні вузли (репліки), як правило, доступні лише для читання.

Загалом, архітектура та можливості MongoDB роблять її чудовим вибором для розробників, яким потрібна масштабована, високопродуктивна система баз даних, що вміщує складні структури даних та забезпечує швидкий пошук даних.

Редіс - це база даних по типу ключ-значення, яка працює за архітектурою клієнт-сервер. Ця архітектура включає такі компоненти, як сервер Редіс, його репліки та клієнти Редіс. На відміну від багатьох баз даних, Редіс не покладається на методи операційної системи для керування віртуальною пам'яттю, а використовує оперативну пам'ять напряму. Для підтримки ефективної роботи Редіс розробив власну систему вивантаження рідко використовуваних даних з

оперативної пам'яті на диск. Такий підхід до віртуальної пам'яті дозволяє Редіс обробляти набори даних, більші за обсяг доступної оперативної пам'яті, без значного погіршення продуктивності [115].

Щодо теореми Брюера, то:

− бази даних Редіс і Монго задовольняють умови стійкості та узгодженості; Редіс досягає узгодженості завдяки своїй однопотоковій природі, що забезпечує послідовну обробку команд.

− база даних Касандра задовольняє умови стійкості та доступності.

База даних Монго підтримує узгодженість за допомогою архітектури «господар-слуга», де господар координує операції запису даних. Він підтверджує операції запису тільки після того, як дані будуть зафіксовані на всіх серверах реплік.

База даних Касандра, з іншого боку, узгоджується з доступністю та толерантністю до розділів. Її узгодженість є евентуальною, тобто вона досягає узгодженості з часом, коли дані поширюються на всі сервери, що обробляють запити на читання. Крім того, база даних Касандра дозволяє налаштовувати рівні узгодженості, що підвищує її гнучкість в управлінні доступністю.

Що стосується доступності, то база даних Касандра вирізняється своєю надійністю завдяки децентралізованій структурі. У разі виходу з ладу сервера інші вузли тимчасово зберігають дані, які синхронізуються з відновлюваним сервером після його повернення. Це контрастує з Редіс та Монго, які страждають від обмеженої доступності; у випадку Монго відповідь системи зупиняється, доки резервний сервер не візьме на себе роль головного сервера, що вийшов з ладу.

З точки зору відмовостійкості, всі досліджені бази даних демонструють високу стійкість до численних відмов серверів, забезпечуючи надійне управління даними в розподілених системах.

Таблиця 3.1.

Порівняння нереляційних баз даних

База даних	Касандра	Монго	Редіс
Мова програмування	Java	C++	C
Тип бази даних	З широким стовпчиком	Документ-орієнтована	Ключ-значення
Найкраще використовувати для:	Зберігання великої кількості даних, маючи зручний інтерфейс	Якщо потрібні робити динамічні запити	Для даних, які швидко змінюються, при обмеженні максимальної кількості даних
Наявність транзакцій	Немає	Немає	Наявні
Реплікація	Децентралізована	По типу «господар-слуга»	По типу «господар-слуга»
Площина	Доступність-стійкість	Стійкість-узгодженість	Стійкість-узгодженість

Проведений аналіз нереляційних баз даних, які застосовуються у високонавантажених розподілених системах, надає підстави зробити такі висновки щодо доцільності застосування баз даних архітектури Касандра, Редіс і Монго залежно від особливостей системи:

1) Редіс, відомий своєю швидкістю завдяки зберіганню даних у пам'яті, найкраще використовувати у сценаріях, що вимагають швидкого доступу до даних. Однак масштабування Редіс передбачає значне ручне втручання, тому для оптимізації продуктивності та масштабованості рекомендується використовувати його разом з Монго, Касандра або іншими базами даних.

2) Монго ідеально підходить для ситуацій, коли структури даних непередбачувані, а операції охоплюють кілька індексів. Гнучка схема та можливості індексування роблять її придатною для виконання складних запитів.

3) Для середовищ з великими обсягами запису і потребою у винятковій масштабованості, Касандра є найкращим вибором. Її здатність обробляти великомасштабні додатки з інтенсивним записом робить її незамінною для систем, що вимагають високої пропускної здатності і широкого розподілу даних.

Вибираючи реляційну базу даних для проекту, дуже важливо оцінити сильні сторони кожного варіанта відповідно до вимог проекту. PostgreSQL, MySQL і Oracle Database пропонують унікальні переваги, адаптовані до різних випадків використання.

PostgreSQL добре відомий своєю надійністю та повною сумісністю з SQL, що робить його придатним для складних операцій із даними та сценаріїв, які потребують розширених можливостей SQL [116]. Він вирізняється своїми можливостями налаштування, підтримкою унікальних типів даних і функцій, які особливо корисні для геопросторових програм і розширених операцій SQL. Крім того, покращення адаптивності та продуктивності PostgreSQL сприяли його зростанню популярності для веб-додатків. Також індекси в PostgreSQL - це потужні інструменти, призначені для підвищення продуктивності запитів до бази даних за рахунок мінімізації обсягу даних, що скануються під час пошукової операції. Поширені типи індексів включають B-дерево, стандартний і найбільш універсальний тип, придатний для загального використання в різноманітних запитах, що включають пошук на рівність і в діапазоні. PostgreSQL також підтримує інші типи індексів, такі як хеш, GiST (узагальнене дерево пошуку) і GIN (узагальнений інвертований індекс), кожен з яких оптимізовано для певних типів запитів і шаблонів даних. Наприклад, GiST ефективний для індексування геометричних даних і повнотекстового пошуку, тоді як GIN краще підходить для індексування масивів даних і повнотекстового пошуку, де індекс повинен зіставляти багато значень з одним рядком. Ефективне використання індексів може значно скоротити час виконання запитів, але вони також вимагають додаткового дискового простору і можуть сповільнювати операції модифікації даних (вставки, оновлення та видалення), оскільки самі індекси потребують оновлення.

З іншого боку, MySQL відома своєю швидкою установкою та швидкістю роботи, що перевершує в середовищах, де необхідне ефективне читання даних [117]. Він зазвичай використовується для веб-додатків через бездоганну інтеграцію з PHP і його поширеність у рішеннях для веб-хостингу. MySQL найкраще підходить для програм із простими вимогами, які надають пріоритет швидким операціям читання.

База даних Oracle розроблена для задоволення вимог корпоративного рівня, пропонуючи масштабованість, високу доступність і розширені функції безпеки [118]. Він надає повний набір інструментів для аналітики, управління та розробки

додатків, позиціонуючи його як кращий вибір для критичних додатків, де надійність і розширені функції, такі як аналітика в реальному часі, є вирішальними.

Вибір серед цих баз даних залежить від різних факторів. PostgreSQL віддають перевагу для складних даних і індивідуальних рішень, MySQL для швидкості в сценаріях з інтенсивним читанням, а Oracle за його широкі можливості та надійність, незважаючи на вищу вартість ліцензування [119]. Крім того, на рішення може вплинути вибір між базами даних із відкритим вихідним кодом, що підтримуються спільнотою, і комерційно підтримуваними варіантами, виходячи з вимог до підтримки та ресурсів проекту.

Для моделі розподіленої системи мережі обрано PostgreSQL так, як ця база даних відома своєю надійністю та повною сумісністю з SQL, підтримує різний типів індексів, ефективну реплікацію та шардинг, та базу даних Касандра оскільки вона надає унікальну масштабованість завдяки своїй децентралізованій архітектурі.

3.2.2 Балансувальники навантаження

Балансувальники навантаження відіграють незамінну роль у розподілі даних між мережами та програмами, підвищуючи ефективність і надійність цифрових екосистем. Ці механізми проявляються в різноманітних конфігураціях, кожна з яких адаптована до окремих шарів інфраструктури взаємозв’язку відкритих систем (OSI), і наділені великою кількістю можливостей для задоволення нюансів вимог різноманітних контекстів.

У межах мережевого або транспортного рівня, який часто позначається як Рівень 4, такі лоад баленсери здійснюють свій вплив, керуючи трафіком, заснованим на мережевих протоколах, зокрема TCP і UDP. Їхня операційна логіка залежить від ретельного вивчення елементарних мережевих параметрів, включаючи IP-адреси, порти та діючі протоколи [120].Ця множниа лоад беленсерів навантаження відома завдяки своїй майстерності в оперативній маршрутизації пакетів, що робить його взірцевим вибором для сценаріїв, де швидкість має

першорядне значення, а складність даних, пов’язаних із програмою, є підлеглими. Зразки в цій категорії, такі як віртуальний сервер Linux і HAProxy, вихваляються за їх масштабованість і адаптивність.

Підйом до прикладного рівня, позначеного як Рівень 7, знайомить нас із більш складним жанром балансувальників навантаження. Ці об’єкти досліджують суть трафіку HTTP/HTTPS, розбираючи такі елементи, як URL-адреси, заголовки, файли cookie та тіла запиту. Така детальна перевірка полегшує обговорення нюансів маршрутизації, створюючи такі можливості, як маршрутизація, орієнтована на вміст, завершення SSL/TLS для зміцнення безпечних з’єднань, постійність сеансу для підтримки безперервності користувачів між обмінами та вдосконалені оцінки працездатності, налаштовані на протоколи додатків [121]. Мешканці цього шару, включаючи Nginx і Envoy, вирізняються своєю багатофункціональністю та узгодженістю з сучасними, орієнтованими на хмару архітектурами.

Перевершуючи елементарну функцію розподілу запитів, програмні балансувальники навантаження використовують набір алгоритмів для вдосконалення керування трафіком. Від елементарних тактик, як-от круговий, який рівномірно розподіляє запити між серверами, до складніших схем, які зважують потужність сервера або тривалість відповіді в рішеннях щодо маршрутизації [122]. Положення про налаштування алгоритмів представляє індивідуальні рішення для особливих потреб.

Розширені функції підвищують ефективність балансувальників навантаження. Регулярні оцінки працездатності підтверджують надійність серверних ресурсів, тоді як стратегії збереження сеансу та розвантаження SSL/TLS збільшують ефективність роботи та безпеку. Додаткові функції, такі як кешування вмісту та інтегровані захисні механізми, допомагають модерувати вимоги до сервера та захищати від кіберзагроз.

Коли ми дивимося на світ балансування навантаження, то бачимо, що існують різні методи, які використовуються для забезпечення ефективного розподілу вхідного трафіку між доступними серверами [123]. Кожен метод має свій

унікальний підхід і підходить для конкретних сценаріїв, беручи до уваги потужність сервера, характер запитів і необхідність підтримки постійності сесії.

Метод Round Robin (циклічний алгоритм), який нагадує карусель, де кожен сервер по черзі обробляє вхідний запит по колу. Цей метод вирізняється своєю простотою, що робить його найкращим вибором для рівномірного розподілу простих завдань, таких як обслуговування статичних веб-сторінок, між серверами зі схожими можливостями [124]. Однак його простота також означає, що йому не вистачає тонкощів для роботи з серверами різної потужності або перевірки стану серверів, що потенційно може призвести до виникнення вузьких місць або продовження розподілу запитів на сервер, що вийшов з ладу.

Стратегія найменшої кількості з'єднань (Least Connections), яка нагадує вибір найкоротшої черги в кав'ярні. Цей динамічний метод розподіляє нові запити на сервер з найменшою кількістю активних з'єднань, що дозволяє більш збалансовано розподіляти довготривалі завдання. Це особливо корисно при роботі із запитами, які сильно відрізняються за часом обробки. Тим не менш, ця стратегія може бути не настільки ефективною для швидких, короткочасних запитів і не враховує обчислювальну потужність серверів, що може призвести до того, що деякі з них будуть перевантажені складними завданнями, незважаючи на меншу кількість з'єднань.

Нарешті, метод Source IP Hash створює унікальне хеш-значення з IP-адреси клієнта, щоб постійно направляти його запити на один і той самий сервер. Цей метод ідеально підходить для додатків, що вимагають постійності сеансів, таких як кошики для онлайн-покупок або логіни користувачів, забезпечуючи безперебійну роботу для користувача. Однак можливість хеш-колізій - коли різні IP-адреси призводять до одного і того ж виділення сервера - може призвести до нерівномірного розподілу навантаження [125]. Крім того, цей метод не враховує поточне навантаження або стан здоров'я серверів, потенційно призначаючи нові запити на вже перевантажені сервери.

Кожен з цих методів балансування навантаження пропонує окремий підхід до управління вхідним трафіком, з певними перевагами та міркуваннями. Вибір

методу залежить від конкретних вимог інфраструктури та додатків, які вона підтримує, балансуючи між простотою, ефективністю та необхідністю постійних сесій. Для системи соціальної мережі на початку буде використано циклічний алгоритм для балансувальника, а при більшій мастабованості системи буде використовуватися стратегія найменшої кількості з’єднань.

3.2.3 Пакетна та потокова обробка

Пакетна обробка - відома парадигма в управлінні даними, працює за принципом агрегування наборів великих даних через визначені інтервали для подальшого комплексного аналізу та послідовного виконання. Цей підхід є особливо вигідним у сценаріях, коли миттєвість результатів підпорядковується вимозі ефективної обробки великих обсягів даних [126]. Бетч процесинг, яка переважно використовується в областях аналізу історичних даних, фінансової звітності та проведення наукового моделювання, надає перевагу тому, що наголошується на ретельному дослідженні даних, а не на терміновості негайного зворотного зв’язку.

Навпаки, потокова обробка вводить сучасний погляд на управління даними, концептуалізуючи дані як безперервний потік. Ця парадигма обробляє окремі елементи даних у міру їх отримання, що різко контрастує з бетч процесингом обробки даних. Аналогічно безперервно працюючому фабричному конвеєру, потокова обробка полегшує аналіз даних у реальному або майже реальному часі, задовольняючи програми, які вимагають прискореного аналізу, наприклад миттєве виявлення шахрайських транзакцій або моніторинг у реальному часі даних датчиків IoT для оперативних сповіщень і профілактичних заходів [127].

Основна розбіжність між бетч процесингом та потоковою обробкою полягає в їхніх відповідних підходах до часу обробки даних. Бетч процесинг за своєю суттю включає затримку через фазу накопичення даних, тоді як потокова обробка призначена для зменшення затримки, що сприяє швидшому управлінню даними. Крім того, хоча пакетна обробка здатна відновлювати процеси після збоїв, потокова

обробка потребує складних методологій для керування станом і відновлення, щоб гарантувати безперервність роботи.

Вибір між цими методологіями залежить від необхідності своєчасного аналізу інформації, обсягу та швидкості обробки даних, а також наявної інфраструктури. Сучасне середовище керування даними все частіше поєднує пакетну та потокову обробку в просунуті гібридні системи, використовуючи сильні сторони, притаманні кожному підходу, для підвищення ефективності та оперативності в операційних рамках [128].

У системі соціальної мережі для забезпечення швидкої класифікації текстів на спам і пропаганду буде використовуватися пакетна обробка даних, і файлове сховище, Amazon S3.

3.2.4 Брокер повідомлень

Брокери повідомлень незамінні у сфері комунікації розподілених систем, особливо в контексті архітектур мікросервісів і складних взаємозв'язків соціальних мереж [129]. Ці брокери, діючи як програмне забезпечення-посередник, є фундаментальними для полегшення обміну повідомленнями між різними програмами, службами та системами. Їхня здатність забезпечувати надійну та асинхронну доставку повідомлень покращує модульну архітектуру за рахунок відокремлення компонентів, що має вирішальне значення для підтримки надійності та гнучкості системи.

Серед низки доступних брокерів повідомлень Apache Kafka є відомою розподіленою потоковою платформою, яка відома своїми чудовими характеристиками продуктивності. Архітектура Kafka, яка вміло поєднує функції посередництва повідомлень і потокової аналітики, особливо добре пристосована для роботи з об'ємними вимогами до даних у реальному часі, поширеними в середовищах соціальних мереж [130]. ЇЇ модель публікації-підписки сприяє швидкому розповсюдженню інформації, позиціонуючи його як зразкове рішення

для забезпечення спілкування в реальному часі, відстеження активності та потокової передачі оновлень.

Відомість Kafka пояснюється декількома ключовими перевагами, зокрема її здатністю швидко обробляти великі дані, великий обсяг повідомлень, що є важливою особливістю соціальних мереж, які характеризуються інтенсивною активністю. Вона також може похвалитися надійними механізмами зберігання, які захищають від втрати даних і підтримують повторне відтворення даних для аналітичних або відновлювальних цілей. Крім того, дизайн Kafka передбачає масштабоване розширення для вирішення зростаючого трафіку даних, характерного для зростаючих платформ соціальних мереж. Конструкція Kafka дозволяє ефективно обробляти величезні обсяги даних, що робить її ідеальним вибором для програм, які мають обробляти або аналізувати великі потоки даних у реальному часі. ЇЇ архітектура створена для забезпечення високої пропускної здатності як для публікації, так і для підписки на повідомлення, навіть незважаючи на тисячі повідомлень на секунду. Однак такі проблеми, як складність керування системою та обмеження її основного, тематичного підходу до маршрутизації, зменшують її ефективність.

Навпаки, RabbitMQ вирізняється своєю адаптивністю та надійністю. Як брокер повідомлень з відкритим вихідним кодом, який підтримує різноманітні протоколи обміну повідомленнями, насамперед AMQP, RabbitMQ вміє керувати різноманітними шаблонами зв’язку. Ця універсальність робить його надійним вибором для задоволення багатогранних вимог до обміну повідомленнями, властивих соціальним мережамх [131]. Розширені можливості маршрутизації RabbitMQ, надійна доставка повідомлень, а також підтримка встановленої екосистеми та комплексного інструментарію додатково підкреслюють її придатність для програм соціальних мереж, незважаючи на потенційні обмеження пропускної здатності та складності керування кластером.

Apache ActiveMQ з його високою продуктивністю, масштабованістю та широкою підтримкою протоколів є ще одним важливим варіантом. Його функції, призначені для запобігання втраті даних і забезпечення стійкості системи,

задовольняють потреби різноманітних та інтегрованих середовищ, типових для соціальних мереж. У той час як ActiveMQ полегшує інтеграцію різних технологій у соціальні мережі завдяки розширеній підтримці протоколів і розроблено для ефективної обробки великих навантажень обміну повідомленнями, операційні вимоги, пов’язані з його конфігурацією та керуванням, становлять значні проблеми.

Redis є широко використовуваним брокером повідомлень на платформах соціальних мереж завдяки винятковим показникам затримки, що робить його ідеальним для інтерактивних функцій у реальному часі, таких як обмін миттєвими повідомленнями та живі сповіщення [132] . Він особливо підходить для обробки ефемерних даних, таких як тимчасові оновлення статусу або індикатори присутності користувачів, де негайна доступність даних має пріоритет над довгостроковим зберіганням. У мікросервісних архітектурах, поширених у соціальних мережах, Redis ефективно сприяє спілкуванню між сервісами, особливо в середовищах обміну повідомленнями великого обсягу.

Однак, впроваджуючи Redis на платформах соціальних мереж, необхідно враховувати критичні міркування. Для компонентів, які вимагають збереження даних, як-от прямий обмін повідомленнями, надійні стратегії збереження, такі як знімки лише файлу додавання (AOF) або Redis Database (RDB), необхідні для запобігання втраті даних. Оцінка масштабованості Redis і його здатності обробляти високу пропускну здатність повідомлень має вирішальне значення для великомасштабних соціальних мереж, що потенційно вимагає пошуку альтернативних брокерів повідомлень [133]. Крім того, для додатків, які потребують суворих гарантій доставки або складних можливостей маршрутизації, можуть знадобитися додаткові інструменти або спеціальні зусилля для розширення Redis.

З іншого боку, Google Cloud Pub/Sub пропонує масштабованість для розповсюдження подій у соціальних мережах, що робить його придатним для розповсюдження оновлень у широких мережах підписників або для трансляції подій. Його можливості географічно різноманітної доставки з низькою затримкою

є перевагою для глобальних соціальних мереж, забезпечуючи швидке розповсюдження даних у різних регіонах. Модель керованого сервісу Pub/Sub підвищує надійність і спрощує керування інфраструктурою, дозволяючи командам розробників зосередитися на вдосконаленні функцій соціальних мереж [134]. Крім того, Pub/Sub просуває слабозв'язані архітектури в соціальних мережах, сприяючи незалежному масштабуванню та розвитку компонентів для підвищення адаптивності та стійкості.

Підсумовуючи, і Redis, і Google Cloud Pub/Sub пропонують унікальні переваги для посередництва повідомлень на платформах соціальних мереж, причому Redis перевершує взаємодію в реальному часі та обробку тимчасових даних, тоді як Google Cloud Pub/Sub забезпечує масштабованість, глобальну оптимізацію доставки та спрощену управління інфраструктурою.

У даній роботі, буде використано комбінацію меседж брокера Кафка, оскільки саме цей меседж брокер має високу масштабованість за рахунок зберігання черги повідомлення в декількох розділах, та базу Редіс, для обробки статусів повідомлень.

3.2.5 Хмарні об'єктні сховища

Хмарне зберігання об'єктів являє собою фундаментальну зміну парадигм зберігання даних, відхід від ієрархічних структур файлових систем або секторної організації блокового зберігання. Натомість він інкапсулює дані як окремі об'єкти, кожен з яких містить описові метадані та глобальний унікальний ідентифікатор [135]. Ця архітектура забезпечує надзвичайну масштабованість, роблячи хмарне сховище об'єктів винятково придатним для керування величезною кількістю неструктурованих даних (наприклад, мультимедіа, документів, даних датчиків).

Широке впровадження хмарного сховища об'єктів можна віднести до його невід'ємних переваг. Можливість легкого масштабування до рівнів петабайтів забезпечує організаційне зростання без ускладнень, пов'язаних із традиційним розширенням сховища. Крім того, стратегії реплікації хмарних провайдерів у

розподілених центрах обробки даних зменшують ризики втрати даних і гарантують високу доступність. Застосування моделей ціноутворення за принципом «оплата за використання» часто призводить до економії коштів порівняно з внутрішньою інфраструктурою зберігання даних. Використання стандартних веб-протоколів (HTTP/HTTPS) і RESTful API спрощує інтеграцію з програмами, сприяючи доступності з будь-якого місця [136] . Хмарні постачальники інвестують значні кошти в заходи безпеки, такі як шифрування, контроль доступу та керування ідентифікацією, щоб захистити конфіденційні дані.

Служба Amazon Simple Storage Service (S3) вирізняється своєю історією та зрілістю, що забезпечує повний набір функцій і надійну роботу, життєво важливу для архівування даних соціальних мереж. Його повна інтеграція з іншими веб-сервісами Amazon підвищує ефективність керування великими даними, а його різноманітні класи зберігання дозволяють використовувати економічно ефективні стратегії архівування даних [137]. Підтримка S3 версії та керування життєвим циклом допомагає підтримувати цілісність даних у розподілених мережах. Однак його складна структура ціноутворення та потенціал мінливості продуктивності, а також проблеми, пов’язані з його можливою моделлю узгодженості, можуть вплинути на досвід пошуку даних.

Хмарне сховище Google (GCS) вирізняється своєю винятковою продуктивністю в отриманні даних, що має вирішальне значення для підтримки чутливого інтерфейсу користувача в розподілених соціальних мережах. GCS спрощує складання бюджету завдяки прямому ціноутворенню та забезпечує узгодженість даних завдяки чіткій узгодженості в усій мережі. Незважаючи на його переваги, міркування щодо регіоналізації та можливості прив’язки постачальника до інструментів Google Cloud Platform можуть вимагати ретельного планування стратегій резервування та міграції.

Microsoft Azure Blob Storage відома своєю інтеграцією з екосистемою Azure, пропонуючи безперебійну роботу з великими даними для мереж, які залежать від набору служб Azure. Його багаторівнева система зберігання підтримує управління витратами відповідно до зменшення частоти доступу до публікацій у соціальних

мережах, а його сумісність із гібридними середовищами підвищує гнучкість організації [138]. Однак складність ціноутворення Azure та обмеження регіонального покриття, а також занепокоєння щодо прив’язки постачальника підкреслюють важливість ретельного планування в процесі впровадження.

У даній роботі буде використовуватися Amazon Simple Storage Service, оскільки це мастабована технологія, та добре інтегрована з іншими компонентами AWS (amazon web services).

3.2.6 Шлюз API

У розподілених системах, особливо тих, що мають великий обсяг трафіку, впровадження шлюзів інтерфейсу прикладного програмування (API) все частіше визнається критичною стратегією. Ці шлюзи функціонують як координаційна точка, через яку керуються та обробляються всі запити API, таким чином слугуючи центральним механізмом контролю в системі. Цей централізований підхід пропонує кілька переваг, зокрема оптимізацію процесів зв’язку та підвищення безпеки системи, а також створює певні проблеми, особливо у великомасштабних архітектурних структурах.

Шлюзи API відіграють вирішальну роль у забезпеченні безперебійної роботи розподілених систем. Вони відповідають за інтелектуальне керування потоком запитів до та від внутрішніх мікросервісів. Це включає ряд функцій від маршрутизації, забезпечення спрямування запитів до відповідної служби, до автентифікації та авторизації, коли шлюз перевіряє особу користувачів і гарантує, що вони мають необхідні дозволи для доступу до певних ресурсів [139] . Крім того, цим шлюзам доручено підтримувати цілісність системи шляхом обмеження швидкості та регулювання, щоб запобігти перевантаженню служби та потенційним зловживанням, тим самим захищаючи серверні ресурси.

На додаток до цих обов’язків шлюзи API служать мостом для перекладу протоколів, забезпечуючи безперебійний зв’язок між клієнтами та службами, навіть якщо вони використовують різні протоколи зв’язку. Деякі шлюзи також

включають функції балансування навантаження, розподіляючи вхідний трафік таким чином, щоб оптимізувати використання системних ресурсів. Крім того, кешуючи дані, які часто запитуються, ці шлюзи можуть значно зменшити робоче навантаження на сервер і покращити загальний час відповіді. Вони також відіграють важливу роль у системному моніторингу та веденні журналів, пропонуючи цінну інформацію, яка допомагає виявити та вирішити проблеми.

Процес вибору відповідного шлюзу API складний і вимагає ретельного розгляду різних факторів. До них належать продуктивність і масштабованість шлюзу, зокрема його здатність обробляти великі обсяги запитів з мінімальною затримкою та його здатність розвиватися разом із системою. Функції безпеки також мають першочергове значення, оскільки шлюз повинен пропонувати надійні механізми автентифікації, авторизації, обмеження швидкості та перевірки вхідних даних для захисту від потенційних загроз. Крім того, у процесі вибору слід враховувати набір функцій шлюзу, щоб забезпечити його відповідність архітектурним потребам системи, а також досвід розробника, який він пропонує, включаючи простоту використання, налаштування та підтримку спільноти [140]. Нарешті, доступний рівень підтримки постачальників має вирішальне значення для вирішення будь-яких проблем або потреб у налаштуванні, які можуть виникнути.

Розглядаючи варіанти для шлюзів API, існує кілька доступних маршрутів, кожен із яких має власний набір переваг. Хмарні шлюзи, такі як AWS API Gateway, Azure API Management і Google Cloud Endpoints, пропонують масштабованість та інтеграцію з відповідними хмарними службами, якими керує постачальник. Шлюзи з відкритим кодом, як-от Kong, Tyk і Ambassador, забезпечують гнучкість і потенціал налаштування, що часто призводить до нижчої загальної вартості володіння. Крім того, створені на замовлення шлюзи забезпечують максимальний контроль над системою, але потребують значних ресурсів для розробки та обслуговування.

Впровадження шлюзу API дає кілька переваг, зокрема спрощену взаємодію з клієнтом через абстракцію серверних служб, покращену безпеку через централізоване застосування та оптимізовану продуктивність завдяки таким

функціям, як кешування та балансування навантаження. Крім того, ці шлюзи дозволяють абстрагувати складність серверної частини, дозволяючи плавно змінювати та керувати в архітектурі. Однак потенційні недоліки включають ризик додаткової затримки для кожного запиту, можливість того, що шлюз стане єдиною точкою збою, якщо він не розроблений з урахуванням резервування, а також додаткову складність в управлінні та підтримці шлюзу, особливо для систем із складними варіантами використання [141].

Таким чином, роль шлюзів API у розподілених системах багатогранна, охоплюючи, серед іншого, керування потоками запитів, безпеку, трансляцію протоколів та оптимізацію продуктивності. Їх реалізація вимагає збалансованого розгляду можливостей, функцій безпеки, масштабованості та конкретних потреб архітектури системи. Незважаючи на труднощі, стратегічне використання шлюзів API може значно підвищити ефективність, безпеку та керованість розподілених систем.

У системі соціальної мережі даної роботи буде використовуватися шлюз - AWS API Gateway.

3.2.7 Формати та протоколи передачі даних

У сфері розподілених мереж вибір відповідного формату даних є ключовим рішенням, яке впливає на ефективність і результативність зв’язку між різними системами. На цей вибір впливають різні фактори, зокрема потреба в зручності для читання, продуктивності, цілісності даних і гнучкості адаптації до майбутніх змін.

Одним із широко поширених форматів є нотація об’єктів (JSON), відома завдяки своїй текстовій структурі, яку можна прочитати людиною. Це вихід для веб-сервісів та інтерфейсів прикладного програмування (API), що сприяє бездоганній інтеграції з різними мовами програмування та платформами. Його легка природа значно сприяє його популярності, пропонуючи баланс між простотою та функціональністю, що є особливо корисним у веб-розробці, де швидкий аналіз та обмін даними є вирішальними.

Розширювана мова розмітки (XML) пропонує інший підхід, віддаючи пріоритет складності та цілісності структури даних. Він виділяється своєю здатністю представляти складні структури даних, що підтримується механізмами перевірки на основі схем, такими як XSD (визначення схеми XML) і DTD (визначення типу документа) [142]. Це робить XML сильним кандидатом для програм, де ретельна перевірка та цілісність даних є важливою, забезпечуючи надійну структуру для забезпечення відповідності даних попередньо визначеним стандартам і форматам.

Коли продуктивність і ефективність пропускної здатності мають першочергове значення, в гру вступають буфери протоколів (Protobuf). Цей двійковий формат чудово підходить у середовищах, які вимагають високошвидкісної серіалізації та мінімального розміру даних, завдяки своїй мовно-нейтральній природі, керованій схемою [143]. Protobuf розроблено для того, щоб навіть найвибагливіші додатки могли ефективно спілкуватися, що робить його ідеальним вибором для внутрішніх протоколів зв’язку у великих розподілених системах.

Avro представляє себе як універсальний двійковий формат, який підтримує еволюцію схеми, дозволяючи легко модифікувати структури даних без порушення сумісності. Ця адаптивність має вирішальне значення для розподілених систем, які мають розвиватися з часом, пропонуючи спосіб адаптуватися до нових вимог, зберігаючи безперебійне обслуговування.

Thrift зосереджується на сприянні міжмовному спілкуванню сервісів, включаючи двійковий формат із інтегрованою платформою Remote Procedure Call (RPC). Це робить його потужним інструментом для побудови розподілених систем, які повинні забезпечити безперебійну взаємодію між службами, написаними різними мовами програмування, підвищуючи взаємодію та зменшуючи складність інтеграції послуг .

Для додатків, які підкреслюють швидкість і компактність даних, MessagePack пропонує ефективний двійковий формат серіалізації. Він розроблений для зменшення накладних витрат на мережу та затримки, забезпечуючи швидкий і

компактний метод обміну даними, що є особливо цінним у середовищах, чутливих до часу та обмежених ресурсів.

YAML зі своїм зручним для людини синтаксисом на основі відступів розроблено для сценаріїв, коли людина взаємодіє з даними часто, наприклад у файлах конфігурації. Його читабельність і простота роблять його кращим вибором для налаштувань і документації, де пріоритетом є ясність і легкість використання.

Отже, для високонавантажених розподілених соціальних мереж, де продуктивність і ефективне використання пропускної здатності є критичними, пропонується модель, яка використовує Protobuf через їх компактний розмір, швидкість і структуровану природу їхньої схеми. Особливо це ефективно при роботі з багатьма мікросервсіми. Для для зовнішніх API або інтерфейсів пропонується використовувати Json, бо продуктивність розробника тут та взаємодія з веб-технологіями є пріоритетними [144].

Протоколи даних, які можуть використовувати для передачі даних в високонавантажених розподілених системах соціальних мереж. Архітектурний дизайн, відомий як Representational State Transfer (REST), дотримується моделі зв’язку клієнт-сервер, головним чином заснованої на механізмах запит-відповідь, переважно через протокол HTTP, причому найпоширенішою версією є HTTP/1.1. REST переважно використовує JSON для обміну даними, хоча він також сумісний з XML та різними іншими форматами даних. Це особливо корисно для виконання операцій створення, читання, оновлення та видалення (CRUD) на веб-ресурсах. Крім того, REST добре підходить для розробки загальнодоступних API, які вимагають широкої сумісності, надаючи перевагу простоті та комплексній підтримці в різних веб-браузерах [145].

Виклик віддаленої процедури (gRPC) — це складна структура, призначена для того, щоб служби могли пропонувати функції, які можна викликати віддалено, використовуючи можливості протоколу HTTP/2 для підвищення продуктивності за допомогою розширених функцій, таких як мультиплексування. Для передачі даних gRPC використовує буфери протоколів, ефективну техніку двійкової серіалізації. Цей фреймворк особливо ефективний для полегшення зв’язку між мікросервісами

у внутрішніх мережах, забезпечуючи швидкий і ефективний обмін даними. Він також ідеально підходить для сценаріїв, що вимагають миттєвого зв’язку з низькою затримкою, а також у багатомовних середовищах, де критично потрібна плавна комунікація різними мовами програмування [146].

Протокол WebSockets забезпечує постійний двонаправлений канал зв’язку через окреме з’єднання TCP. Він здатний обробляти як текстові, так і двійкові формати даних, пропонуючи гнучке рішення для різноманітних потреб додатків. WebSockets ідеально підходять для програм реального часу, включаючи платформи миттєвого обміну повідомленнями, інтерактивні інформаційні панелі, фінансові тікери, а також для передачі повідомлень або оновлень із серверів клієнтам без регулярного опитування [147].

Передача повідомлень представляє більш охоплюючу стратегію системного зв’язку, що характеризується обміном дискретними повідомленнями між компонентами, на відміну від використання прямих викликів функцій. Ця стратегія не лише обмежується REST, gRPC і WebSockets, але й поширюється на інші комунікаційні технології, кожна з яких надає певні переваги, адаптовані до конкретних операційних вимог і сценаріїв. Використовуючи передачу повідомлень, системи можуть досягти вищих рівнів модульності та відокремленості, тим самим підвищуючи масштабованість і полегшуючи обслуговування. Ця методологія підкреслює важливість гнучких і динамічних механізмів зв’язку в сучасній архітектурі програмного забезпечення та проектуванні системи.

Для розподіленої системи соціальної мережі важливо використовувати комбінацію цих протоколів, для отримання найкращої швидкості та надійності системи. Для стандартних API запитів до сервісів спочатку в моделюванні використовуються REST запити, адже вони є найпростішими в імплементації, за допомогою метрик оцінювати слабкі місця отриманої системи і покращувати їх за рахунок переходу на gRPC.

3.3 Проектування моделі системи

Користувачі можуть бути зареєстровані за допомогою пост запитів в запропонованій системі. Під час входу в систему користувач отримає токен JWT [148] , за допомогою якого він зможе виконувати інші операції в системі.

Цей JWT токен буде підписаний ключев в DigiCert. Центр сертифікації DigiCert буде використаний для запобігання атаці «людина посередині», шифрування та захисту даних користувачів. Він використовується у великих корпораціях, таких як Amazon, Microsoft, NASA [149].

У цій роботі для цієї мети та всіх подальших операцій буде використовуватися протокол https.

Приклад структури таблиці користувача (табл. 3.1):

Таблиця 3.1.

Користувач

Field name	**Field type**
user_id (pk)	int
email	varchar
user_name	varchar
country	varchar
created_at	timestamp
updated_at	timestamp
password	varchar

Другий набір операцій - це підписка та відписка від інших користувачів. Ці операції можна виконати за допомогою запиту REST API. Таблиця зв'язків між користувачами може виглядати так (табл. 3.2):

Статус тут може приймати два значення «Підписався» і «Відписатися».

Таблиця 3.2.

Підписники

Field name	Field type
follower_id	int
following_id	int
follow_date	datetime
status	varchar
created_at	timestamp
updated_at	timestamp

Третя операція. Для створення публікацій дані публікації (зображення, текст) і метадані публікації відокремлені один від одного. Дані публікацій можна зберігати в сегментах S3 веб-сервісів Amazon і зберігати посилання на цей сегмент у таблиці Postgresql.

Таблиця 3.3 показує, як буде виглядати структура постів.

Таблиця 3.3

Пости

Field name	Field type
post_id (pk)	uuid
user_id (fk)	int
created_at	timestamp
updated_at	timestamp
deleted_at	timestamp
s3_bucket_path	varchar

Четверта операція. Користувачі повинні мати можливість залишати коментарі під іншими публікаціями користувачів.

Ось запропонована структура (табл. 3.4):

Таблиця 3.4

Коментарі

Field name	Field type
comment_id (pk)	uuid
post_id (fk)	uuid
user_id (fk)	int
created_at	timestamp
updated_at	timestamp
text	varchar

Для цієї таблиці в полі user_id використовуватимуться шардинг Postgresql [150]. Або ми можемо використовувати тут базу даних (БД) Касандра, якщо нам потрібна більш висока масштабованість [151].

Коментарі стають видимими лише тоді, коли користувач натискає на якусь публікацію. Операції перетину реляційних баз даних тут не потрібні, тому за потреби тут можна вибрати базу даних із широким стовпцем Cassandra. І, нарешті, користувач може відкрити свою сторінку та отримувати свої новини.

Завдання створення стрічки новин полягає в тому, щоб отримати всі повідомлення від користувачів, на яких підписані поточні користувачі.

Простий спосіб зробити це — написати SQL-запит таким чином (синтаксис Postgresql):

```
SELECT posts.*
FROM posts
JOIN followers
ON posts.user_id = followers.following_id
WHERE follow.follower_id = :my_user_id
ORDER BY posts.post_date DESC;
```

І це справді спрацює для невеликої кількості користувачів. Але для масштабованості це спричинить проблеми, оскільки тут ми припускаємо, що всі

метадані публікацій зберігаються в одній таблиці в одному місці. Однак дані про таку кількість щоденних активних користувачів, як у Facebook (приблизно 2 мільярди [152] і Twitter (приблизно 200 мільйонів [153]) , не можуть зберігатися в одному місці, на одному сервері. Отже, потрібне розділення даних. Двома розумними варіантами розділення таблиці повідомлень були б за user_id або created_at/updated_at time. Але проблема з updated_at time з'являється в тому, що значення updating_at видалить поточну публікацію зі старого місця та створить поточну публікацію в новому місці. Отже, User_id можна використовувати як ключ розділу.

Деякі реляційні бази даних, такі як Postgresql, дозволяють користувачам об'єднувати таблиці, які мають декілька кластерів. Але ці запити повільніші за звичайні. З метою оптимізації рішення, новини можуть бути попередньо обчислені для користувачів офлайн. І коли користувач зайде в мережу, йому будуть показані попередньо обчислені дописи.

Усіх користувачів можна розділити на дві групи: зірки (більше 5-ти тисяч підписників) і звичайні люди. Усі дописи користувачів-зірок будуть розміщені в одному кластері таблиці Postgresql. Отже, коли користувач запитує публікацію для користувача-зірки, він запитує це за допомогою SQL-запиту на вибір, як вище.

Але для дописів звичайних людей буде використано попереднє обчислення. Поява публікації буде додана, коли звичайна особа, за якою стежить поточний користувач, публікує щось, у таблиці precomputed_post. Отже, коли користувач запитує свою стрічку новин, йому потрібно буде просто запитати цю precomputed_post таблицю за своїм user_id. Тут будемо використовувати базу даних Кассандра, [154] оскільки вона має властивість масштабованості, і тут нам не потрібні операції з'єднання. Ми можемо створити індекс для user_id у цій таблиці для швидшого пошуку.

Іншим способом зберігання цього списку попередніх обчислень може бути Redis DB. Але важче налаштувати Redis, щоб він був настільки масштабованим [155].

На даний момент запропонована архітектура системи соціальної мережі (рис. 3.1).

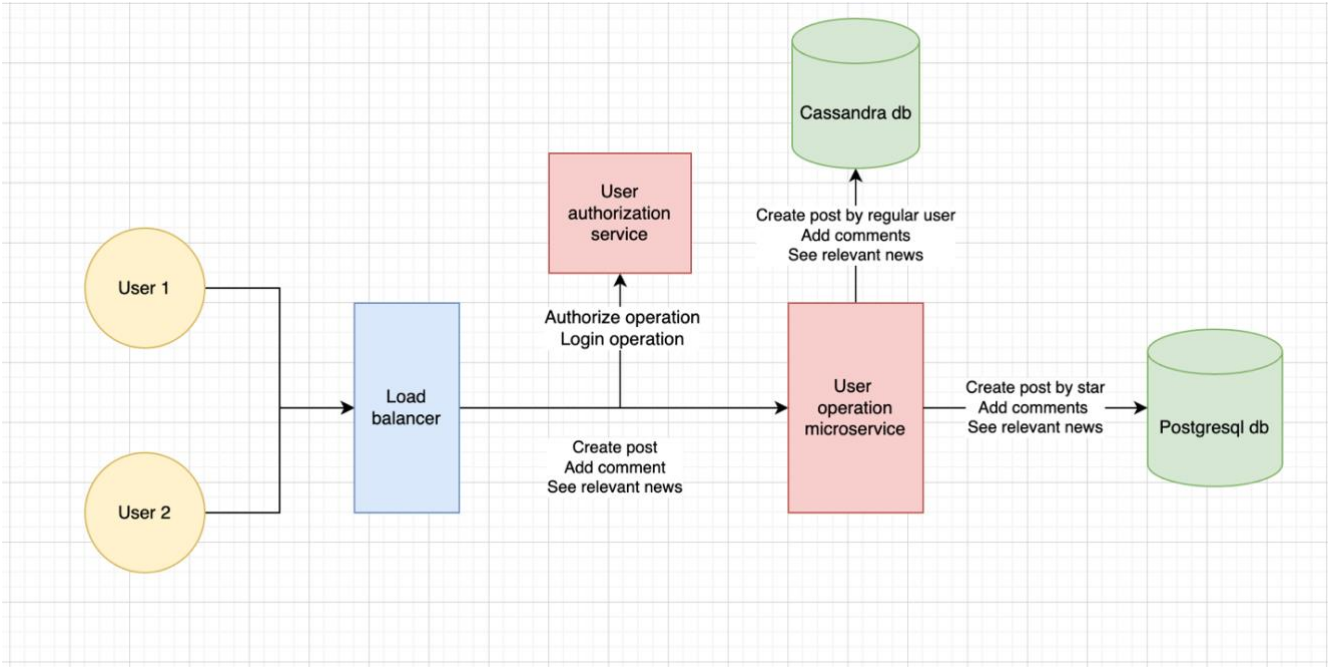


Рис. 3.1 Архітектура системи соціальної мережі

3.4 Інтеграція методів для обробки спаму, пропаганди та ефекту «бульбашки» в модель системи

Опишемо процес інтеграції нейронних мереж в модель системи високонавантаженої розподіленої мережі.

Після створення кожного посту чи коментаря це буде записано в меседж брокер Кафка. Це можна робити асинхронно під час пост-запиту. Після цього за допомогою мікросервісу, який буде отримувати повідомлення з Kafka, ці повідомлення будуть пакетуватися і зберігатися в Amazon S3 бакети, які досить ефективно зберігають довільні дані [14], а інший топік Кафки буде використовуватися для подальшої передачі цих файлів. Після цього нейронна мережа обробки спаму і пропаганди буду паралельно обробляти ці s3-файли один за одним і оновлювати дані про пости в базі даних Кассандра. Управління знаннями буде відбуватися таким чином, що коли користувач буде отримувати дані про пости

в своїй стрічкці новин, він також буде робити один додатковий запит, щоб отримати інформацію про пост, а саме чи є пост пропагандою чи спамом.

Оновлена архітектура, для фільтрації спаму і пропаганди, представлена на рис. 3.2.

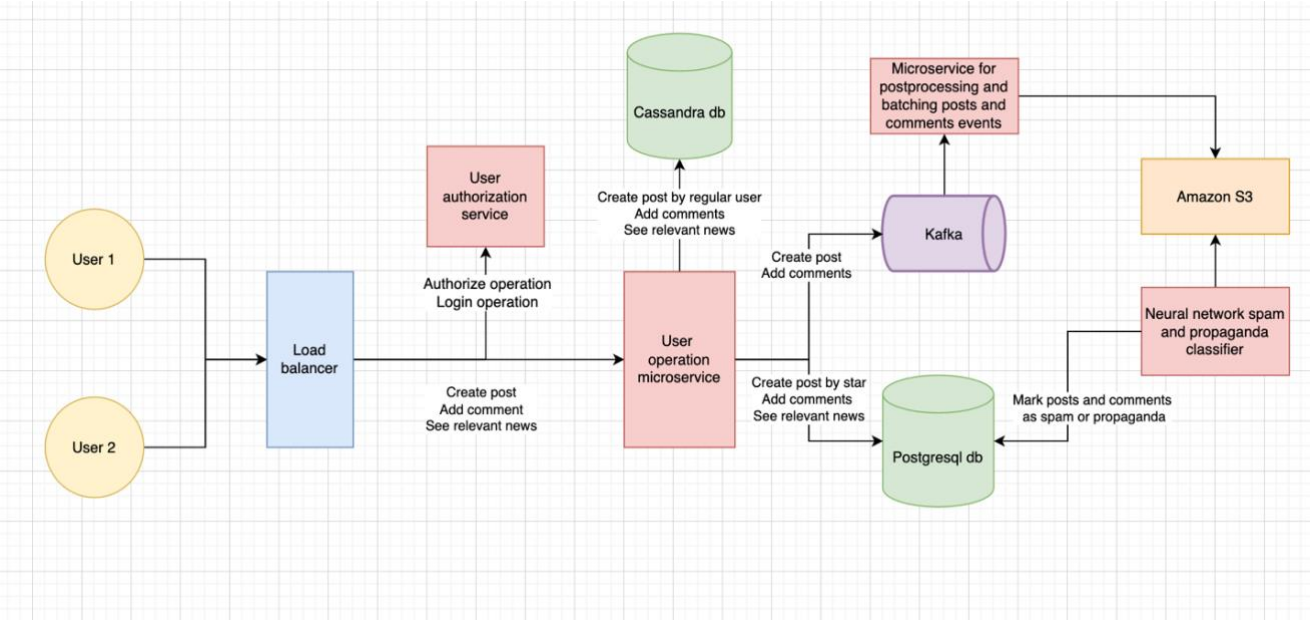


Рисунок 3.2 Інтеграція методу класифікації на спам і пропаганду в архітектуру системи соціальної мережі

Наступним етапом побудови архітектури соціальної мережі - це інтегрувати алгоритм генерації стрічки новин для уникнення ефекту бульбашки (табл. 3.3).

Першою частиною збору інформації буде її отримання через брокер повідомлень Кафка. Це дозволить не мати надмірного навантаження на подальші сервіси [156].

Якщо в майбутньому виникнуть проблеми з масштабованістю, можна використати шардинг за user_id, щоб вирішити цю проблему.

Якщо виникає проблема із затримкою запитів, можна використовувати індекси бази даних на основі B-дерева або дерева Log-Structured Merge (LSM), залежно від того, яких запитів у нас більше, запитів на читання чи запис. B-дерева є популярними рішеннями для важких запитів на читання, а LSM-дерева є популярними рішеннями для важких запитів на запис.

Самі дані публікації, як-от зображення, аудіо чи текст, зберігаються в сховищі S3. Це популярне рішення для зберігання необроблених великих даних, яке добре масштабується [157].

Повідомлення мікросервісу обробки запитів користуваі також буде передано до служби нейронної мережі кластеризації тем, щоб кластеризувати повідомлення за темами та розділяти їх за протилежними та різними думками. Вектор ознак буде створено для кожної публікації як результат нейронної мережі кластеризації теми.

Пізніше в потоці обробки повідомлення цей вектор буде наданий, як вхідні дані для визначення пріоритету постів нейронною мережі.

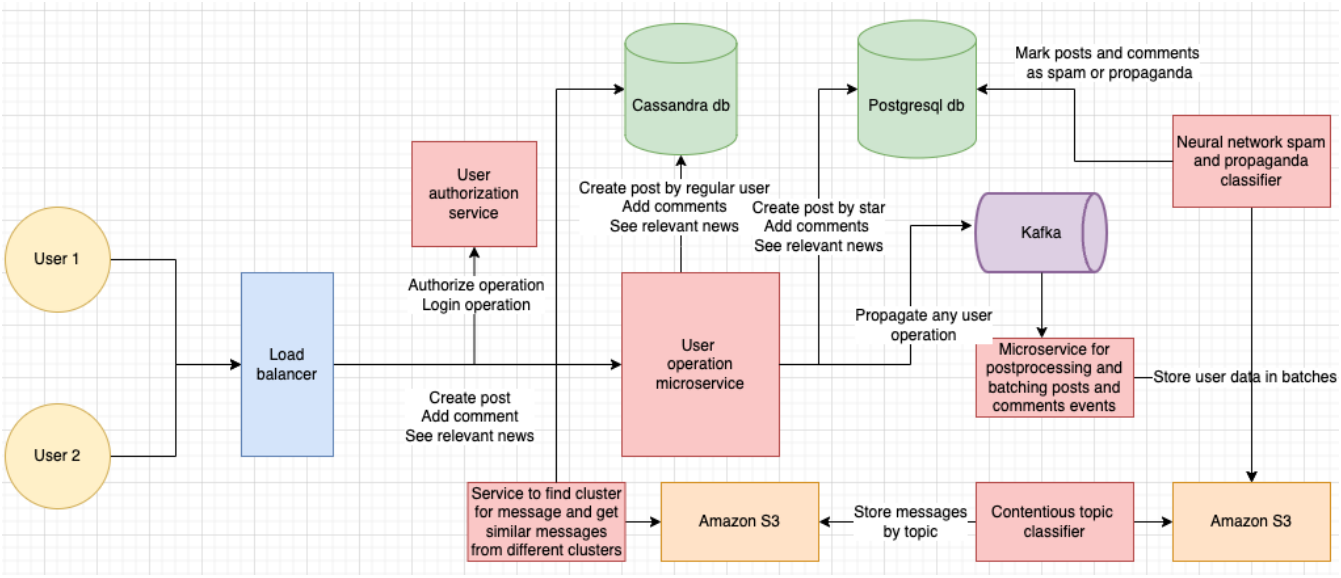


Рисунок 3.3 Інтеграція методу диверсифікації новин на основі кластерихації по дискусійних темах

Сервіс пріоритезації нейронною мережі надаватиме пріоритет усім публікаціям користувачів, у тому числі публікаціям, які відрізняються від поточної думки користувачів, а сервіс генерації стрічки новин повертатиме їх за запитом користувача.

3.5 Оцінка швидкодії оновленої системи

Кожна частина системи може бути масштабована горизонтально, включаючи сам сервіс нейронної мережі. У середовищі kubernetes використовується масштабований об'єкт для того, щоб динамічно обробляти кількість екземплярів кожного сервісу. Переміщення більшої частини даних до бази даних Касандра також забезпечує високу масштабованість завдяки її децентралізованій архітектурі.

Нейронні мережі описані в минулому розділі здатні обробляти 1 тисячу текстів за 1-1.5 сек. в залежності від довжини текстів. Кількість постів в Ікс (Твіттері) в середньому за секунду становить 6000. Тому в середньому для обробки даних в реальному часу необхідно на 6-9 сервісів на якому виконуються класифікація за допомогою нейронними мережами.

У разі різкого збільшення кількості постів, наприклад, у 10 разів, є два варіанти: збільшити в 10 разів кількість екземплярів нейромережевого сервісу, щоб продовжити обробку майже в режимі реального часу, або дозволити деяку затримку в обробці текстів, щоб тексти були промарковані протягом наступної години, або навіть довшого проміжку часу.

У другому випадку система буде збільшувати кількість екземплярів нейронної мережі на основі вікна агрегації кількості постів протягом певного часу. Оскільки система обробляє пости через події брокера повідомлень Кафки, то можна використовувати обидва підходи.

Отже, якщо в системі не спостерігається сплесків кількості дописів, то дописи можуть бути позначені як спам і пропаганда через кілька десятків секунд після створення.

У випадку сплесків трафіку у kubernetes, поточна система потребує запуску нових сервісів нейромережев, що може зайняти від десятків секунд до декількох хвилин, залежно від обсягу пам'яті та процесорних ресурсів, виділених для кожного сервісу. Крім того, брокер повідомлень Кафка потребує перебалансування розподілу розділів споживачів під час створення нових нейромережевих сервісних блоків, що також приблизно займе 1 хвилину. Це значення може бути налаштоване

в залежності від розміру пакету повідомлень у S3, що створюється сервісом для постобробки та пакетування постів.

Таким чином, система здатна позначати пости як спам і пропаганду в реальному часі з середнім часом - кілька десятків секунд, а в разі сплесків трафіку - кілька хвилин. Недоліком цього варіанту, є потреба у великій кількості обчислювальних потужностей, у разі сплесків трафіку. Іншим варіантому є масштабування системи використовуючи більші (наприклад, 1 година) вікна агрегації. Перевагою другого варіанту є те, що він потребує менших обчислювальних потужностей. Недоліком те, що йде більше часу на маркування постаб як спам чи пропаганда.

3.6 Моніторинг і спостережуваність даної системи.

Тонкощі розподілених систем, які визначаються їхньою мережею взаємопов’язаних компонентів, розкиданих у різних місцях, вимагають передових інструментів для ефективного моніторингу та спостереження. Розгортання надійних рішень, таких як Grafana та OpenSearch, має важливе значення для підтримки працездатності, ефективності та надійності системи [158]. Grafana, провідна платформа візуалізації та аналітики з відкритим кодом, надає низку функцій для аналізу та представлення даних часових рядів. Він виділяється своєю здатністю збирати та змінювати дані з безлічі джерел, що робить його найкращим вибором для завдань спостереження.

Водночас OpenSearch, який розвинувся на основі відомого проекту Elasticsearch, пропонує потужний пошуковий і аналітичний пакет, відомий своїми можливостями швидкого збору даних, індексування та аналізу. Він діє як величезна база даних для журналів, показників і додаткових операційних даних із розподілених систем. Grafana використовує OpenSearch як цінне джерело даних, застосовуючи розширені запити, щоб виявити ідеї, закономірності та порушення в зібраних даних. Завдяки настроюваним інформаційним панелям Grafana користувачі отримують графічний огляд системних умов, що допомагає приймати

рішення на основі даних. Крім того, система сповіщень Grafana надає миттєві попередження, коли виконуються заздалегідь визначені умови або виникають певні інциденти, сприяючи швидкому реагуванню на потенційні проблеми [159].

Ця комбінована стратегія пропонує численні переваги, включаючи масштабованість, адаптивність, миттєвий аналіз і підтримку спільноти. І Grafana, і OpenSearch створені для роботи з великомасштабними проектами, пропонуючи гнучкість, необхідну для адаптації до зростаючих обсягів даних і зростаючої складності системи. Їхня здатність відстежувати низку показників, від продуктивності апаратного забезпечення до журналів додатків, забезпечує комплексне уявлення про розподілену систему. Здатність виконувати аналіз в реальному часі допомагає швидко виявляти та вирішувати проблеми. Крім того, активні спільноти з відкритим кодом для обох інструментів надають цінні ресурси, плагіни та знання, покращуючи систему моніторингу [160].

Таким чином, використання Grafana та OpenSearch представляє потужний підхід до управління складністю розподілених систем. Їх інтеграція покращує нагляд за системою, швидке виявлення проблем і попереджувальне усунення несправностей, забезпечуючи цілісність і ефективність роботи систем [161].

У даній системі соціальної мережі Opensearch буде логувати основні події системи, а також, будь які помилки, та нотифікувати власника системи на електрону скриньку. Також за допомогою Grafana, буде встановлено метрики, такі як кількість часу від запиту користувача, до отриманням ним результату. Також будуть логуватися метрики детальних запитів до бази даних, коли вони займать час довший від очікуваного.

3.7 Удосконалення швидкодії отриманої системи за допомогою кешування та мережі доставки контенту

3.7.1 Кешування

У сфері високопродуктивних розподілених архітектур стратегічне використання кешування даних у пам'яті є наріжним каменем оптимізації,

пристосованим для вирішення проблем продуктивності та масштабованості, притаманних великим розподіленим системам. Цей підхід використовує швидкодію пам'яті з довільним доступом (RAM) для створення проміжного шару для зберігання даних [162]. Такий рівень ефективно зберігає дані, що часто запитуються, або результати обчислень, тим самим запобігаючи потенційним затримкам, що виникають через повільнішу швидкість доступу до рішень постійного зберігання даних.

Кеш-пам'ять вирізняється тим, що пропонує значно менші затримки доступу до даних на відміну від традиційних дискових сховищ, ефективно мінімізуючи системний час відгуку. Ця перевага особливо важлива в умовах інтенсивного операційного навантаження. Виконуючи роль витісняючого буфера, механізм кешування поглинає значну частину операцій читання, тим самим зменшуючи навантаження на основні сховища даних. Такий перерозподіл робочого навантаження підвищує загальну пропускну здатність і швидкість реакції розподіленої системи [163]. Крім того, архітектура розподіленого кешування в пам'яті, що охоплює декілька вузлів, забезпечує безперебійний доступ до даних, полегшуючи при цьому масштабування системи шляхом простого додавання вузлів. Крім того, завдяки реплікації кешованих даних на різних вузлах, система набуває певного рівня надмірності, що захищає від перебоїв у роботі навіть у випадку відмови вузла, підвищуючи тим самим відмовостійкість системи.

Корисність кешування в пам'яті поширюється на кілька ключових областей, таких як управління інформацією про сеанси користувача для зменшення взаємодії з базами даних для персоналізованих сервісів і кешування відповідей від зовнішніх API для прискорення доступу до повторюваних запитів. Крім того, воно відіграє важливу роль у розвантаженні результатів складних або часто виконуваних обчислень від основного обчислювального навантаження [164]. Контраст між локальним кешуванням, яке приносить користь одному екземпляру програми, і розподіленим кешуванням, яке поширює переваги на всю систему за допомогою таких технологій, як Redis, Memcached або Hazelcast, підкреслює універсальність стратегій кешування в підвищенні продуктивності системи.

Враховуючи обмежений обсяг пам'яті, розгортання політик витіснення кешу, таких як Least Recently Used (LRU) або Time-to-Live (TTL), стає обов'язковим для ефективного управління життєвим циклом збережених даних [165]. Забезпечення узгодженості даних у розподіленому кеші вимагає надійних механізмів для своєчасного розповсюдження та синхронізації оновлень, щоб запобігти ризику обслуговування застарілих даних [166]. Крім того, вибір між стратегіями кешування з прямим та зворотним записом є критично важливим рішенням, яке балансує між необхідністю забезпечення узгодженості даних та оптимізацією продуктивності, визначаючи, як оновлення в кеші відображаються у внутрішній системі зберігання даних.

У даній системі соціальній мережі будемо кешувати дописи користувача у базі даних Редіс.

3.7.2 Мережа доставки контенту

У заплутаному просторі сучасного Інтернету, що характеризується інтенсивним обміном даними через складні мережі, мережі доставки контенту (англ. CDN) викристалізувалися як фундаментальна технологія. Вони вправно вирішують завдання поширення контенту, особливо мультимедійних ресурсів, до глобальної аудиторії, ефективно зменшуючи затримки та покращуючи користувацький досвід.

За своєю суттю CDN - це широка мережа проксі-серверів і центрів обробки даних, розподілених у різних географічних точках. Основним принципом роботи CDN є дублювання та кешування контенту з вихідних серверів у місцях, стратегічно близьких до кінцевих користувачів [167]. Такий архітектурний вибір значно зменшує відстані - як фізичні, так і через мережу - які повинні долати дані, тим самим помітно прискорюючи продуктивність.

Центральним елементом типової архітектури CDN є взаємодія між вихідними серверами - сховищами оригінального контенту, такого як код веб-сайту, зображення, відео тощо - і периферійними серверами, або точками

присутності (PoP), які ретельно розміщуються ближче до користувачів, щоб полегшити швидку доставку контенту.

Невід'ємною частиною цієї архітектури є використання систем балансування навантаження, які розподіляють вхідні запити користувачів по всій CDN, оптимізуючи використання ресурсів і запобігаючи перевантаженню серверів. Доповнюють цю систему складні за своєю конструкцією алгоритми маршрутизації, які визначають найефективніший шлях доставки контенту від периферійних серверів до пристроїв користувачів, беручи до уваги такі фактори, як завантаженість мережі та доступність серверів.

Робота CDN характеризується стратегічною реплікацією контенту між периферійними серверами в мережі. Коли користувач запитує контент, механізми маршрутизації CDN оцінюють місцезнаходження користувача, щоб спрямувати запит на найближчий периферійний сервер, який містить кешований контент. Якщо на найближчому периферійному сервері є запитуваний контент, він негайно надається користувачеві. І навпаки, якщо вміст відсутній, периферійний сервер отримує його з сервера-джерела або іншого периферійного сервера, доставляє користувачеві і кешує копію для майбутніх запитів.

Впровадження CDN дає безліч переваг, які є незамінними для веб-сайтів і додатків, що містять багато контенту. Серед них - значне покращення користувацького досвіду завдяки підвищенню продуктивності, що характеризується зменшенням затримок і підвищенням швидкості відгуку. Крім того, CDN пропонують масштабованість, яка дозволяє плавно керувати сплесками трафіку і стрибками попиту, забезпечуючи доступність контенту навіть під час пікових навантажень. Розподіл контенту між кількома серверами також підвищує надійність мережі, захищаючи її від збоїв і локальних перебоїв. З економічної точки зору, CDN є економічно вигідним рішенням, оскільки зменшує витрати на пропускну здатність і мінімізує вимоги до інфраструктури для власників вихідних серверів. Крім того, багато провайдерів CDN підвищують безпеку мережі за допомогою таких заходів, як захист від DDoS-атак та брандмауерів веб-додатків (WAF) [168].

CDN не є статичними утвореннями, вони постійно розвиваються, щоб адаптуватися до динамічних вимог інтернет-ландшафту. Ця еволюція позначена важливими тенденціями, такими як перехід до периферійних обчислень, які наближають обчислювальну потужність до користувача, тим самим дозволяючи запускати більш складні додатки і сервіси поруч з користувачами [169]. У сфері потокового відео CDN вдосконалюють свої архітектури, щоб краще відповідати специфічним вимогам високошвидкісного відеоконтенту в режимі реального часу. Більше того, інтеграція CDN з хмарними сервісами являє собою злиття технологій, пропонуючи цілісну, інтегровану інфраструктуру, яка розширює можливості та охоплення обох доменів.

Мережі доставки контенту (CDN) є основою швидкого та безперебійного доступу до Інтернету. Ось короткий огляд деяких важковаговиків галузі та їхніх переваг:

Amazon CloudFront: Частина AWS, CloudFront використовує мережу Amazon для високошвидкісної доставки контенту з низькою затримкою у величезних масштабах. Його інтеграція з AWS ідеально підходить для розробників, які вже інтегровані в цю екосистему [170].

Microsoft Azure CDN: побудована на базі глобальної мережі Microsoft, Azure CDN скорочує час завантаження та підвищує надійність. Це природний вибір для тих, хто використовує Azure, оскільки він забезпечує згуртоване середовище розробки.

Google Cloud CDN: Працюючи на базі інфраструктури Google, цей CDN гарантує швидкість і стабільність. Для тих, хто використовує хмарні сервіси Google, вона створює спрощене рішення для доставки контенту.

У даній системі соціальній мережі будемо використовувати мережу доставки контенту Amazon CloudFront. Там будуть зберігатися тексти і коментарі постів користувачів. Метадані ж про пости і коментарі зберігатимуться у базах даних.

Висновки до розділу 3

1. Створена модель ахітектури високонавантаженої розподіленої системи соціальної мережі. Модель архітектури використовує комбінацію масштабованих технології, такий, як балансувальники навантаження, брокери повідомлень, хмарні об’єктні сховища, що гарантує системі велику доступність. Також у даній архітектурі всі користувачі розбиваються на користувачів у яких велика кількість підписників і стандартних користувачів. Для підписників стандартних користувачів, коли вони створюють пост, система записує його в таблицю бази даних Касандра для всіх користувачів, на яких вони підписані. Це дозволяє швидко відповідати на запит генерування стрічки новин для користувача.

2. У модель системи інтегровано методи класифікації спаму та пропаганди, та метод пом’якшення ефекту бульбашки. Сервіси, які відповідають за виконання коду на основі даних методів, працюють через брокер повідомлень Кафка. Таким чином при напливі великої кількості користувачів кількість даних сервісів також буде масштабуватися пропроційно. Це дозволяє системі класифікувати повідомлення на спам і пропаганду в режимі близькому до реального часу.

РОЗДІЛ 4. ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ ОТРИМАНИХ МОДЕЛЕЙ ТА МЕТОДІВ

У четвертому розділі розроблено інструментальні засоби для класифікації текстів на спам і пропаганду, та кластеризації текстів; проведено порівняння результатів класифікації текстів на спам і пропаганду за допомогою різних моделей і методів машинного навчання і нейронних мереж на основі оцінок влучності, повноти, точності та f-міри; проведено експерименти для кластеризації текстів по дискусійних темах.

4.1 Розробка інструментальних засобів для класифікації тексту на спам і пропаганду

Основною мовою для експериментальних досліджень було вибрано Python версія 3.12.2 оскільки вона має багато корисних бібліотек та інструментів для тренування моделей машинного навчання і нейромереж та їх модифікації, побудови графіків, оцінки якостей моделей.

Опишемо основні бібліотеки пайтон, які будемо використовувати.

Використовується Keras бібліотека від TensorFlow – це високорівневий інтерфейс для нейромережевих архітектур, що працює як невід'ємний компонент TensorFlow, великої бібліотеки з відкритим вихідним кодом, призначеної для виконання чисельних обчислень. Спочатку задуманий Франсуа Шолле у 2015 році як незалежний фреймворк, Keras був включений до складу TensorFlow, де він функціонує як основний метод побудови та навчання моделей глибокого навчання. Основні класи бібліотеки Keras це layers (шари), models (моделі), optimizers (оптимізатори). Шари є основними елементами нейронних мереж в Keras, в той час, як моделі забезпечують структуру для організації шарів і полегшують функції навчання, оцінки та прогнозування. Keras підтримує різноманітні моделі, від простих послідовних моделей - лінійного стеку шарів - до більш складних функціональних моделей API, які дозволяють будувати складні архітектури. У

Keras можна знайти широкий вибір типів шарів, включаючи: fully connected layers, convolutional layers, pooling layers, recurrent layers. За домопогою Keras можна навчати моделі, як на CPU (центральному процесорі), так і на GPU (графічний процесор). За допомогою цієї бібліотеки, ми будемо модифікацію нейронної мережі LSTM для класифікації спаму та пропаганди.

Для тренування моделей машинного навчання (в тому числі ансамблевих) використовується бібліотека sklearn. Вона побудована на основі надійних наукових обчислювальних бібліотек NumPy, SciPy та matplotlib, надаючи повний набір інструментів для задач машинного навчання та статистичного моделювання. Sklearn здобув величезну популярність завдяки зручному інтерфейсу, великій документації та широкому спектру алгоритмів і утиліт, які він пропонує. В експериментальних дослідження будуть використовуватися обробка тексту за домогою sklearn CountVectorizer, та TfidfTransformer, які перетворюють текст в вектор, за допомогою bag-of-words та tf-idf описаних в другому розділі даної роботи. Також з sklearn.ensembe будуть використані ансамблеві моделі для класифікації, такі як RandomForestClassifier, AdaBoostClassifier, GradientBoostingClassifier, StackingClassifier.

Для попередньої обробки слів в текстах, будемо використовувати бібліотеку nltk (natural language toolkit) - це популярна бібліотека на мові Python для роботи з даними на природній мові. Вона широко використовується в дослідженнях і розробках для створення програм, які працюють з текстом і виконують такі завдання, як статистична обробка природної мови, та інших задач пов’язаних із штучним інтелектом і машинним навчанням.

У роботі буде використано WordNetLemmatizer – для лематизації, PorterStemmer – для стемінгу, nltk.download('punkt') – для токенізації і розбиття тексту на послідовність слів, nltk.download('wordnet') – лексичну базу даних для удосконалення лематизації за допомогою WordNetLemmatizer.

Для побудови векторних вбудувань слів використовуються класи бібліотеки transformers, такі як: AutoTokenizer та TFAutoModel. Параметр, який передається для ініціалізації класу – «distilroberta-base». Для побудови текстових вбудувань

використовується клас SentenceTransformers із бібліотеки sentence_transformers, на основі моделі paraphrase-distilroberta-base-v1. Векторні вбудування слів, та векторні вбудування текстів, мають однакову довжину – 768.

Для оцінки ми використовуємо такі міри (рис. 4.1), як точність (accuracy), влучність (precision), повнота (recall), та F-міра (F-score).

Рисунок 4.1 Обчислення влучності та повноти. Передрук з [171]

Формули для обчислення цих величин:

$$\text{Точність } (accuracy) = \frac{\text{ІП+ІН}}{\text{ІП+ХП+ІН+ХН}} \quad (4.1)$$

$$\text{Влучність } (precision) = \frac{\text{ІП}}{\text{ІП+ХП}} \quad (4.2)$$

$$\text{Повнота } (recall) = \frac{\text{ІП}}{\text{ІП+ХН}} \quad (4.3)$$

$$\text{F} - \text{міра } (F - score) = \frac{2}{\frac{1}{\text{Точність}} + \frac{1}{\text{Повнота}}} \quad (4.4)$$

Де ІП – істинно позитивні, ІН – істинно негативні, ХП – хибно позитивні, ХН – хибно негативні. ІН наприклад означає, що дане значення є негативним і модель визначила це значення правильно. ХП – що дане значення модель хибно визначила, як позитивне, насправді дане значення є негативним.

Для оцінки точності, влучності, повноти, та F – міри також будемо використовувати бібліотеку sklearn і відповідні метрики (metrics) precision_score, recall_score, f1_score, accuracy_score.

4.2 Тренування та тестування розроблених моделей та методів класифікації текстів на спам та пропаганду

Спочатку потестуємо корпус спаму описаний в другому розділі за допомогою класифіктору випадкових лісів. До всіх текстів застосуємо стемінг, лематизацію, приберемо стоп слова, та нормалізуємо слова до нижнього регістру. Перетворимо тексти у вектори множин слів, а відповідні числові значення визначимо за допомогою алгоритму TF-IDF.

Приклад результатів методів машинного навчання ансамблевих моделей з бібліотеки sklearn (рис. 4.2):

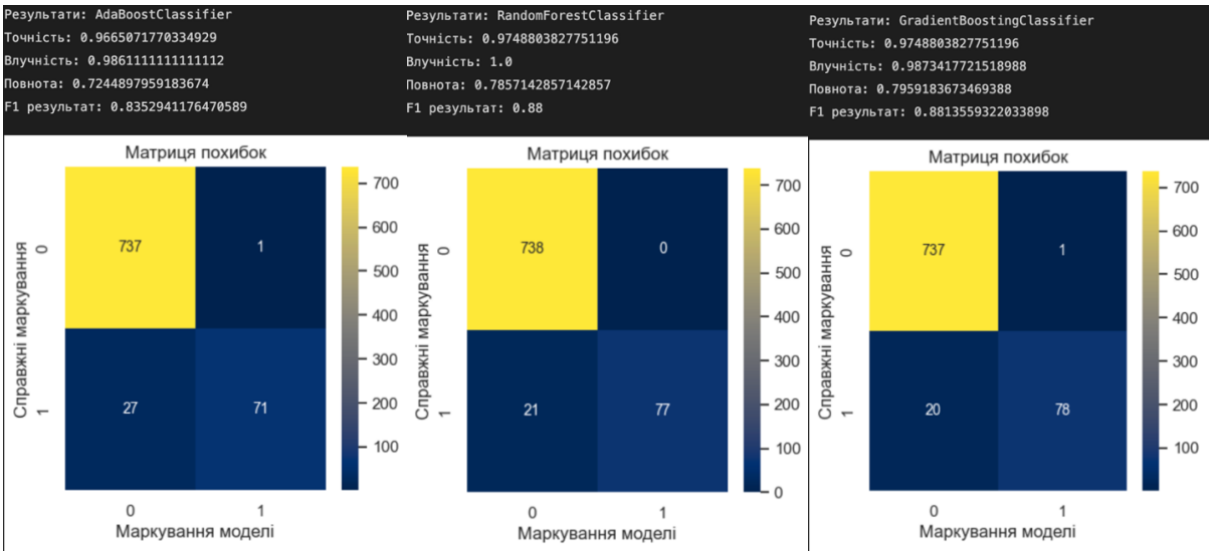


Рисунок 4.2 Результати класифікації тексту на спам за допомогою методів машинного навчання

Найточнішим на даний момент є класифікатор з градієнтним прискоренням. Точність найкращого методу класифікатора з градієнтним прискоренням 97,5 відсотки, а F міра 88,1 відсотки.

Наступною будемо навчати стандартну мережу ДКЧП та мережу НМКТуСН з другого розділу. На вхід моделі НМКТуСН буде подаватися вектор чисел на рис. 4.4.

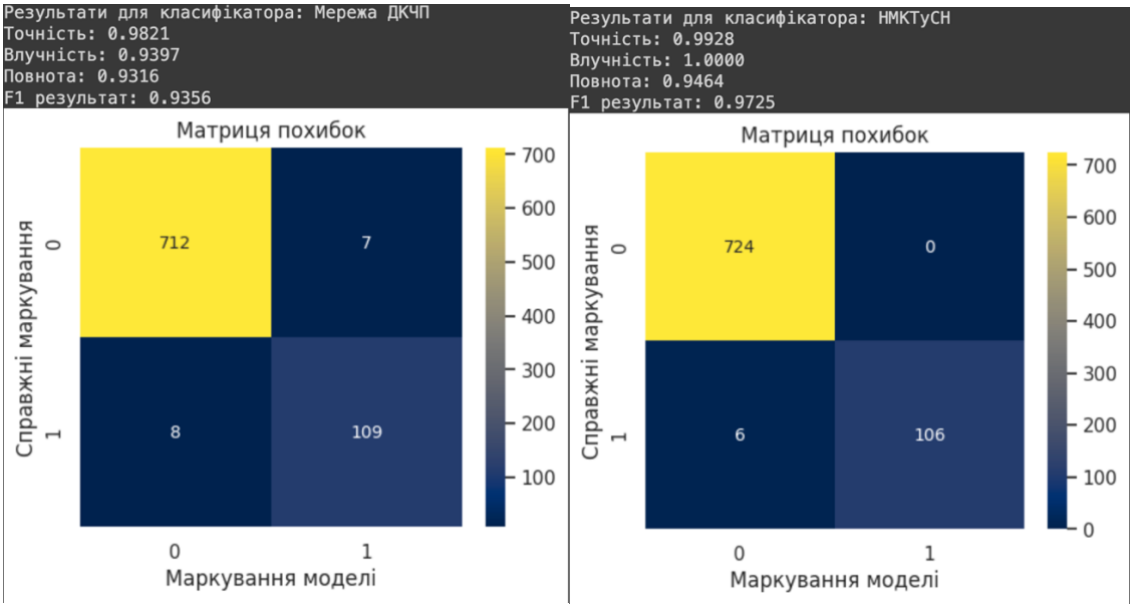


Рисунок 4.4 Результати класифікації спам текстів за допомогою ДКЧП та НМКТуСН

Бачимо, що результати модифікованої НМКТуСН моделі 99,28 відсотків для класифікації спаму, що є значно точнішим, за результати ансамблевих моделей на даному корпусі даних, та більш ніж на 1 відсоток точнішим від стандартної мережі ДКЧП.

Нейронна модель навчалася в 5 епох.

Результати loss функції і точності можна бачити на рисунку 4.5.

Приведемо також порівняння швидкості і точності класифікації текстів на спам.

Результат експериментів порівняння швидкості і точності моделей машинного навчання на нейронних мереж показав, що найвидшим є класифікатор

є з градієнтним прискоренням, який обробляє 1000 текстів за 0.41 секунду. Серед класифікаторів, які дають точність більшу 97.5 відсотків (більшу ніж класифікатор з градієнтним прискоренням), найшвидшим є НМКТуСН, за допомогою цього класифікатора досягається і найбільша точність класифікації.

```
Epoch 1/5
148/148 [==============================] - 13s 32ms/step - loss: 0.0839 - accuracy: 0.9713
Epoch 2/5
148/148 [==============================] - 6s 41ms/step - loss: 0.0442 - accuracy: 0.9867
Epoch 3/5
148/148 [==============================] - 5s 31ms/step - loss: 0.0321 - accuracy: 0.9905
Epoch 4/5
148/148 [==============================] - 5s 35ms/step - loss: 0.0245 - accuracy: 0.9932
Epoch 5/5
148/148 [==============================] - 7s 48ms/step - loss: 0.0139 - accuracy: 0.9960
<keras.src.callbacks.History at 0x7b2b18604ee0>
```

Рис. 4.5 Тренування моделі НМКТуСН для класифікації тексту на спам

Таблиця 4.1

Порівняння швидкості і точності класифікації текстів на спам

Класифікатор	Швидкість обробки 1000 текстів	Точність
Адабуст	0.64 сек.	96.65%
Випадковий ліс	0.50 сек.	97.48%
З градієнтним прискоренням	0.41 сек.	97.49%
РНН	1.6 сек.	97.96%
ВРВ	2.53 сек.	98.07%
ДКЧП	3.22 сек.	98.21%
НМКТуСН	1.10 сек.	99.28%

Це досягається за рахунок зменшення довжини вектора, який подається на вхід мережі довгої короткочасної пам’яті за рахунок об’єднуючого шару згорткової нейронної мережі.

Наступним кроком є навчання нейронної мережі на корпусі даних пропаганди. Обробку даних залишимо такою ж, як і у випадку зі спамом. Але

додамо також тексти із синтетичної заміної слів на синоніми, для того, щоб класифікатори могли шукати більш глибокі залежності в текстах.

Бінарну класифікацію тексту на пропаганду навчатимемо у 10 епох, тренування моделі показано на рис. 4.6.

```
Epoch 1/10
739/739 [==============================] - 32s 36ms/step - loss: 0.3630 - accuracy: 0.8548
Epoch 2/10
739/739 [==============================] - 27s 36ms/step - loss: 0.3237 - accuracy: 0.8702
Epoch 3/10
739/739 [==============================] - 26s 35ms/step - loss: 0.3001 - accuracy: 0.8830
Epoch 4/10
739/739 [==============================] - 26s 35ms/step - loss: 0.2729 - accuracy: 0.8927
Epoch 5/10
739/739 [==============================] - 23s 32ms/step - loss: 0.2376 - accuracy: 0.9063
Epoch 6/10
739/739 [==============================] - 27s 37ms/step - loss: 0.1965 - accuracy: 0.9229
Epoch 7/10
739/739 [==============================] - 26s 36ms/step - loss: 0.1528 - accuracy: 0.9404
Epoch 8/10
739/739 [==============================] - 26s 35ms/step - loss: 0.1154 - accuracy: 0.9558
Epoch 9/10
739/739 [==============================] - 26s 35ms/step - loss: 0.0806 - accuracy: 0.9700
Epoch 10/10
739/739 [==============================] - 24s 33ms/step - loss: 0.0625 - accuracy: 0.9765
<keras.src.callbacks.History at 0x7a6380230c40>
```

Рисунок 4.6 Навчання НМКТуСН для класифікації текстів на пропаганду

Порівняння результатів навчання НМКТуСН та ДКЧП для бінарної класифікації текстів на пропаганду наведено на рис. 4.7

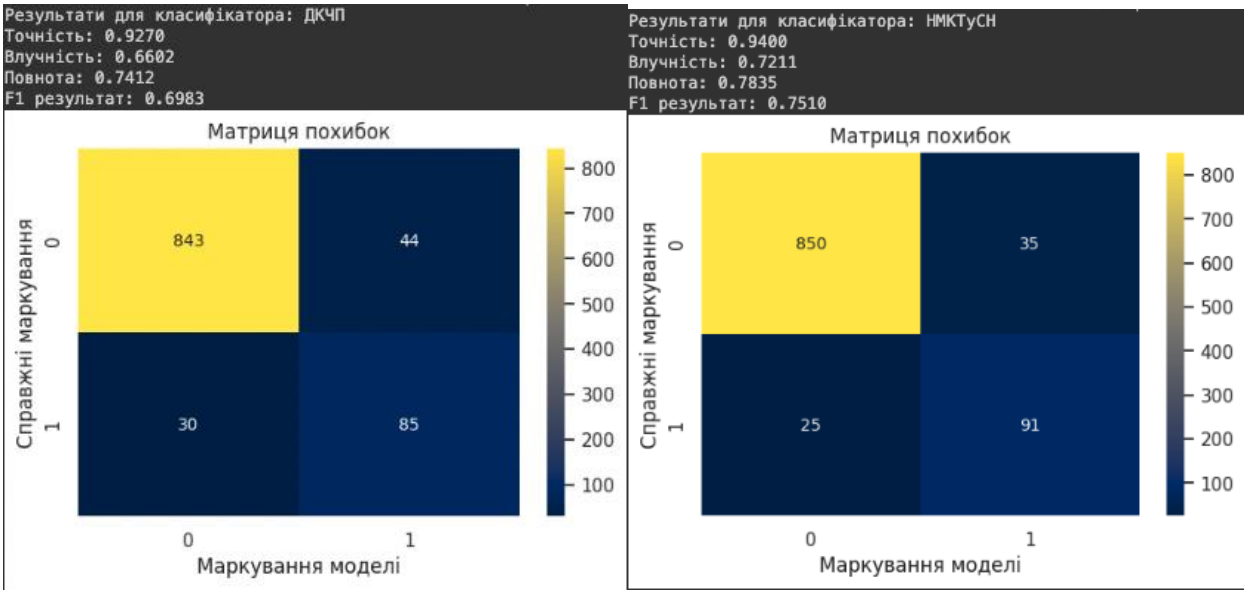


Рисунок 4.7. Порівняння результатів класифікації пропаганди для ДКЧП та НМКТуСН

Отримані результати показують, що класифікація на основі НМКТуСН більш ніж на 1 відсоток точніша за стандартну класифікацію на основі ДКЧП і рівна 94-ом відсоткам.

Таблиця 4.2

Порівняння швидкості класифікації текстів на пропаганду

Класифікатор	Швидкість обробки 1000 текстів	Точність
ВРВ	3.25 сек	92.1%
ДКЧП	4.11 сек	92.7%
НМКТуСН	1.49 сек	94%

На рис. 4.8. показано результати застосування моделі НМКТуСН до задачі багатокласової класифікації.

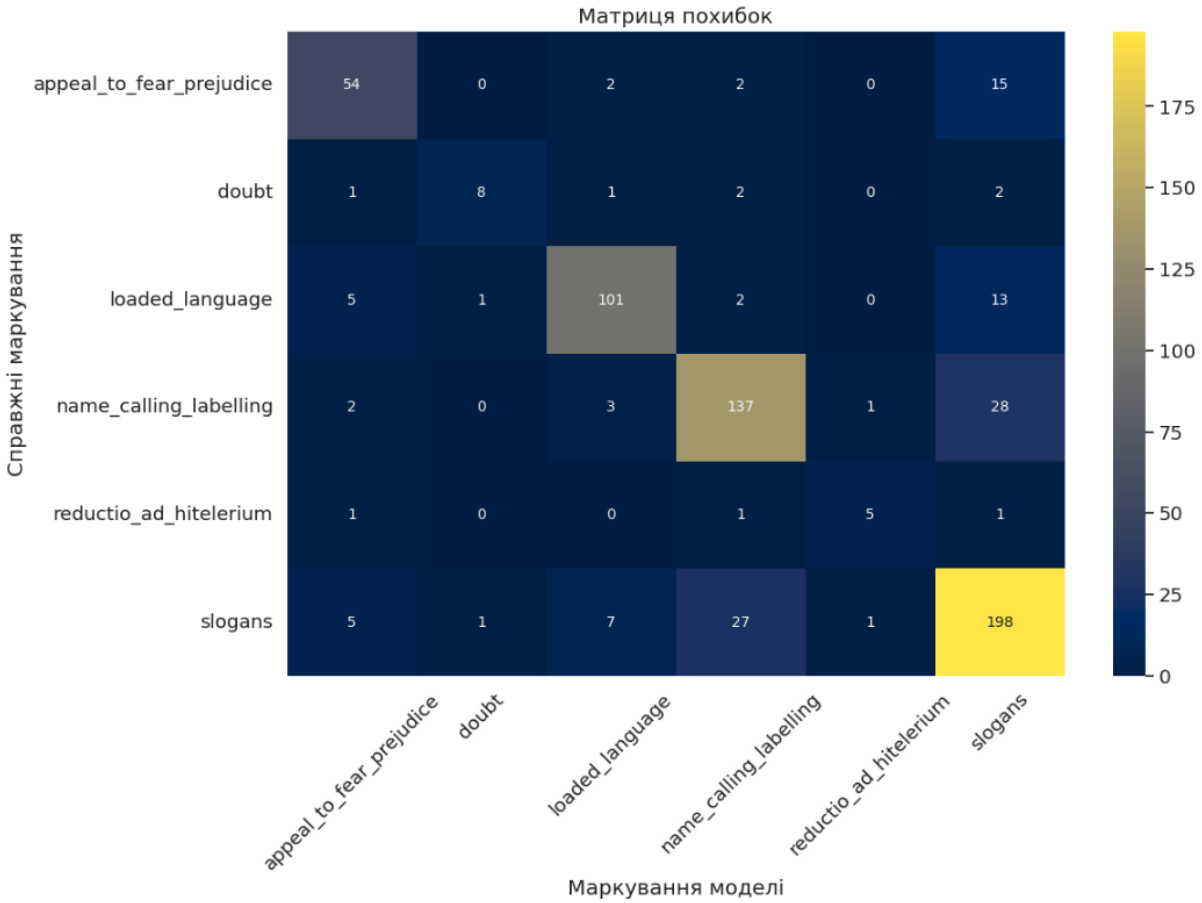


Рисунок 4.8. Результати класифікації різних типів пропаганди

4.3 Тестування класифікації і кластеризації дискусійних текстів

На вхід даної задачі подається текст, спочатку його класифікуємо за допомогою мережі НМКТуСН, щоб дізнатися, який тип тексту отримано на вхід.

Таким чином на вхід для кластеризації, приходять різні тексти на дискусійні теми і потрібно їх кластеризувати, щоб цю кластеризацію можна було використовувати для диверсифікованої стрічки новин (рис.4.9).

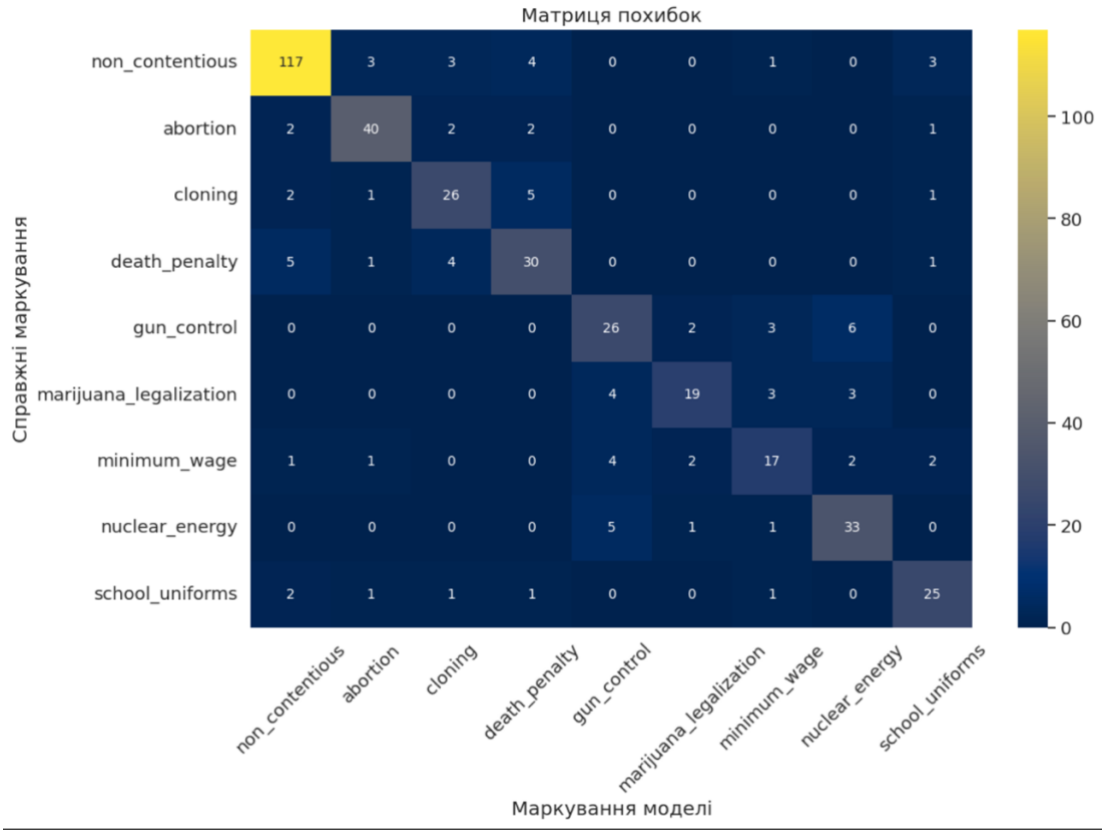


Рисунок 4.9. Результати для класифікації різних типів дискусійних тем

Для перетворення текстів в вектор чисел використовується Пайтон бібліотека sentence_transformers. Для цього використовується модель «paraphrase-distilroberta-base-v1», яка перетворює тексти в вектор дійсних чисел довжиною 768. Кожен елемент цього вектора має значення від мінус одиниці до одиниці. Сума квадратів всіх елементів вектора рівна одиниці, тобто вектор нормалізований. Таким чином, щоб знайти подібність між текстами на основі даної моделі достатньо поелементно перемножити вектори. Чим ближче значення буде до одиниці, тим ближчими є значення текстів.

Після перетворення текстів за допомогою sentence_transformers використовується алгоритм, KMeans для кластеризації текстів. Реалізацію алгоритму взято з бібліотеки scikit-learn.

Для графічної побудови отриманих кластерів використовується – МГК - метод головних компонент (англ. PCA – Principal Component Analysis). МГК – це статистичний метод, який широко використовується в машинному навчанні для зменшення розмірності зі збереженням якомога більшої дисперсії. Головні компоненти - це нові змінні, утворені методом МГК, які є ортогональними (тобто не корелюють одна з одною) і є лінійними комбінаціями вихідних змінних. Інструментально використовується бібліотека scikit-learn, модуль decomposition, клас PCA.

Також для графічної побудови використовується бібліотека plotly модуль express, для побудови інтерактивних графіків.

При наведенні на точку на графіку, з’явиться віконце, яке покаже, якому елементу відповідає дана точка (рис.4.10).

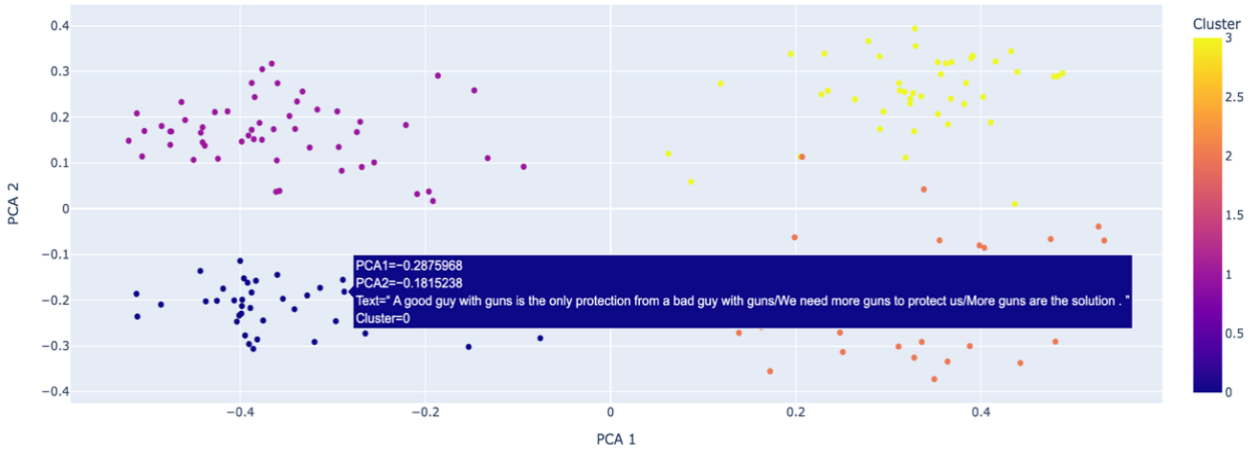


Рисунок 4.10 Результати кластеризації текстів щодо, контролю зброї

Приклад кластеризації на 4-ри кластери (рис.4.11), на основі алгоритму К-середніх, де позиції щодо володіння зброєю варіюються від сильно позитивної, до сильно негативної.

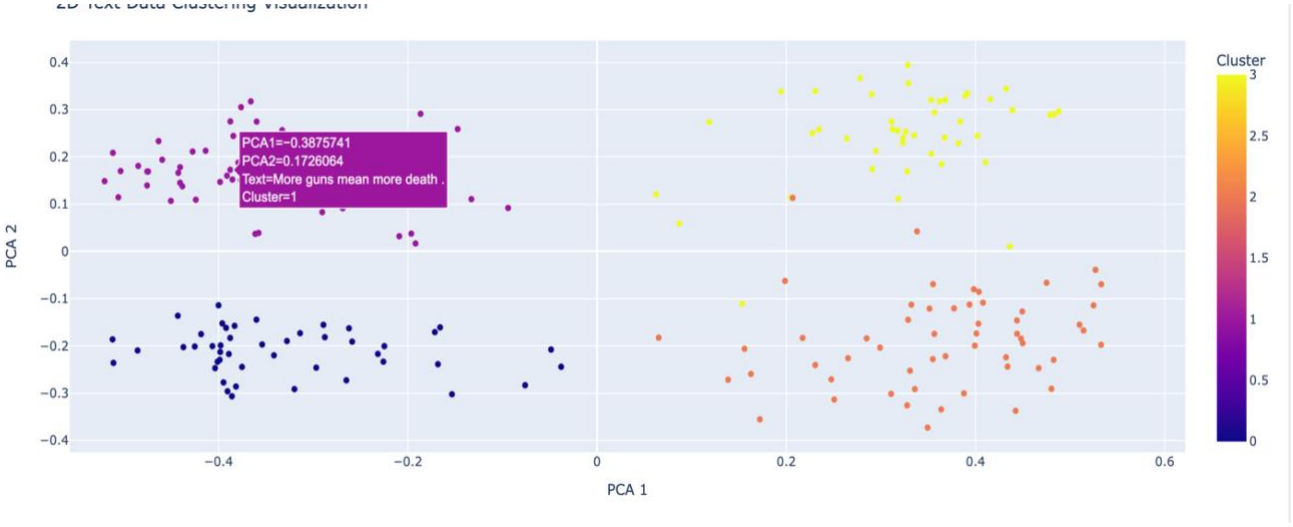


Рис. 4.11 Результати кластеризації текстів щодо, контролю зброї

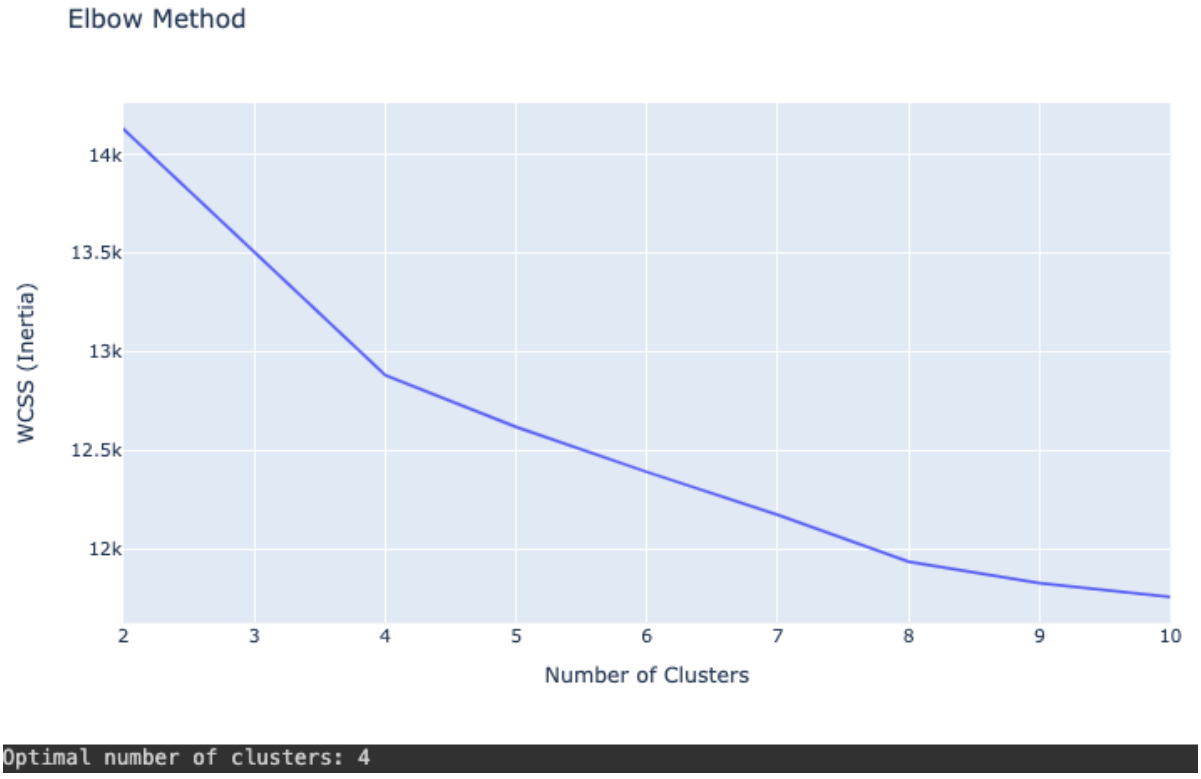


Рисунок 4.12 Використання методу ліктя, для знаходження оптимальної кількості кластерів

Приклади текстів з різних кластерів :

1. За строгий контроль над зброєю (кластер 2): «Повинен бути дуже суворий контроль над зброєю! Я так вважаю тому, що без суворого контролю над зброєю божевільні і злі люди можуть поранити або вбити багато людей.

Візьмемо, наприклад, стрілянину в школі Ньютона: божевільний хлопець взяв зброю своєї матері, вбив її, поїхав до школи, застрелив 20 дітей і шістьох дорослих, а потім наклав на себе руки. Якби був дуже суворий контроль над зброєю, цього б не сталося»

2. За контроль над зброєю (кластер 0) «Коли я говорю про контроль над зброєю, я маю на увазі обмеження, встановлені на вогнепальну зброю. Термін «контроль над зброєю» останнім часом в основному стосується обмежень, встановлених для штурмових гвинтівок. Коли я говорю про контроль над зброєю в цій дискусії, я маю на увазі більш пізнє визначення. Я вважаю, що людині не потрібна штурмова гвинтівка для самозахисту. Моє бачення полягає в тому, що все, що вам потрібно - це дробовик, щоб відбиватися від зловмисників».
3. Проти, контролю над зброєю (кластер 1): «Заборона штурмових гвинтівок. На штурмові гвинтівки припадає незначна кількість смертей від вогнепальної зброї. Однак, оскільки вони виглядають страхітливо, лобі з контролю над зброєю постійно намагається їх заборонити. Звинуватити зброю. У трагедіях, пов'язаних зі зброєю, часто звинувачують зброю. Хоча зброя була механізмом смерті, людина, яка її використовувала, в кінцевому підсумку вирішила натиснути на курок. Це підводить мене до розуміння, що це непрактично. Більшість смертей від вогнепальної зброї пов'язані з бандитськими розбірками. Банди не отримують зброю легально»
4. Сильно проти контролю над зброєю (кластер 3): «Контроль над зброєю не спрацював і ніколи не спрацює!! Щоразу, коли уряди приймають закони про контроль над зброєю, кількість злочинів, скоєних із застосуванням зброї, злітає до небес. Чому? Тому що злочинцям начхати на закони!»

Висновки до розділу 4

1. Порівняння результатів класифікації спаму і пропаганди за допомогою перетворення текстів у вектори чисел і подальшого використання методів машинного навчання і нейромереж показало високу ефективність застосування методу класифікації на основі НМКТуСН. Оцінки, які використовувалися це влучність, повнота, точність та f1-міра. Даний метод для класифікації спаму та пропаганди показує, на 1 відсоток вищу точність, за кращий результат за допомогою методів машиного навчання та нейронних мереж. Класифікатор на основі НМКТуСН є також найшвидшим, серед тих, які досягають високої точності.

2. Розроблена імплементація методу кластеризації, дозволяє ефективно кластеризувати тексти по дискусійних темах і отримувати групи текстів, які мають подібну думку, щодо дискусійної теми.

ВИСНОВКИ

1. Проаналізовано слабкі місця високонавантажених систем. Визначено, що пріоритетним для соціальних мереж є висока доступність та низька затримка відповіді на запит.
2. Визначено необхідність розробки методу, який підтримує баланс між точністю та швидкістю класифікації, на основі досліджень існуючих методів боротьби із спамом та пропаганди. З’ясовано, що ефект "бульбашки" виникає, коли у користувачів є різні позиції щодо дискусійної теми і їм мало показуються помірні за тональністю новини з іншої бульбашки.
3. Створено модель НМКТуСН основі модифікації гібриду згорткової нейронної мережі та двосторонньої мережі довгої короткочасної пам’яті, яка на відмінну від існуючих ефективно поєднує архітектурні рішення, оптимізовані гіперпараметри та натреновані наперед векторні вбудування слів, для того, щоб класифікувати текст ефективно та швидко.
4. Запропоновано метод на основі НМКТуСН, для класифікації текстів на спам та пропаганду. Точність класифікації за допомогою НМКТуСН спаму та пропаганди вища на 1 відсоток, ніж класифікація за допомогою ДКЧП, ВРВ, РНМ та інші методи машинного навчання.
5. Запропоновано метод на основі кластеризації текстів дискусійних тем та подальшого врахування цих результатів в нейронній мережі прямого поширення для пом’якшення ефекту «бульбашки».
6. Створено архітектурну модель розподіленої системи масштабованої соціальної мережі, яка використовуючи комбінації масштабованих технологій і натренованої НМКТуСН. Модель здатна класифікувати тексти і генерувати диверсифіковані стрічки новин в режиму реального часу.
7. Інтегровано методи нейронних мереж в модель соціальної мережі, таким чином, щоб модель не втратила свої властивості високої доступності та низької затримки відповіді на запит.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] D. Adisa, "Everything you need to know about social media algorithms," *Sprout Social. ,* 30 October 2023. URL: https://sproutsocial.com/insights/social-media-algorithms.

[2] I. Yurtseven, S. Bagriyanik and S. Ayvaz, "A Review of Spam Detection in Social Media," *International Conference on Computer Science and Engineering,* pp. 383-388, 2021. DOI: 10.1109/UBMK52708.2021.9558993..

[3] O. Stetsyk, P. Pasieka and B. Yeremenko, "Designing an Effective System Architecture for Detecting Propaganda and Spam in Social Media News Feed," *IEEE 4th KhPI Week on Advanced Technology (KhPIWeek),* 2023. DOI: https://doi.org/10.1109/khpiweek61412.2023.10312932.

[4] O. Stetsyk and S. Terenchuk, "Model development of the system for avoiding echo chambers in social networks.," *Management of Development of Complex Systems,* vol. 57, pp. 77-82, 2024. DOI: 10.32347/2412-9933.2024.57.77-82

[5] A. Hetler, "6 common social media privacy issues," *TechTarget,* 30 January 2023. URL: https://www.techtarget.com/whatis/feature/6-common-social-media-privacy-issues.

[6] E. Peebles, "Cyberbullying: Hiding behind the screen," *Paediatrics & Child Health,* vol. 19, no. 10, p. 527–528, 2014. DOI: https://doi.org/10.1093/pch/19.10.527.

[7] I. Pantic, "Online social networking and mental health," *Cyberpsychology, Behavior, and Social Networking,* vol. 17, no. 10, 2014. DOI: https://doi.org/10.1089/cyber.2014.0070.

[8] M. Iqbal, "Twitter Revenue and Usage Statistics," 2024. URL: https://www.businessofapps.com/data/twitter-statistics/.

[9] D. Sayce, "The Number of tweets per day in 2022," 2022. URL: https://www.dsayce.com/social-media/tweets-day/.

[10] B. Dean, "Facebook User & Growth Statistics," 2024. URL: https://backlinko.com/facebook-users.

[11] J. Guo and H. Chen, "How does multi-platform social media use lead to biased news engagement? examining the role of counter-attitudinal incidental exposure, cognitive elaboration, and network homogeneity," *Social Media + Society,* vol. 8, no. 4, p. 205630512211291, 2022. DOI: https://doi.org/10.1177/20563051221129140.

[12] A. Beck, "Platforms of Propaganda: How Social Media Sites Facilitate and Spread Disinformation," 2022. URL: https://www.mironline.ca/platforms-of-propaganda-how-social-media-sites-facilitate-and-spread-disinformation/.

[13] N. Agarwal, S. Al-khateeb, R. Galeano and R. Goolsby, "Examining the use of botnets and their evolution in propaganda dissemination," *Defence Strategic Communications,* vol. 2, no. 1, pp. 87-112, 2017. DOI: https://doi.org/10.30966/2018.riga.2.4.

[14] C. Shao, G. Ciampaglia and O. Varol, "The spread of low-credibility content by social bots," *Nature Communications,* vol. 9, no. 1, 2018. DOI: https://doi.org/10.1038/s41467-018-06930-7.

[15] D. Stacy, "Countries using organized social media manipulation campaigns," *Statista,* 2022. URL: https://www.statista.com/statistics/1023881/organized-social-media-manipulation-campaigns-worldwide/.

[16] K. Shu, A. Sliva, S. Wang, J. Tang and H. Liu, "“Fake News Detection on Social Media: A Data Mining Perspective”," *ACM SIGKDD Explorations Newsletter,* vol. 19, no. 1, pp. 22-36, June 2017.

[17] A. Khanday, Q. R. Khan and S. T. Rabani, "Detecting Textual Propaganda Using Machine Learning Techniques," *Baghdad Science Journal,* vol. 18, no. 1, pp. 199-209, December 2020.

[18] J. Kaur and D. M. Sabharwal, "Spam Detection in Online Social Networks Using Feed Forward Neural Network," *Independently published,* 3 October 2008.

[19] V. Kumar, A. Kumar, A. K. Singh and A. Pachauri, "Fake News Detection using Machine Learning and Natural Language Processing," *International Conference on Technological Advancements and Innovations (ICTAI),* 2021.

[20] A. Lada, M. Wang and T. & Yan, "How machine learning powers Facebook's News Feed ranking algorithm," *Core Infra, ML Applications,* 26 January 2021. URL: https://engineering.fb.com/2021/01/26/ml-applications/news-feed-ranking.

[21] N. Koumchatzky and A. Andryeyev, "Using deep learning at scale in Twitter's timelines.," *Twitter Engineering Blog,* 9 May 2017. URL: https://blog.twitter.com/engineering/en_us/topics/insights/2017/using-deep-learning-at-scale-in-twitters-timelines.

[22] A. Moehring, "News Feeds and User Engagement: Evidence from the Reddit News Tab.," *Massachusetts Institute of Technology (MIT) - Sloan School of Management,* pp. 4-6.

[23] F. Alatawi, L. Cheng, A. Tahir, M. Karami, B. B. T. Jiang and H. Liu, "A Survey on Echo Chambers on Social Media," *Description, Detection and Mitigation,* 2021. DOI: https://doi.org/10.48550/arXiv.2112.05084.

[24] M. Cinelli, G. De Francisci Morales, A. Galeazzi, W. Quattrociocchi and M. & Starnini, "The echo chamber effect on social media.," *Proceedings of the National Academy of Sciences,* vol. 118, no. 9, p. Article e2023301118, 2021. DOI: https://doi.org/10.1073/pnas.2023301118.

[25] K. Garimella, G. De Francisci Morales, A. Gionis and M. Mathioudakis, "Political Discourse on Social Media.," *In the 2018 World Wide Web Conference. ACM Press,* 2018.DOI: https://doi.org/10.1145/3178876.3186139.

[26] C. Ruiz and T. Nillson, "Disinformation and Echo Chambers: How Disinformation Circulates on Social Media Through Identity-Driven Controversies," *Journal of Public Policy and Marketing,* vol. 42, no. 1, 2022. DOI: https://doi.org/10.1177/07439156221103852.

[27] D. Choi, S. Chun, H. Oh, J. Han and T. Kwon, "Rumor Propagation is Amplified by Echo Chambers in Social Media," *Scientific Reports,* vol. 10, no. 1, 2020. DOI: https://doi.org/10.1038/s41598-019-57272-3.

[28] F. Baumann, Lorenz-Spreen, S. P., I. M. and M. Starnini, "Modeling Echo Chambers and Polarization Dynamics in Social Networks.," *Physical Review Letters,* vol. 124, no. 4, 2020. DOI: https://doi.org/10.1103/PhysRevLett.124.048301.

[29] S. Du and S. Gregory, "The Echo Chamber Effect in Twitter: does community polarization increase? In Studies in Computational Intelligence," *Springer International Publishing,* p. 373–378, 2016. DOI: https://doi.org/10.1007/978-3-319-50901-3_30.

[30] E. Colleoni, A. Rozza and A. Arvidsson, "Echo Chamber or Public Sphere? Predicting Political Orientation and Measuring Political Homophily in Twitter Using Big Data," *Journal of Communication,* 2014. DOI:10.1111/jcom.12084.

[31] P. Bibek and A. Bernstein, "Random Walks with Erasure: Diversifying Personalized Recommendations on Social and Information Networks," *Proceedings of the Web Conference 2021,* p. 2046–2057, 2021. DOI: https://doi.org/10.1145/3442381.3449970.

[32] F. Xia, J. Liu, H. Nie, L. Wan and X. Kong, "Random Walks: A Review of Algorithms and Applications," *Transactions on Emerging Topics in Computational Intelligence,* pp. 95-107, 2020. DOI: 10.1109/TETCI.2019.2952908.

[33] V. Serbanescu, K. Azadbakht and F. Boer, "A java-based distributed approach for generating large-scale social network graphs.," *Computer Communications*

and Networks, pp. 401-417, 2016. DOI: https://doi.org/10.1007/978-3-319-44881-7_19.

[34] Z. Shi, X. Zeng and Z. Yu, "A scalable and reconfigurable fault-tolerant distributed routing algorithm for nocs," *IEICE Transactions on Information and Systems,* pp. 1386-1397, 2011. DOI: https://doi.org/10.1587/transinf.e94.d.1386.

[35] M. Kleppmann, Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems, O'Reilly Media, 2017, pp. 25-43.

[36] S. Henning and W. Hasselbring, "How to measure scalability of distributed stream processing engines?," *Companion of the ACM/SPEC International Conference on Performance Engineering,* p. 85–88, 2021. DOI: https://doi.org/10.1145/3447545.3451190.

[37] T. Repantis and V. Kalogeraki, "Replica placement for high availability in distributed stream processing systems," *Proceedings of the Second International Conference on Distributed Event-Based Systems.,* p. 181–192, 2008. DOI: https://doi.org/10.1145/1385989.1386012.

[38] F. E. A. M. Hagit Attiya, "Limitations of Highly-Available Eventually-Consistent Data Stores," *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing,* p. 385–394, July 2015. DOI: https://doi.org/10.1145/2767386.2767419.

[39] N. Bhatt and A. Thakkar, "Big Data Stream Processing: Latency and Throughput," *International Journal of Advanced Science and Technology,* vol. 28, no. 16, pp. 1429-1435, 2019.

[40] D. Abadi, "Consistency tradeoffs in modern distributed database system design: cap is only part of the story," *Computer,* vol. 45, no. 2, pp. 37-42, 2012. DOI: https://doi.org/10.1109/mc.2012.33.

[41] R. Kune, P. Konugurthi, A. Agarwal, R. Chillarige and R. Buyya, The anatomy of big data computing, vol. vol. 46, Software: Practice and Experience, 2015; https://doi.org/10.1002/spe.2374, pp. 79-105.

[42] P. Bailis and A. and Ghodsi, Eventual consistency today: limitations, extensions, and beyond, Queue, 11(3), 2013, DOI: https://doi.org/10.1145/2460276.2462076, pp. 20-32.

[43] I. Fetai, A. Stiemer and H. Schuldt, "Quad: a quorum protocol for adaptive data management in the cloud", IEEE International Conference on Big Data (Big Data), 2017, DOI: https://doi.org/10.1109/bigdata.2017.8257952.

[44] M. Kleppmann, "A critique of the cap theorem", 2015, DOI: https://doi.org/10.48550/arxiv.1509.05393.

[45] P. Almeida and C. Baquero, ""Scalable eventually consistent counters over unreliable networks"," vol. vol. 32, pp. 69-89, 2017, DOI: https://doi.org/10.1007/s00446-017-0322-2.

[46] F. Muñoz-Escoí, R. Juan-Marín, J. García-Escrivá, J. Mendívil and J. Bernabéu-Aubán, ""Cap theorem: revision of its related consistency models"," *The Computer Journal,* vol. 62, pp. 943-960, 2019, DOI: https://doi.org/10.1093/comjnl/bxy142.

[47] E. Lee, A., S. Bateni, S. Lin, M. Lohstroh and C. Menard, Quantifying and generalizing the cap theorem, 2021, DOI: https://doi.org/10.48550/arxiv.2109.07771.

[48] A. Krechowicz, S. Deniziak and G. Łukawski, "Highly scalable distributed architecture for nosql datastore supporting strong consistency," *IEEE Access,* pp. 69027 - 69043, 2021. DOI: https://doi.org/10.1109/access.2021.3077680.

[49] J. Wofford, "Designing a scalable framework for declarative automation on distributed systems," 2021. DOI: https://doi.org/10.48550/arxiv.2104.13263.

[50] W. Vogels, "Eventually Consistent: Building reliable distributed systems at a worldwide scale demands trade-offs?between consistency and availability.,"

Queue, vol. 6, no. 6, p. 14–19, 2008. DOI: https://doi.org/10.1145/1466443.1466448.

[51] P. Bailis, A. Ghodsi, J. Hellerstein and I. Stoica, "Bolt-on causal consistency," *ACM SIGMOD International Conference on Management of Data,* p. 761–772, 2013. DOI: https://doi.org/10.1145/2463676.2465279.

[52] F. F. Moghaddam, A. Kanso and A. Gherbi, "Simple Leaderless Consistency Protocol," *IEEE International Conference on Cloud Engineering Workshop ,* 2016. DOI: https://doi.org/10.1109/ic2ew.2016.23.

[53] A. Gautreau, A. Barrat and M. Barthélemy, "Microdynamics in stationary complex networks," *Proceedings of the National Academy of Sciences, 106(22),* pp. 8847-8852, 2009. DOI: https://doi.org/10.1073/pnas.0811113106.

[54] F. Schneider, ""Implementing fault-tolerant services using the state machine approach: a tutorial"," *ACM Computing Surveys,* vol. 22, no. 4, pp. 299-319, 1990. DOI: https://doi.org/10.1145/98163.98167.

[55] Y. Du, B. Jiang and Y. Ma, ""Distributed fault-tolerant control for over-actuated multi-agent systems with uncertain perturbations using control allocation"," *International Journal of Robust and Nonlinear Control,* vol. 33, no. 11, pp. 6011-6030, 2023. DOI: https://doi.org/10.1002/rnc.6688.

[56] S. H. Zhu, J. Li and A. Wang, "Adaptive event-triggered asynchronous fault-tolerant control for stochastic systems with multiple-types of failures," *International Journal of Adaptive Control and Signal Processing,* vol. 36, no. 10, pp. 2401-2418, 2022. DOI: https://doi.org/10.1002/acs.3463.

[57] J. Liu, L. Wang and L. Wang, "Redundancy-based fault tolerance strategy for cloud computing systems," *IEEE Access, 6,* pp. 11667-11677, 2018.

[58] E. Elnozahy, N., L. Alvisi, Y. M. Wang and D. B. Johnson, "A survey of rollback-recovery protocols in message-passing systems," *ACM Computing Surveys (CSUR),* vol. 34, no. 3, pp. 375-408, 2002.

[59] A. S. Tanenbaum and M. Van Steen, "Distributed systems: principles and paradigms," *Pearson Prentice Hall,* 2007.

[60] M. Moreira, T. Jesus and J. Basilio, ""Polynomial time verification of decentralized diagnosability of discrete event systems"," *IEEE Transactions on Automatic Control,* vol. 56, no. 7, pp. 1679-1684, 2011. DOI: https://doi.org/10.1109/tac.2011.2124950.

[61] E. Chuah, A. Jhumka, M. Malek and N. Suri, ""A survey of log-correlation tools for failure diagnosis and prediction in cluster systems"," *IEEE Access,* vol. 10, pp. 33487-133503, 2022. DOI: https://doi.org/10.1109/access.2022.3231454.

[62] X. Geng, D. Ouyang and X. & Zhao, "Failure diagnosis for distributed stochastic discrete event systems.," *Mathematical Problems in Engineering,* pp. 1-12, 2017. DOI: https://doi.org/10.1155/2017/7680698.

[63] I. Mugarza, I. Yarza, I. Agirre, F. Lussiana and S. Botta, ""Safety and security concept for software updates on mixed-criticality systems"," *5th International Conference on System Reliability and Safety (ICSRS),* 2021. DOI: https://doi.org/10.1109/icsrs53853.2021.9660658.

[64] A. Younge, G. Laszewski, L. Wang, S. Lopez-Alarcon and W. Carithers, ""Efficient resource management for cloud computing environments"," *International Conference on Green Computing,* 2010. DOI: https://doi.org/10.1109/greencomp.2010.5598294.

[65] L. Cherkasova, K. Ozonat, N. Mi, J. Symons and E. Smirni, ""Automated anomaly detection and performance modeling of enterprise applications"," *ACM Transactions on Computer Systems,* vol. 27, no. 3, pp. 1-32, 2009. DOI: https://doi.org/10.1145/1629087.1629089.

[66] Wikipedia, "Stemming," URL: https://en.wikipedia.org/wiki/Stemming.

[67] Wikipedia, "Lemmatization," URL: https://en.wikipedia.org/wiki/Lemmatization.

[68] Z. Kastrati, A. Kurti and S. A, "WET: Word Embedding-Topic distribution vectors for MOOC video lectures dataset," *Elseiver,* 2020. DOI:10.1016/j.dib.2019.105090.

[69] D. Stewart, "Predicting project delivery rates using the Naive–Bayes classifier," *Journal of Software Maintenance and Evolution Research and Practice,* 2002. DOI:10.1002/smr.250.

[70] Langseth and Nielsen, "Classification using Hierarchical Naïve Bayes models," *Machine Learning,* 2006. DOI:10.1007/s10994-006-6136-2.

[71] "Attribute Weighted Naive Bayes Classifier," *Computers Materials & Continua,* 2022. DOI:10.32604/cmc.2022.022011.

[72] M. Patel, C. Andreescu, J. Price, K. Edelman, C. Reynolds and H. Aizenstein, "Machine learning approaches for integrating clinical and imaging features in late-life depression classification and response prediction," *International Journal of Geriatric Psychiatry,* vol. 30, no. 10, pp. 1056-1067, 2015. DOI: https://doi.org/10.1002/gps.4262.

[73] F. Liu, R. Yang, Y. Zhang, L. Qiao, S. Wang and Y. Y., "Distribution of olivine and pyroxene derived from clementine data in crater copernicus," *Journal of Earth Science,* vol. 22, no. 5, pp. 586-594, 2011. DOI: https://doi.org/10.1007/s12583-011-0209-2.

[74] P. Maan and M. Sharma, "Fuzzy improved decision tree approach for outlier detection in sms," *International Journal of Computer Applications,* vol. 119, no. 16, pp. 6-10, 2015. DOI: https://doi.org/10.5120/21149-4130.

[75] S. Jiang, H. Mao, Z. Ding and Y. Fu, "Deep decision tree transfer boosting," *IEEE Transactions on Neural Networks and Learning Systems,* vol. 31, no. 2, pp. 383-395, 2020. DOI: https://doi.org/10.1109/tnnls.2019.2901273.

[76] L. Breiman, "Untitled," *Machine Learnin,* vol. 45, no. 1, pp. 5-32, 2001. DOI: https://doi.org/10.1023/a:1010933404324.

[77] V. Svetnik, A. Liaw, C. Tong, J. Culberson, R. Sheridan and Feuston. B, "Random forest: a classification and regression tool for compound classification and qsar modeling," *Journal of Chemical Information and Computer Sciences,* vol. 43, no. 6, pp. 1947-1958, 2003. DOI: https://doi.org/10.1021/ci034160g.

[78] V. Arya, A. Almomani, A. Mishra, D. Peraković and M. Rafsanjani, "Email Spam Detection Using Naive Bayes and Random Forest Classifiers," *nternational Conference on Cyber Security, Privacy and Networking (ICSPN 2022) (Lecture Notes in Networks and Systems,* vol. 599, 2023. DOI: https://doi.org/10.1007/978-3-031-22018-0_31.

[79] "How to detect propaganda from social media?," *Exploitation of semantic analysis,* 2023. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10280574/.

[80] W. Noble, "What is a support vector machine," *Nature Biotechnology,* vol. 24, no. 12, pp. 1565-1567, 2006. DOI: https://doi.org/10.1038/nbt1206-1565.

[81] W. Huang, P. Lee, Y. Liu, A. Chiang and F. Lai, "Support vector machine prediction of obstructive sleep apnea in a large-scale chinese clinical sample," *Sleep,* vol. 43, no. 7, 2020. DOI:https://doi.org/10.1093/sleep/zsz295.

[82] E. Dada, J. Bassi, H. Chiroma, S. Abdulhamid, A. Adetunmbi and O. Ajibuwa, "Machine learning for email spam filtering: review, approaches and open research problems," *Heliyon,* vol. 5, no. 6, p. e01802, 2019. DOI: https://doi.org/10.1016/j.heliyon.2019.e01802.

[83] A. Chavda, K. Potika, F. D. Troia and M. Stamp, "Support Vector Machines for Image Spam Analysis," *SJSU ScholarWorks,* 2018. URL: https://scholarworks.sjsu.edu/cgi/viewcontent.cgi?article=1024&context=computer_sci_pub.

[84] A. Krizhevsky, I. Sutskever and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM,* vol. 60, no. 6, pp. 84-90, 2017. DOI: https://doi.org/10.1145/3065386.

[85] D. Zhou, "Universality of deep convolutional neural networks," *Applied and Computational Harmonic Analysis,* vol. 48, no. 2, pp. 787-794, 2020. DOI: https://doi.org/10.1016/j.acha.2019.06.004.

[86] A. Amidi and S. Amidi, "Convolutional Neural Networks cheatsheet," 2018. URL: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks.

[87] H. Saleh, A. Alharbi and S. Alsamhi, "Opcnn-fake: optimized convolutional neural network for fake news detection," *IEEE Access,* vol. 9, pp. 129471-129489, 2021. DOI: https://doi.org/10.1109/access.2021.3112806.

[88] M. Alhussein, K. Aurangzeb and S. Haider, "Hybrid cnn-lstm model for short-term individual household load forecasting," *IEEE Access,* vol. 8, pp. 180544-180557, 2020. DOI: https://doi.org/10.1109/access.2020.3028281.

[89] M. Malik, T. Imran and J. Mamdouh, "How to detect propaganda from social media? Exploitation of semantic and fine-tuned language models," *PeerJ Comput. Sci.,* vol. 9, p. e1248, February 2023. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10280574/.

[90] "Strengthening Cybersecurity: A Comparative Study of KNN and Random Forest for Spam Detection," *EngineeringEngineering (R0),* pp. 1-10, 2024. URL: https://link.springer.com/chapter/10.1007/978-981-99-9811-1_27.

[91] D. Lazar, "RNN in tensorflow," URL: https://medium.com/nabla-squared/creating-a-simple-rnn-from-scratch-with-tensorflow-8995a03c976d.

[92] S. Jadon, "Introduction to Different Activation Functions for Deep Learning," 2018. URL: https://medium.com/@shrutijadon/survey-on-activation-functions-for-deep-learning-9689331ba092.

[93] A. Afshine and A. Shervine, "Recurrent Neural Networks cheatsheet," 2020. URL: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks.

[94] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones and A. Gomez, "Attention Is All You Need," *Advances in Neural Information Processing Systems ,* pp. 5998-6008, 2017.

[95] T. Almeida and J. Hidalgo, "SMS Spam Collection," *UCI Machine Learning Repository,* 2012. DOI: https://doi.org/10.24432/C5CC84.

[96] A. Maarouf, D. Bar, D. Geissler and S. Feurriegel, "HQP: A Human-Annotated Dataset for Detecting Online Propaganda," 2023. DOI: https://arxiv.org/abs/2304.14931.

[97] Wikipedia, "East StratCom Task Force," 2024. URL: https://en.wikipedia.org/wiki/East_StratCom_Task_Force.

[98] A. Taparia, "Bidirectional LSTM in NLP," URL: https://www.geeksforgeeks.org/bidirectional-lstm-in-nlp/].

[99] T. Kanungo, D. Mount, N. Netanyahu, C. Piatko, R. Silverman and A. Wu, "An efficient k-means clustering algorithm: analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 24, no. 7, pp. 881-892, 2002. DOI: https://doi.org/10.1109/tpami.2002.1017616.

[100] M. Ahmed, R. Seraj and S. Islam, "The k-means algorithm: a comprehensive survey and performance evaluation," *Electronics,* vol. 9, no. 8, p. 1295, 2020. DOI: https://doi.org/10.3390/electronics9081295.

[101] A. Gupta and R. Katarya, "Pan-lda: a latent dirichlet allocation based novel feature extraction model for covid-19 data using machine learning," *Computers in Biology and Medicine,* vol. 138, p. 104920, 2021. DOI: https://doi.org/10.1016/j.compbiomed.2021.104920.

[102] K. Ravi and V. Ravi, "A survey on opinion mining and sentiment analysis: tasks, approaches and applications," *Knowledge-Based Systems,* vol. 84, pp. 14-46, 2015. DOI: https://doi.org/10.1016/j.knosys.2015.06.015.

[103] B. Enjolras and A. Salway, "Homophily and polarization on political twitter during the 2017 norwegian election," *Social Network Analysis and Mining,* vol. 13, no. 1, 2022. DOI: https://doi.org/10.1007/s13278-022-01018-z.

[104] S. Christian, M. Tristan, R. Pranav, B. Schiller and I. Gurevych, "UKP Sentential Argument Mining Corpus," 2018. URL: https://tudatalib.ulb.tu-darmstadt.de/handle/tudatalib/2345.

[105] H. Wachsmuth, M. Potthast and K. Al-Khatib, "Building an Argument Search Engine for the Web," *Proceedings of the 4th Workshop on Argument Mining,* p. 49–59, 2017. DOI: 10.18653/v1/W17-5106.

[106] M. Ahmed, R. Seraj and S. M. S. Islam, "The k-means Algorithm: A Comprehensive Survey and Performance Evaluation," *Electronics,* vol. 9, no. 8, p. 1295, 2020. DOI: https://doi.org/10.3390/electronics9081295.

[107] E. Umargono, J. E. Suseno and S. K. V. Gunawan, "K-Means Clustering Optimization Using the Elbow Method and Early Centroid Determination Based on Mean and Median Formula.," *n Proceedings of the 2nd International Seminar on Science and Technology (ISSTEC,* 2019. DOI: https://doi.org/10.2991/assehr.k.201010.019.

[108] A. Petukhova and N. Fachada, "MN-DS: A Multilabeled News Dataset for News Articles Hierarchical Classification," *MDPI,* 2023. DOI: https://doi.org/10.3390/data8050074.

[109] Tudorica, Bogdan, George, Bucur and Cristian, "A comparison between several NoSQL databases with comments and notes," *RoEduNet International Conference 10th Edition: Networking in Education and Research,* pp. 1-5, 2011.

[110] A. M. P. Lakshman, "Cassandra: a decentralized structured storage system," *ACM SIGOPS Operating Systems Review,* vol. 44, no. 2, pp. 36-39, 2010.

[111] C. Gyorodi, R. Gyorodi and G. Pecherle, "A comparative study: MongoDB vs. MySQL. 13th International Conference on Engineering of Modern Electric Systems (EMES)," pp. 1-5, 2105..

[112] R. Mazurenko and O. Stetsyk, "Intelligent Management of Traffic Flows in Large Cities," *International Workshop on Modern Experience for PhD students and Young Researchers,* pp. 40-41, 14-18 November 2022.

[113] D. Karger, A. Sherman, A. Berkheimer, B. Bogstad, R. Dhanidina and K. Iwamoto, "Web caching with consistent hashing," *Computer Networks,* vol. 31, no. 11-16, p. 1205–1206, 1999.

[114] Gyorodi, C, Gyorodi, R and G. Pecherle, "A comparative study: MongoDB vs. MySQL," *13th International Conference on Engineering of Modern Electric Systems (EMES),* pp. 1-5, 2015.

[115] C. A. Baron, "NoSQL Key-Value DBs Riak and Redis," *Database Systems Journal,* vol. 7, pp. 7-9, 2015..

[116] Ľ. Molčan, ""Time distributed data analysis by cosinor.online application"," 2019. DOI: https://doi.org/10.1101/805960.

[117] T. Brooks, A. Mrčela, N. Lahens, G. Paschos, T. Großer and C. Skarkeet, ""Nitecap: an exploratory circadian analysis web application"," *Journal of Biological Rhythms,* vol. 37, no. 1, pp. 43-52, 2021. DOI: https://doi.org/10.1177/07487304211054408.

[118] Y. Yulisman, H. Juliani, A. Muhaimin and A. & Zulkifli, "Aplikasi buku tamu undangan dengan menerapkan qr code berbasis web di wedding reception donys pelaminan.," *Jurnal Ilmu Komputer,* vol. 11, no. 2, pp. 69-79, 2022. DOI: https://doi.org/10.33060/jik/2022/vol11.iss2.281.

[119] A. G. McDonald, S. Boyce and K. F. Tipton, "Explorenz: the primary source of the iubmb enzyme list.," *Nucleic Acids Research,* 2009. DOI: https://doi.org/10.1093/nar/gkn582.

[120] T. Han and N. Ansari, "A traffic load balancing framework for software-defined radio access networks powered by hybrid energy sources," *IEEE/ACM Transactions on Networking,* vol. 24, no. 2, pp. 1038-1051, 2016. DOI: https://doi.org/10.1109/tnet.2015.2404576.

[121] X. Cao, S. Gao and L. Chen, ""Gossip-based load balance strategy in big data systems with hierarchical processors"," *Wireless Personal Communications,* vol. 98, no. 1, pp. 157-172, 2017. DOI: https://doi.org/10.1007/s11277-017-4861-4.

[122] S. Rabiu, H. Chan and S. Mohamad, "A cloud-based container microservices: a review on load-balancing and auto-scaling issues"," *International Journal of Data Science,* vol. 3, no. 2, pp. 80-92, 2022. DOI: https://doi.org/10.18517/ijods.3.2.80-92.2022.

[123] S. Afzal and G. Kavitha, ""Load balancing in cloud computing – a hierarchical taxonomical classification"," *Journal of Cloud Computing,* vol. 8, no. 1, 2019. DOI: https://doi.org/10.1186/s13677-019-0146-7.

[124] R. Malavika and M. Valarmathi, ""Load balancing based on closed loop control theory (lbbclct): a software defined networking (sdn) powered server load balancing system based on closed loop control theory"," *Concurrency and Computation: Practice and Experience,* vol. 34, no. 11, 2022. DOI: https://doi.org/10.1002/cpe.6854.

[125] W. Saber, W. Moussa, A. Ghuniem and R. Rizk, ""Hybrid load balance based on genetic algorithm in cloud environment"," *International Journal of Electrical and Computer Engineering (IJECE),* vol. 11, no. 3, p. 2477, 2021. DOI: https://doi.org/10.11591/ijece.v11i3.pp2477-2489.

[126] J. Dorsman, R. Mei and E. Winands, "Polling systems with batch service," *Or Spectrum,* vol. 34, no. 3, pp. 743-761, 2011. DOI: https://doi.org/10.1007/s00291-011-0275-y.

[127] B. Babcock, S. Babu, M. Datar, R. Motwani and J. Widom, "Models and issues in data stream systems," *Proceedings of the Twenty-First ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems - PODS '02,* 2002. DOI: https://doi.org/10.1145/543614.543615.

[128] J. Montiel, R. Mitchell, E. Frank, B. Pfahringer, T. Abdessalem and A. Bifet, "Adaptive xgboost for evolving data streams," 2020. DOI: https://doi.org/10.48550/arxiv.2005.07353.

[129] S. Hassan, R. Bahsoon and R. Buyya, "Systematic scalability analysis for microservices granularity adaptation design decisions," *Software: Practice and Experience,* vol. 52, no. 6, pp. 1378-1401, 2022. DOI: https://doi.org/10.1002/spe.3069.

[130] C. Kim and J. Ahn, "Gossip-based causal order delivery protocol respecting deadline constraints in publish/subscribe systems," *International Journal of Multimedia and Ubiquitous Engineering,* vol. 8, no. 6, pp. 245-256, 2013. DOI: https://doi.org/10.14257/ijmue.2013.8.6.25.

[131] L. Rodrigues, R. Baldoni, E. Anceaume and M. M. Raynal, "Deadline-constrained causal order," *Proceedings Third IEEE International Symposium on Object-Oriented Real-Time Distributed Computing,* 2000. DOI: https://doi.org/10.1109/isorc.2000.839535.

[132] W. Du, D. Navarro and F. Mieyeville, "Performance evaluation of ieee 802.15.4 sensor networks in industrial applications," *International Journal of Communication Systems,* vol. 28, no. 10, pp. 1657-1674, 2014. DOI: https://doi.org/10.1002/dac.2756.

[133] Y. Mansouri, V. Prokhorenko and M. Babar, "An automated implementation of hybrid cloud for performance evaluation of distributed databases," *Journal of*

Network and Computer Applications, vol. 167, p. 102740, 2020. DOI: https://doi.org/10.1016/j.jnca.2020.102740.

[134] M. Baddeley, U. Raza, A. Stanoev, G. Oikonomou, R. Nejabati and M. Sooriyabandaraet, "Atomic-sdn: is synchronous flooding the solution to software-defined networking in iot?," *IEEE Access,* vol. 7, pp. 96019-96034, 2019. DOI: https://doi.org/10.1109/access.2019.2920100.

[135] C. Wang, Q. Wang, K. Ren, N. Cao and W. Lou, "Toward secure and dependable storage services in cloud computing," *IEEE Transactions on Services Computing,* vol. 5, no. 2, pp. 220-232, 2012. DOI: https://doi.org/10.1109/tsc.2011.24.

[136] C. f. f. p. c. s. a. i. china, *Journal of data and information science,* vol. 1, no. 2, pp. 60-74, 2016 DOI: https://doi.org/10.20309/jdis.201614.

[137] M. Armbrust, T. Das, L. Sun, B. Yavuz, S. Zhu and M. Murthyet, "Delta lake," *Proceedings of the VLDB Endowment,* vol. 13, no. 12, pp. 3411-3424, 2020. DOI: https://doi.org/10.14778/3415478.3415560.

[138] Z. Daher and H. Hajjdiab, "Cloud storage comparative analysis amazon simple storage vs. microsoft azure blob storage," *International Journal of Machine Learning and Computing,* vol. 8, no. 1, pp. 85-89, 2018. DOI: https://doi.org/10.18178/ijmlc.2018.8.1.668.

[139] V. Raj and R. Sadam, "Performance and complexity comparison of service oriented architecture and microservices architecture," *International Journal of Communication Networks and Distributed Systems,* vol. 27, no. 1, p. 100, 2021. DOI: https://doi.org/10.1504/ijcnds.2021.116463.

[140] M. Pierce, S. Marru, L. Gunathilake, D. Wijeratne, R. Singh and C. Wimalasenaet, "Apache airavata: design and directions of a science gateway framework," *Concurrency and Computation: Practice and Experience,* vol. 27, no. 16, pp. 4282-4291, 2015, 2015. DOI: https://doi.org/10.1002/cpe.3534.

[141] S. Wang and Y. Liu, "Teragrid giscience gateway: bridging cyberinfrastructure and giscience," *International Journal of Geographical Information Science,* vol. 23, no. 5, pp. 631-656, 2009. DOI: https://doi.org/10.1080/13658810902754977.

[142] B. Gil and P. Trezentos, ""Impacts of data interchange formats on energy consumption and performance in smartphones"," *Proceedings of the 2011 Workshop on Open Source and Design of Communication,* 2011. DOI: https://doi.org/10.1145/2016716.2016718.

[143] B. Gil and P. Trezentos, "Impacts of data interchange formats on energy consumption and performance in smartphones," *Proceedings of the 2011 Workshop on Open Source and Design of Communication.,* 2011. DOI: https://doi.org/10.1145/2016716.2016718.

[144] J. C. Viotti and M. Kinderkhedia, "A benchmark of json-compatible binary serialization specifications," 2022. DOI: https://doi.org/10.48550/arxiv.2201.03051.

[145] B. Costa and P. Pires, ""Evaluating a representational state transfer (rest) architecture"," *Anais Do XXVIII Concurso De Teses E Dissertações (CTD 2015),* 2015. DOI: https://doi.org/10.5753/ctd.2015.10002.

[146] D. Wilusz and J. Rykowski, ""Orchestration of distributed hetero-geneous sensor networks and internet of things"," *Informatyka Ekonomiczna,* vol. 3, no. 33, 2014. DOI: https://doi.org/10.15611/ie.2014.3.09.

[147] K. Ogundeyi and C. Yinka-Banjo, "Websocket in real time application," *Nigerian Journal of Technology,* vol. 38, no. 4, p. 1010, 2019. DOI: https://doi.org/10.4314/njt.v38i4.26.

[148] J. Jones, J. Bradley and N. Sakimura, ""JSON Web Token (JWT) in Request for Comments"," *Internet Engineering Task Force (IETF),* May 2015. URL: https://www.rfc-editor.org/rfc/rfc7519.

[149] AboutSSL, "Top 6 Best SSL Certificate Authority List & SSL Certificate Brands," *AboutSSL,* 2022. URL: https://aboutssl.org/the-worlds-most-trusted-ssl-brands/.

[150] P. G. D. Group, "Partitioning," in PostgreSQL 13.4," *Documentation,* 2023. URL: https://www.postgresql.org/docs/current/ddl-partitioning.html.

[151] O. Stetsyk and S. Terenchuk, "Comparative analysis of NoSQL databases architecture.," *Management of Development of Complex Systems,* vol. 47, p. 78–82, 2021. DOI: dx.doi.org\10.32347/2412-9933.2021.47.78-82.

[152] Statista, "Facebook: number of daily active users worldwide 2011-2021," *Statista,* 2021. URL: https://www.statista.com/statistics/346167/facebook-global-dau/.

[153] Statista, "Monetizable daily active Twitter users worldwide Q1 2017 to Q2 2021," *Statista,* 2021. URL: https://www.statista.com/statistics/970920/monetizable-daily-active-twitter-users-worldwide/.

[154] A. Lakshman and P. Malik, ""Cassandra: a decentralized structured storage system"," *ACM SIGOPS Operating Systems Review,* vol. 44, no. 2, p. 35–40, April 2010.

[155] J. L. Carlson, ""Scaling Redis," in Redis in Action, 1st ed., ch. 10 “Scaling redis”," *Stamford, CT: Manning Publications,* pp. 228-248, June 2013.

[156] V. John and X. Liu, "A Survey of Distributed Message Broker Queues," 3 April 2017. DOI: https://doi.org/10.48550/arXiv.1704.00411..

[157] " What is Azure Blob storage?," *Microsoft Azure,* 27 February 2024. URL: https://learn.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction.

[158] M. Poliakov, O. Stetsyk and B. Yeremenko, "Modeling of the System for the Electronic Estimation of Adolescents’ Special Abilities," *IEEE 4th*

International Conference on Smart Information Systems and Technologies (SIST), 15 - 17 May 2024, Astana, Kazakhstan..

[159] A. C. C. Ali, M. Heidarinejad and B. Stephens, ""Elemental: an open-source wireless hardware and software platform for building energy and indoor environmental monitoring and control"," *Sensors,* vol. 19, no. 18, p. 4017, 2019. DOI: https://doi.org/10.3390/s19184017.

[160] K. Alhamazani, R. Ranjan, K. Mitra, F. A. Rabhi, P. P. Jayaraman, S. U. Khan and V. Bhatnagar, "An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the-art," *Computing,* vol. 97, no. 4, pp. 357-377, 2014. DOI: https://doi.org/10.1007/s00607-014-0398-5.

[161] J. Achkoski and V. Trajkovik, ""Distributed system reliability in intelligence information system"," *Informatyka Ekonomiczna,* vol. 3, no. 33, 2014. DOI: https://doi.org/10.15611/ie.2014.3.01.

[162] U. Niesen and M. Maddah-Ali, "Coded caching with nonuniform demands," *IEEE Transactions on Information Theory,* vol. 63, no. 2, pp. 1146-1158, 2017. DOI: https://doi.org/10.1109/tit.2016.2639522.

[163] E. Piovano, H. Joudeh and B. Clerckx, "Generalized degrees of freedom of the symmetric cache-aided miso broadcast channel with partial csit," *IEEE Transactions on Information Theory,* vol. 65, no. 9, pp. 5799-5815, 2019. DOI: https://doi.org/10.1109/tit.2019.2914204.

[164] X. Yu, Y. Xia, A. Pavlo, D. Sánchez, L. Rudolph and S. Devadas, "Sundial," *Proceedings of the VLDB Endowment,* vol. 11, no. 10, pp. 1289-1302, 2018. DOI: https://doi.org/10.14778/3231751.3231763.

[165] M. Ji, A. Tulino, J. Llorca and G. Caire, "On the average performance of caching and coded multicasting with random demands," *International Symposium on Wireless Communications Systems (ISWCS),* 2014. DOI: https://doi.org/10.1109/iswcs.2014.6933485.

[166] V. Zyuban and P. Kogge, "Inherently lower-power high-performance superscalar architectures," *IEEE Transactions on Computers,* vol. 50, no. 3, pp. 268-285, 2001. DOI: https://doi.org/10.1109/12.910816.

[167] M. Calder, X. Fan, Z. Hu, E. Katz-Bassett, J. Heidemann and R. Govindan, "Mapping the expansion of google's serving infrastructure," *Proceedings of the 2013 Conference on Internet Measurement Conference,* 2013. DOI: https://doi.org/10.1145/2504730.2504754.

[168] G. Peng, "CDN: Content distribution network," 2004. URL: https://arxiv.org/abs/cs/0411069.

[169] D. Xu, S. Kulkarni, C. Rosenberg and H. Chai, "Analysis of a CDN–P2P hybrid architecture for cost-effective streaming media distribution," *Multimedia Systems,* vol. 11, pp. 383-399, 2006.

[170] S. Yi, A. Andrzejak and D. Kondo, "Monetary cost-aware checkpointing and migration on amazon cloud spot instances," *IEEE Transactions on Services Computing,* vol. 5, no. 4, pp. 512-524, 2011.

[171] Wikipedia, "Влучність та повнота," URL: https://uk.wikipedia.org/wiki/Влучність_та_повнота.

ДОДАТКИ

Додаток А

УКРАЇНА
ВИКОНАВЧИЙ ОРГАН КИЇВСЬКОЇ МІСЬКОЇ РАДИ
(КИЇВСЬКА МІСЬКА ДЕРЖАВНА АДМІНІСТРАЦІЯ)
ДЕПАРТАМЕНТ СУСПІЛЬНИХ КОМУНІКАЦІЙ
вул. Хрещатик, 50Б, м.Київ, 01001 тел.(044) 2350700, (044) 235 05 70
Контактний центр міста Києва (044)15 51 E-mail: communications@kyivcity.gov.uaКод ЄДРПОУ 25695762

АКТ

впровадження

результатів дисертаційного дослідження

Відділом організаційно-аналітичного забезпечення та контролю впроваджено метод класифікації текстів на пропаганду, запропонований Стециком О.А. в дисертаційному дослідженні «Інтелектуальна система обробки даних в високонавантажених розподілених системах соціальних мереж» для визначення стратегії пропаганди в соціальних мережах.

Перший заступник директора Департаменту **Дмитро РУБАН**

Додаток Б

УКРАЇНА
ВИКОНАВЧИЙ ОРГАН КИЇВСЬКОЇ МІСЬКОЇ РАДИ
(КИЇВСЬКА МІСЬКА ДЕРЖАВНА АДМІНІСТРАЦІЯ)
ДЕПАРТАМЕНТ СУСПІЛЬНИХ КОМУНІКАЦІЙ
вул. Хрещатик, 50Б, м.Київ, 01001 тел.(044) 2350700, (044) 235 05 70
Контактний центр міста Києва (044)15 51 E-mail: communications@kyivcity.gov.uaКод ЄДРПОУ 25695762

АКТ

впровадження

результатів дисертаційного дослідження

Відділом управління інформаційної політики та комунікацій впроваджено метод диверсифікації новин, запропонований Стециком О.А. в дисертаційному дослідженні «Інтелектуальна система обробки даних в високонавантажених розподілених системах соціальних мереж».

Перший заступник директора Департаменту **Дмитро РУБАН**

Додаток В

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
БУДІВНИЦТВА І АРХІТЕКТУРИ

ФАКУЛЬТЕТ АВТОМАТИЗАЦІЇ І ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

«27» лютого 2024 р.

ДОВІДКА
про впровадження результатів дослідження
на здобуття наукового ступеня доктора філософії
зі спеціальності 126 Інформаційні системи та технології

Запропонована в дисертаційній роботі Стецика Олексія Андрійовича модель ефективної інтелектуальної системи виявлення пропаганди і спаму в стрічках новин соціальних мереж є прикладом інноваційного проекту системи соціальних мереж, що здатна швидко обробляти, аналізувати і класифікувати великі обсяги текстової інформації в режимі часу, наближеного до реального.

Використані в роботі гібридні моделі штучного інтелекту при вирішенні задачі класифікації текстової інформації, алгоритми стійкого хешування і віртуальних вершин для сегментування даних та алгоритму Murmur3 для хешування ключа впроваджені в курс «Теорія алгоритмів» освітньої програми спеціальностей 126 – «Інформаційні системи і технології» і 126 – «Інформаційні системи і технології. Штучний Інтелект. Когнітивні технології» факультету автоматизації і інформаційних технологій Київського національного університету будівництва і архітектури в 2022-2023 роках.

Це надає можливість впроваджувати необхідні теоретичні знання та практичні навички фахівцям галузі інформаційних технологій при вирішенні задач цифрової трансформації і підтверджує актуальність і вагоме теоретичне і практичне значення у процесі підготовки майбутніх спеціалістів галузі.

В.о. декана,
доктор технічних наук, професор Олександр ТЕРЕНТЬЄВ

Додаток Г

Апробація матеріалів дисертації

TRANSBALTICA 2022: Transportation Science and Technology

September 15-16, 2022, Vilnius, Lithuania

Virtual Room 1: *Intelligent Vehicles and Infrastructure* *Aerospace Technologies* **Chairman: Viktor Skrickij** **Co-Chairman: Laurynas Šišovas** https://liedm.zoom.us/my/transbaltica004
Comprehensive vehicle safety diagnostics and management system (*Maksym Delembovskyi, Terenchuk Svitlana*)
Validating adverse weather influence on LiDAR with an outdoor rain simulator (*Michał Brzozowski, Krzysztof Parczewski*)
Conditions of effective application of energy-saving programs for the movement of heavy trucks on the highway (*Myroslav Oliskevych, Viktor Danchuk*)
Crack open / close effect on impedance based system of structural health monitoring (*Pavithra Nagaraj, Vitalis Pavelko*)
Influence of the ground effect on the precise landing of an unmanned aircraft (*Andrius Dubovas, Domantas Bručas*)
Intelligent Management of Traffic Flows in Large Cities (*Bohdan Yeremenko, Roman Mazurenko, Oleksii Stetsyk and Anatolii Buhrov*)
Development of a technology for monitoring passenger traffic in the context of intelligent transport systems (*Mykyta Volodarets, Igor Gritsuk, Sergii Pronin, Alona Yurzhenko*)
Improvement of methodology of calculation and assessment of transport and operational condition of airfield pavement (*Viktor Karpov, Oleksandr Stepanchuk, Oleksandr Dubyk, Oleksandr Rodchenko, Olegas Prentkovskis*)

Додаток Ґ

Апробація матеріалів дисертації

The joint Second Int. Workshops on Reliability Engineering and Computational Intelligence, Delft, The Netherlands, 13-15.11.2022
and ACeSYRI: Modern Experience for PhD students and Young Researchers, Žilina, Slovakia, 14-18.11.2022

Wednesday, November 16, 2022

09:00 – 13:00 The First ACeSYRI Section

Moderator: Dr. Martin Lukac Click here to join.

1. Aelita Saurbayeva and Lyudmila Tarshilova (WKATU - Zhangir Khan West Kazakhstan Agrarian-Technical University, Uralsk, Kazakhstan), Digitalization as a Factor in the Development of the Global Economy
2. Daniyara Galiyeva and Zamzagul Sultanova (WKATU - Zhangir Khan West Kazakhstan Agrarian-Technical University, Uralsk, Kazakhstan), Information Technology in Agriculture Sector
3. Štefan Melich, Matúš Verény and Ján Rabčan (UNIZA - Zilinska Univerzita v Ziline, Slovakia), The Classification of ECG signals
4. Guy Attia and Sidorenko Ludmila, Prognostical Risk Modelling of Lifestyle Factors in Genetic Forms of Obesity
5. Erez Arad, Roee Levinberg, Guy Attia, Ilan Davidov, Lital Haim, Ludmila Sidorenko, Ablatogenomics – a Computer-Modelled Approach in Treatment of Genetically Determined Atrial Fibrillation
6. Aneta Gabrisova (UNIZA - Zilinska Univerzita v Ziline, Slovakia), Online Application for Teaching Biomedical Informatics
7. Irene Khalina and Aleksander Usatov, Genetic Algorithms for Frequency Analysis in Breaking Substitution Ciphers Problem
8. Kaldybek Makhambetov (KISA - Kazakhstan Information Security Association, Almaty, Kazakhstan) Designing an AES Algorithm based on FPGA
9. Ainura Gumarova and Gaukhar Kamalova (WKATU - Zhangir Khan West Kazakhstan Agrarian-Technical University, Uralsk, Kazakhstan), Modern Frameworks for Web-Application Development
10. Paul Pasieka and Bohdan Yeremenko, Methods and Technologies of Object Recognition in Dynamic Flows
11. Matej Kopera and Ján Rabčan (UNIZA - Zilinska Univerzita v Ziline, Slovakia), Reliability Analysis based on Fuzzy Decision Trees
12. Roman Mazurenko and Oleksii Stetsyk, Intelligent Management of Traffic Flows in Large Cities

Додаток Д

Апробація матеріалів дисертації

2024 IEEE 4th International Conference on Smart Information Systems and Technologies (SIST)

ID 222	1) Arailym Tleubayeva, Astana IT University, Kazakhstan 2) Alina Mitroshina, Astana IT University, Kazakhstan 3) Alpar Arman, Astana IT University, Kazakhstan 4) Arystan Shokan, Astana IT University, Kazakhstan 5) Shaikhanova Aigul, L.N. Gumilyov Eurasian National University, Kazakhstan **Systemic approach to optimizing natural language processing technologies in Astana IT University's admissions process**
16:00	**Coffee break**
16:20 18:00	**SECTION 2. IT in Education and Research**
ID 246	1) Mariya Utarbayeva, Nazarbayev Intellectual School, Kazakhstan 2) Makpal Mukanova, Nazarbayev Intellectual School, Kazakhstan **Integrated Computer Network Security System: Intrusion Detection and Threat Prediction Using Machine Learning Algorithms**
ID 253	1) Mykyta Poliakov, Kyiv National University of Construction and Architecture, Ukraine 2) Oleksii Stetsyk, Kyiv National University of Construction and Architecture, Ukraine 3) Bogdan Yeremenko, Kyiv National University of Construction and Architecture, Ukraine **Modeling of the System for the Electronic Estimation of Adolescents' Special Abilities**
ID 264	1) Olha Bielienkova, Kyiv National University of Construction and Architecture, Ukraine 2) Tetiana Kishchenko, Kyiv National University of Construction and Architecture, Ukraine 3) Matsapura Olena, Kyiv National University of Construction and Architecture, Ukraine 4) Abay Aryn, Astana IT University, Kazakhstan 5) Galyna Ryzhakova, Kyiv National University of Construction and Architecture, Ukraine 6) Oleksiy Mostovenko, Kyiv National University of Construction and Architecture, Ukraine **Institutional measurement of structural characteristics of residential real estate markets using the method of cluster analysis**