НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

«На правах рукопису»
УДК _____004.8_____

До захисту допущено:
Завідувач кафедри
_____ Сергій СТІРЕНКО
«___»_____2024 р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-науковою програмою «Інженерія програмного забезпечення комп'ютерних та інформаційних систем» зі спеціальності 121 "Інженерія програмного забезпечення" на тему: «Методи глибинного навчання для виявлення елементів пропаганди у текстових даних»

Виконав (-ла):
студент (-ка) ІІ курсу, групи ІМ-21мн
Шахова Поліна Миколаївна _____

Керівник:
доц.,к.т.н., доцент
Волокита А.М.

_____

Консультант з нормоконтролю:
проф. каф. ОТ, д.т.н., професор
Жабін Валерій Іванович

_____

Рецензент:

_____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2024 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет *Інформатики та обчислювальної техніки*
(повна назва)

Кафедра *Обчислювальної техніки*
(повна назва)

Рівень вищої освіти – другий (магістерський)

Спеціальність *121. Інженерія програмного забезпечення*
(код і назва)

Освітньо-наукова програма *Інженерія програмного забезпечення комп'ютерних та інформаційних систем*
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Сергій СТІРЕНКО
(підпис)
«____» _____ 2024 р.

ЗАВДАННЯ
на магістерську дисертацію студентці
Шаховій Поліні Миколаївні
(прізвище, ім'я, по батькові)

1. Тема дисертації Методи глибинного навчання для виявлення елементів пропаганди у текстових даних

Науковий керівник дисертації Волокита Артем Миколайович, доцент, кандидат технічних наук
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «25» березня 2024 р. № 1445-с

2. Строк подання студентом дисертації 30 травня 2024

3. Об'єкт дослідження процес класифікації текстових даних за наявністю пропаганди

4. Предмет дослідження методи виявлення елементів пропаганди в текстових даних

5. Перелік завдань, які потрібно розробити: узагальнення та аналіз методів глибинного навчання для роботи з текстами, дослідження та опис процесу розпізнавання пропаганди, розробка способу виявлення елементів пропаганди на основі глибинного навчання, реалізація моделі глибинного навчання та аналіз отриманих результатів.

6. Консультанти розділів дисертації:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1.	доц., Волокита А.М.		
2.	доц., Волокита А.М.		
3.	доц., Волокита А.М.		
4.	доц., Волокита А.М.		

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів дисертації	Примітка
1.	Затвердження теми дослідження. Визначення предмету дослідження.	22.05.2023 – 01.10.2023	
2.	Дослідження існуючих рішень та проблем в області виявлення елементів пропаганди у текстах.	05.10.2023 – 10.03.2024	
3.	Дослідження датасетів для даної задачі. Збір та підготовка даних для навчання.	01.11.2023 – 15.04.2024	
4.	Формування архітектури моделі для виявлення пропаганди. Обгрунтування вибору архітектурних компонентів та їх конфігурації.	01.11.2023 – 20.04.2024	
5.	Реалізація моделі з використанням обраних технологій та бібліотек. Навчання моделі на підготовленому датасеті, налаштування гіперпараметрів.	01.11.2023 – 20.04.2024	
6.	Проведення експериментів та оцінка ефективності моделі.	01.11.2023 – 20.04.2024	
7.	Розробка рекомендацій щодо практичного застосування розробленої моделі та потенційних напрямків для подальшого вдосконалення.	01.11.2023 – 25.04.2024	
8.	Оформлення магістерської дисертації.	01.11.2023 – 20.04.2024	
9.	Захист	13.06.2024	

Студентка Поліна ШАХОВА _____
(підпис)

Науковий керівник дисертації Артем ВОЛОКИТА _____
(підпис)

РЕФЕРАТ

на магістерську дисертацію

виконану на тему: Методи глибинного навчання для виявлення елементів пропаганди у текстових даних

студенткою: Шаховою Поліною Миколаївною

Робота складається із вступу та чотирьох розділів. Загальний обсяг роботи: 70 аркушів основного тексту, 15 ілюстрацій, 2 таблиці. При підготовці використовувалася література з різних джерел.

Актуальність. З розвитком цифрових технологій та інтернету стрімко зростають і обсяги доступної інформації. Соціальні мережі, онлайн-ЗМІ та інші цифрові платформи стали основними джерелами новин та думок для багатьох людей. Однак, разом із збільшенням обсягів інформації, зростає і кількість фейкових новин, маніпуляцій та пропаганди, що поширюються через ці канали.

Пропаганда здатна впливати на думки, емоції та поведінку людей. Вона може використовуватися для просування ідеологій, політичних поглядів або комерційних інтересів. Ефективне виявлення та протидія пропаганді є важливими для захисту інформаційного простору та забезпечення доступу людей до надійної та неупередженої інформації. Це особливо важливо для забезпечення стабільності демократичних процесів, протидії інформаційним операціям та захисту національної безпеки в умовах активної інформаційної війни, яку нині веде росія проти України.

Таким чином, актуальність дослідження зумовлена необхідністю розуміння механізмів пропаганди для критичного сприйняття інформації і протидії маніпулятивним впливам. Розробка нових методів та вдосконалення існуючих підходів до автоматизованого виявлення пропаганди сприятиме зміцненню стійкості суспільства до інформаційних загроз та забезпеченню здорового інформаційного середовища.

Мета і завдання дослідження. Метою дослідження є розробка методу на основі глибинного навчання для визначення наявності пропаганди в текстових даних, як статті у ЗМІ та пости у соціальних мережах.

Для досягнення означеної мети магістерської роботи сформовано такі завдання:

- Вивчення та класифікація мовового вжитку, лінгвістичних конструкцій та структур, які можуть бути характерними для пропагандистських матеріалів.
- Узагальнення та аналіз методів глибинного навчання для роботи з текстами. Дослідження та опис процесу розпізнавання пропаганди.
- Розробка способу виявлення елементів пропаганди на основі глибинного навчання.
- Реалізація моделі глибинного навчання.
- Тестування розробленого методу з метою оцінки його ефективності та точності виявлення пропагандистського контенту.

Об'єкт дослідження – процес класифікації текстових даних за наявністю пропаганди.

Предмет дослідження – методи виявлення елементів пропаганди в текстових даних.

Методи досліджень. Для реалізації поставлених завдань використано методи глибинного навчання, зокрема рекурентні нейронні мережі, згорткові нейронні мережі, моделі трансформерів та механізми уваги. До навчального датасету застосовано методи обробки природної мови.

Наукова новизна отриманих результатів. Запропоновано метод виявлення пропаганди в текстових даних який базується на комбінації рекурентних нейронних мереж LSTM, згорткових нейронних мереж та архітектури трансформерів. Розроблений метод за рахунок імплементованих механізмів уваги на трьох різних рівнях обробки тексту та дозволяє враховувати як локальні, так і глобальні залежності, що підвищує точність виявлення пропагандистських технік.

Особистий внесок здобувача полягає в самостійному виконанні магістерського дослідження, в якому відображено авторський підхід та особисто отримані теоретичні й прикладні результати, що стосуються вирішення задачі виявлення пропаганди в текстових даних. Формулювання мети та завдань дослідження проводилось спільно з науковим керівником..

Практична цінність. Розроблений метод може бути використаний для автоматизованого аналізу та фільтрації текстового контенту в різних сферах, таких як соціальні мережі, новинні ресурси, політичні кампанії, щоб забезпечити більш достовірну інформацію для користувачів.

Метод може бути інтегрований в існуючі системи моніторингу та аналізу текстового контенту, що забезпечить більш надійний захист від пропагандистських впливів.

Аналіз механізмів уваги дозволяє зрозуміти, на які лінгвістичні особливості, ключові слова або фрази модель звертає увагу при виявленні пропаганди. Така інтерпретація може бути корисною для експертів у галузі комунікацій, журналістики та медіа, щоб краще розуміти техніки та методи пропаганди, які використовуються в текстах.

Отримані результати та висновки можуть бути використані як основа для подальших досліджень в області виявлення пропаганди та розробки нових методів боротьби з дезінформацією в текстових даних.

Ключові слова

Виявлення пропаганди, глибинні нейронні мережі, механізми уваги, LSTM, CNN, Transformers, обробка природної мови.

ABSTRACT

Topicality. With the advancement of digital technologies and the internet, the volume of accessible information is rapidly increasing. Social networks, online media, and other digital platforms have become the primary sources of news and opinions for many people. However, along with the increase in information volume, there is also a rise in fake news, manipulations, and propaganda spreading through these channels. Propaganda can influence people's opinions, emotions, and behaviors. It can be used to promote ideologies, political views, or commercial interests. Effective detection and counteraction of propaganda are crucial for protecting the information space and ensuring people's access to reliable and unbiased information. This is particularly important for ensuring the stability of democratic processes, counteracting information operations, and protecting national security amid the active information warfare that russia is currently waging against Ukraine.

The relevance of this research is driven by the need to understand the mechanisms of propaganda for critical information perception and counteracting manipulative influences. The development of new methods and improvement of existing approaches to automated propaganda detection will enhance society's resilience to information threats and ensure a healthy information environment.

The aim of the research is to develop a deep learning based method to determine the presence of propaganda in textual data, such as articles in the media and posts on social networks.

To achieve the stated purpose of the master's thesis, the following objectives have been formed:

Study and classification of language usage, linguistic constructions, and structures that may be characteristic of propaganda materials.

Generalization and analysis of deep learning methods for working with texts. Research and description of the process of recognizing propaganda.

Development of a method for detecting elements of propaganda based on deep learning.

Implementation of a deep learning model.

Testing the developed method to evaluate its effectiveness and accuracy in detecting propaganda content.

The object of the study is the process of classifying textual data for the presence of propaganda.

The subject of the study is methods for detecting propaganda elements in textual data.

Research Methods. To accomplish the tasks, deep learning methods were used, including recurrent neural networks, convolutional neural networks, transformer models, and attention mechanisms. Natural language processing techniques were applied to the training dataset.

Scientific Novelty of the Obtained Results. A method for detecting propaganda in textual data has been proposed, based on a combination of LSTM recurrent neural networks, convolutional neural networks, and transformer architecture. The developed method uses attention mechanisms at three different levels of text processing, allowing it to consider both local and global dependencies, which improves the accuracy of detecting propaganda techniques.

Personal Contribution of the Researcher. The researcher independently conducted the master's research, reflecting an author's approach and personally obtained theoretical and practical results related to solving the problem of detecting propaganda in textual data. The formulation of the research objective and tasks was carried out jointly with the scientific advisor.

Practical Value. The developed method can be used for automated analysis and filtering of textual content in various areas such as social networks, news resources, and political campaigns to provide users with more reliable information. The method can be integrated into existing systems for monitoring and analyzing textual content, providing more robust protection against propaganda influences.

The analysis of attention mechanisms helps to understand which linguistic features, keywords, or phrases the model focuses on when detecting propaganda. This interpretation can be useful for experts in communication, journalism, and media to better understand the techniques and methods of propaganda used in texts.

The obtained results and conclusions can serve as a foundation for further research in the area of propaganda detection and the development of new methods to combat disinformation in textual data.

Keywords. Propaganda detection, deep neural networks, attention mechanisms, LSTM, CNN, Transformers, natural language processing.

ЗМІСТ

ВСТУП

Розвиток інформаційних технологій спричинив революцію у способах спілкування – про це свідчить кількість інформації, яка нині генерується та публікується у засобах масової інформації та в соціальних мережах. В той же час, в епоху Big data інформаційно-комунікаційні системи дедалі більше стають інструментом для поширення дезінформації та пропаганди.

Поняття «пропаганда» розглядається як інформація, яка цілеспрямовано формується для підтримки заздалегідь визначеного порядку денного, передбачає використання психологічних та риторичних прийомів для запобігання критичному аналізу інформації, які зокрема включають використання логічних помилок, звернення до емоцій аудиторії.

Робота з пропагандою та екстрапольованими фактами була проблемою протягом тривалого часу. Пропаганда 21-го століття найчастіше характеризується психологічними операціями задля політичної вигоди. Крім того, удосконалення технологій призвело до автоматизації поширення пропагандистського контенту на таких платформах соціальних мереж, як Twitter, Facebook, Instagram. Такий контент складається зі змовницьких, сенсаційних і екстремістських статей, які майже не ґрунтуються на фактах і дослідженнях і можуть впливати на громадську думку.

Основне завдання елементів пропаганди залишатися непоміченими для споживачів контенту для досягнення максимального ефекту. Саме складність виявлення без належного аналізу робить їх потужним та небезпечним важелем впливу на соціум і доводить необхідність створення засобів інформаційно-психологічної безпеки, призначених для аналізу та перевірки контенту на наявність риторичних прийомів, лінгвістичних конструкцій та зворотів властивих пропаганді.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ЗАСАДИ ДОСЛІДЖЕННЯ ПРОПАГАНДИ. ОГЛЯД ІСНУЮЧИХ ПІДХОДІВ РОЗВ'ЯЗАННЯ ЗАДАЧІ КЛАСИФІКАЦІЇ ТЕКСТОВИХ ДАНИХ НА ПРЕДМЕТ ВИЯВЛЕННЯ ПРОПАГАНДИ

1.1. Поняття пропаганди

Згідно з визначенням IPA (The Institute for Propaganda Analysis), пропаганда – це "навмисна і систематична спроба вплинути на сприйняття і поведінку способами, які сприяють досягненню бажаних цілей пропагандиста". Вона присутня не лише в тоталітарних режимах минулого, але і в сучасному світі, особливо в епоху цифрових технологій та соціальних мереж.

Інформаційна революція та розвиток цифрових технологій створили сприятливе середовище для поширення пропаганди. В умовах перенасичення інформаційного простору люди переважно змушені швидко обробляти великі обсяги даних, що часто призводить до некритичного сприйняття повідомлень та активізації когнітивних упереджень, що робить їх вразливими до технік маніпулювання емоціями, експлуатації невпевненості та використання двозначності мови для просування певних нав'язаних ідей та формування потрібних установок.

Особливістю сучасної пропаганди є активне використання можливостей цифрових технологій, зокрема соціальних мереж та онлайн-платформ. Пропагандисти вдаються до створення ботів і фейкових профілів для штучного нагнітання потрібних настроїв, контролю над інформаційними потоками та безпосереднього впливу на думку користувачів. Ці методи дозволяють охопити широку аудиторію та створити ілюзію масової підтримки певних ідей чи позицій. На відміну від інших форм комунікації, пропаганда завжди має на меті вплив не на окрему особистість, а на певну соціальну групу чи суспільство в цілому, і, як правило, здійснюється не одним індивідом, а організованими групами, що мають

чітку структуру, ресурси та стратегію дій. Це дозволяє досягати синергетичного ефекту та забезпечувати масштабність і тривалість пропагандистських кампаній.

1.2. Огляд основних методів пропаганди

Для ефективної протидії пропаганді необхідне чітке розуміння її основних методів і технік. У даному дослідженні проаналізовано 18 ключових прийомів пропаганди, які лягли в основу розробленої моделі автоматизованої класифікації пропагандистських текстів. Розглянуті техніки відображають різноманітні способи маніпулювання громадською думкою - від емоційних аргументів до підміни понять і викривлення фактів.

Звернення до страху і упереджень (Appeal to fear-prejudice). Ця техніка базується на експлуатації існуючих фобій, тривог, стереотипів аудиторії для просування певних ідей. Замість раціональної аргументації, пропагандист апелює до емоцій, гіперболізуючи потенційні загрози і негативні наслідки. Механізм впливу ґрунтується на підсвідомому прагненні людини уникнути небезпеки і дискомфорту.

Повторення (Repetition). Багаторазове повторення тези, незалежно від її істинності, підвищує суб'єктивне сприйняття її достовірності. Ефект зумовлений особливостями людської пам'яті і процесу обробки інформації. Знайомі твердження сприймаються як більш правдиві і переконливі. Цей пропагандистський прийом активно використовується в рекламі і політичній комунікації.

Обзивання, навішування ярликів (Name-Calling, Labeling). Техніка полягає у використанні пейоративних назв, епітетів щодо опонента або ідеї для їх дискредитації. Вона підміняє об'єктивну характеристику суб'єктивною негативною оцінкою, впливаючи на емоційне сприйняття. Ярлики визначають ставлення аудиторії, блокуючи неупереджений аналіз фактів.

Хибна дилема (Black-and-White Fallacy). Прийом передбачає редукцію множини альтернатив до двох полярних точок зору. Він спрощує багатовимірну

реальність, ігноруючи спектр проміжних позицій. Штучна поляризація змушує аудиторію прийняти одну з крайнощів, відкидаючи компромісні варіанти. Техніка часто вживається для загострення суперечностей і розколу суспільства.

Відвернення уваги(Whataboutism). Використовується для ухилення від незручних питань або критики шляхом зміщення фокусу на дійсні або надумані провини іншої сторони та дозволяє уникнути предметної дискусії, переводячи її в русло взаємних звинувачень. Така аргументація є маніпулятивною, оскільки не спростовує початкову тезу і не пропонує конструктивного вирішення проблеми.

Порівняння з Гітлером (Reductio ad Hitlerum). Техніка передбачає проведення аналогій між опонентом або його ідеями і універсальними символами зла. Таким чином відбувається пряма дискредитація об'єкту порівняння, водночас з блокуванням раціональної оцінки аргументів. В той же час більшість подібних паралелей є поверхневими і маніпулятивними, оскільки ігнорують історичний і фактологічний контекст.

Звернення до авторитету (Appeal to Authority). Прийом полягає у використанні висловлювань або підтримки з боку визнаних авторитетних осіб чи інституцій для надання легітимності певній ідеї або позиції, підміняючи таким чином логічну аргументацію некритичним прийняттям думки певної особи в силу її статусу або популярності. Репутація експерта не завжди корелює з істинністю конкретного твердження, особливо у сферах поза його компетенцією.

Заплутування, навмисна неясність (Obfuscation, Intentional Vagueness, Confusion). Техніка передбачає використання абстрактних, неоднозначних, надмірно складних формулювань для приховування істини або ухилення від прямої відповіді та ставить за мету перешкодити чіткому розумінню ситуації і формуванню власної позиції. Аудиторія втрачає можливість критично оцінити дійсний стан речей.

Сумнів (Doubt). Прийом полягає у формуванні недовіри до достовірної інформації шляхом її безпідставного оскарження. Навіть за відсутності спростування, сама постановка питання про надійність фактів породжує

невизначеність. Техніка дозволяє дискредитувати науковий консенсус, офіційні дані, думку експертів без надання альтернативних доказів.

Стадний інстинкт (Bandwagon). Техніка апелює до конформізму і прагнення більшості людей бути частиною домінуючої групи. Аргумент будується не на змістовних доказах, а на твердженні “так думають всі” чи “ так роблять всі”. Страх соціальної ізоляції змушує погоджуватись із загальноприйнятою думкою, навіть всупереч особистим переконанням.

Гасла (Slogans). Прийом передбачає використання коротких, влучних і емоційно насичених висловів або кліше для просування ідей. Гасла спрощують комплексні концепції до легко відтворюваних формул, що призводить до викривлення суті питання, ігнорування важливих нюансів і деталей. Слогани апелюють до емоцій та упереджень аудиторії замість раціонального мислення.

Відволікаючий маневр (Red Herring). Техніка полягає у введенні в дискусію питань, не пов'язаних із основною темою, для відволікання уваги аудиторії. Маніпулятивна зміна фокусу дозволяє пропагандисту уникнути предметної відповіді, замовчати незручні факти. Обговорення другорядних деталей створює ілюзію змістовності, проте не вирішує сутність проблеми.

Надмірне спрощення причинно-наслідкових зв'язків (Causal Oversimplification). Прийом редукує комплексні соціальні явища до елементарних причинно-наслідкових моделей, ігноруючи множинність факторів впливу. Монокаузальне пояснення створює враження легкості вирішення проблеми, проте не відповідає складності реальних процесів. Техніка часто вживається для просування популістських гасел.

“Солом'яне опудало” (Straw Men). Маніпуляція полягає у свідомому перекрученні аргументів опонента з наступним спростуванням цієї викривленої позиції. Пропагандист приписує "супротивнику" слабкі або абсурдні доводи, які не відповідають реальним поглядам. Деконструкція цих псевдо-аргументів створює ілюзію перемоги в дискусії, проте справжня позиція залишається незачепленою.

Кліше, що зупиняють мислення (Thought-terminating Cliches). Прийом передбачає використання банальних фраз, максим, приказок з метою обірвати процес критичного осмислення ситуації. Ці висловлювання подаються як самоочевидні істини або прописні мудрості, які не потребують рефлексії. Техніка блокує подальший аналіз проблеми, закріплюючи status quo у свідомості.

Перебільшення, мінімізація (Exaggeration, Minimization). Маніпуляція масштабом подій, якостей, результатів використовується для створення потрібного враження. Гіперболізація позитивних аспектів власної позиції і применшення досягнень опонента формує викривлену інформаційну картину. Емоційне нашарування домінує над фактологічною точністю.

Розмахування прапором (Flag-Waving). Техніка експлуатує патріотичні почуття, національну ідентичність, символіку держави для генерування підтримки певного курсу. Будь-яке рішення подається як вияв відданості Батьківщині, а незгода прирівнюється до зради. В цій системі координат раціональна оцінка витісняється емоційною лояльністю і конформізмом.

Емоційно забарвлені слова (Loaded Language). Прийом полягає у насиченні повідомлення лексикою з потужним конотативним значенням. Добір слів з позитивними або негативними асоціаціями дозволяє маніпулювати сприйняттям інформації без зміни її денотативного змісту. Навішування ярликів визначає емоційну реакцію аудиторії навіть за нейтрального фактажу.

1.3. Аналіз існуючих рішень автоматичного розпізнавання пропаганди в текстах

Тема виявлення пропаганди зацікавила дослідників і науковців головним чином після Першої світової війни. Ретельний аналіз численних випадків пропаганди показав, що врахування психічних умов цільової аудиторії та використання публічних ЗМІ є основними чинниками поширення пропаганди. Аналіз пропаганди спочатку був сформульований як 10-етапний процес, від ідентифікації пропагандиста до оцінки впливу новин на цільову аудиторію, але

це виявилося виснажливим через велику кількість новин, які поширюються через різні соціальні медіа. Таким чином виникла необхідність автоматизувати аналіз контенту в пошуках пропаганди. Розвиток технологій уможливив використання методів штучного інтелекту у сфері обробки природної мови.

У [1] описано використання методів машинного навчання, як опорні векторні машини (Support Vector Machines), стохастичний градієнтний спуск (Stochastic Gradient Descent), посилення градієнта (Gradient Boosting), обмежені дерева рішень (Bounded Decision Trees) і випадкові ліси (Random Forests) для виявлення фейкових новин. Набір даних був отриманий із сигнальних ЗМІ та списку джерел із OpenSources.co, щоб передбачити, правдиві чи фальшиві статті.

Для виявлення пропагандистських новинних статей і зменшення впливу пропаганди на аудиторію у [2] було запропоновано першу загальнодоступну систему виявлення пропаганди під назвою proppy, яка в режимі реального часу відстежувала пропагандистські статті в онлайн-новинах. Система складається з чотирьох модулів, які включають пошук статей, ідентифікацію подій, дедуплікацію та обчислення індексу пропаганди.

Крім того, згорткові нейронні мережі (CNN), відомі в основному для завдань класифікації в комп'ютерному зорі, зрештою показали свою ефективність і для класифікації речень. Було виявлено, що одношарова CNN дає чудові результати в класифікації речень з точністю понад 85%. Зокрема, це можна прослідкувати у [3], де було застосовано кілька архітектур нейронних мереж, таких як мережа довготривалої короткочасної пам'яті (LSTM), ієрархічна двонаправлена LSTM (H-LSTM) і згорткова нейронна мережа (CNN), щоб класифікувати текст на пропаганду та не пропаганду. У роботі використовували різні моделі представлення слів, включаючи word2vec, GloVe та TF-IDF. Результати показали, що CNN з представленням word2vec перевершує інші моделі з точністю, що дорівнює 88,2%.

У [4] запропоновано комплексний підхід до визначення пропаганди на основі емоційного забарвлення текстів, який поєднує методи обробки природної мови (NLP), мультимодальний аналіз, машинне навчання та сентиментальний

аналіз. Методи NLP, такі як аналіз лінгвістичних патернів, виявлення аномалій та визначення емоційних тонів і сенсу слів, використовуються для розбору текстової інформації. Мультимодальний аналіз дозволяє паралельно аналізувати текст, зображення та відео з метою виявлення маніпуляцій або неузгодженостей у різних медіаформатах, залучаючи алгоритми обробки зображень та глибокого навчання. Методи машинного навчання застосовуються для навчання на великих наборах даних задля розпізнавання патернів дезінформації. Сентиментальний аналіз є основним методом для автоматичної класифікації тексту як позитивного, негативного або нейтрального на основі оцінювання емоційних тонів та розуміння смислу слів, що дозволяє визначати емоційне забарвлення текстів. Цей підхід забезпечує автоматизацію, швидкість та адаптивність, проте має недолік у точності класифікації емоційного забарвлення.

[5] пропонує ієрархічну модель на основі трансформера XLM-RoBERTa для виявлення та характеристики технік пропаганди в текстах. Цей підхід складається з трьох етапів:

1. Класифікація тексту як пропагандистського чи непропагандистського.
2. Визначення типу застосованої техніки пропаганди у тексті.
3. Присвоєння конкретних виявлених технік пропаганди до відповідних категорій або груп технік.

Для навчання моделі використовувався не лише набір даних DIPROMATS, але й додаткові набори даних SemEval'23, MBIC та BABE з метою забезпечення широкого охоплення різноманітних пропагандистських технік. Завдяки ієрархічній структурі та використанню потужної мовної моделі трансформера XLM-RoBERTa, підхід демонструє здатність виявляти пропаганду в текстах, класифікувати типи застосованих технік та групувати їх для подальшого аналізу.

[6] описує ансамблеву модель глибокого навчання для виявлення пропаганди в новинних статтях. Модель поєднує в собі BiLSTM, XGBoost та BERT. В якості ознак використовуються векторні представлення слів (GloVe embeddings), а також лексичні та афективні ознаки, отримані за допомогою

пакету AffectiveTweets. BiLSTM модель приймає два входи - векторні представлення речень і афективні ознаки. XGBoost використовує суму векторних представлень слів у реченні. Різні варіанти BERT моделей (Cased і Uncased) застосовуються окремо, а потім комбінуються. Фінальна ансамблева модель поєднує результати BiLSTM, XGBoost та BERT зважуванням їхніх прогнозів. Вона перевершила базову модель, досягнувши F1-оцінки 0.6112 на тестовому наборі даних. Однак, автори відмічають і певні недоліки запропонованого підходу, зокрема його відносну складність, обчислювальну затратність і недостатню інтерпретованість. Робота окреслює декілька перспективних напрямків для подальших досліджень, таких як розширення простору ознак, використання більш потужних моделей трансформерів і їхніх версій, дотренованих на релевантних даних, інтеграцію механізмів уваги, врахування ширшого контексту та застосування методів переносу знань.

У [7] представлено поглиблене розуміння методу навчання векторних представлень слів GloVe (Global Vectors for Word Representation). Даний метод поєднує переваги глобальної матричної факторизації та методів локального контекстного вікна. Ключовою особливістю GloVe є ефективне використання статистичної інформації шляхом навчання лише на ненульових елементах замість обробки всієї розрідженої матриці або окремих контекстних вікон у великому корпусі текстових даних. Такий підхід дозволяє моделі генерувати векторний простір слів із значущою підструктурою, як продемонстровано результатом точності 75%. Крім того, модель GloVe показала перевагу над спорідненими моделями у завданнях визначення семантичної подібності слів та розпізнавання іменованих сутностей.

У [8] запропоновано непараметричний підхід до автоматичного виявлення організованих груп користувачів, які поширюють пропаганду та дезінформацію в соціальній мережі Twitter. Автори зосередились на ідентифікації скоординованих дій груп пропагандистів, а запропонований метод базується на аналізі поведінкових патернів користувачів та текстового вмісту їхніх повідомлень. Основними етапами підходу є: попередня обробка та фільтрація

твітів, сегментація набору даних на часові проміжки, кластеризація твітів у кожному проміжку з використанням алгоритму k-середніх, ідентифікація груп користувачів, які часто потрапляють в один кластер, за допомогою алгоритму асоціативних правил AprioriTID, визначення найбільш впливових розповсюджувачів з використанням міри центральності PageRank, а також візуалізація та аналіз отриманої мережі розповсюдження. Автори висувають припущення щодо характерних ознак поведінки пропагандистів у соціальних мережах. Зокрема, передбачається, що вони діють у скоординованих групах, публікують дуже схожі повідомлення протягом короткого проміжку часу, роблять це з високою частотою, можуть використовувати декілька облікових записів, зосереджуються на політичній тематиці та узгоджуються з офіційною позицією влади. Ефективність запропонованого методу була продемонстрована на наборі даних з твітами, пов'язаними з військовими ударами у Сирії у вересні 2017 року. Виявлені підозрілі облікові записи були вручну перевірені дослідниками, що підтвердило високу точність підходу у ідентифікації організованих груп розповсюджувачів пропаганди. Серед переваг даного підходу можна відзначити його непараметричність, тобто відсутність необхідності у попередньому навчанні моделі на розмічених даних. Крім того, метод дозволяє виявляти не лише окремих розповсюджувачів, а й скоординовані групи, що діють узгоджено. Проте, даний підхід вимагає ручної валідації виявлених підозрілих облікових записів, що може бути трудомістким процесом. Також автори не надають оцінки повноти виявлення пропагандистських груп, тобто частки виявлених груп серед усіх наявних у даних.

Систему виявлення фейкових новин з використанням методів машинного навчання та глибоких нейронних мереж, зокрема моделей довгої короткочасної пам'яті (LSTM) детально описано у [9]. Автори здійснили ґрунтовний аналіз існуючих підходів, заснованих на техніках машинного та глибокого навчання, а також запропонували власну модель для вирішення цієї задачі. У роботі використано набір даних з платформи Kaggle, який містить новинні статті, розмічені відповідно до їх достовірності. Методологія дослідження включає

етапи попередньої обробки тексту, розбиття даних на навчальну та тестову вибірки, векторизації текстових даних, навчання та оцінки ефективності моделей. Серед методів машинного навчання, розглянутих у дослідженні, були дерева рішень, логістична регресія, випадковий ліс, наївний баєсів класифікатор та метод k найближчих сусідів. Найкращу точність на тестових даних продемонструвала модель логістичної регресії, досягнувши показника 91.22%. Запропонована авторами модель на основі архітектури LSTM показала суттєве покращення результатів у порівнянні з традиційними методами машинного навчання, досягнувши точності 94% на тестовій вибірці. Ключовою перевагою використання моделей LSTM є їх здатність виявляти та враховувати часові залежності в текстових даних, що є особливо важливим для задачі класифікації новинних статей. Для подальшого вдосконалення системи автори пропонують кілька потенційних напрямків. По-перше, розширення та збагачення навчальних даних, що може допомогти моделі краще узагальнювати та бути більш стійкою до різноманітності стилів і змісту текстів. По-друге, інтеграція різнотипних ознак, таких як метадані новин, візуальна інформація та сигнали з соціальних мереж, може підвищити точність класифікації. Крім того, використання інших архітектур глибоких нейронних мереж та включення компоненту для пояснення рішень моделі може збільшити довіру користувачів до системи. Врахування лінгвістичних та культурних особливостей при адаптації моделі для різних мов також може покращити якість її роботи.

Як і попередня робота, [10] пропонує новий підхід до виявлення фейкових новин у соціальній мережі Twitter. Представлена модель базується на методах слабко контрольованого навчання. Ключовою відмінністю від традиційних підходів є автоматизація процесу формування навчальних даних, що дозволяє уникнути трудомісткої ручної розмітки твітів. Основна ідея запропонованого методу полягає у використанні інформації про надійність джерел новин для автоматичного маркування твітів. Замість безпосередньої класифікації кожного повідомлення як правдивого чи фейкового, твітам присвоюються мітки на основі репутації їхніх джерел - надійне або ненадійне. У рамках дослідження автори

зібрали масштабний датасет, що містить 401,414 твітів з 65 ненадійних та 46 надійних джерел новин. Незважаючи на те, що така розмітка не гарантує повної відповідності між репутацією джерела та достовірністю кожного окремого повідомлення, експериментальні результати показують, що класифікатор, навчений на даному датасеті, здатний виявляти фейкові новини з високою точністю - до 0.9. Запропонований підхід передбачає вилучення різноманітних ознак з текстів твітів та метаданих облікових записів користувачів, зокрема статистичних характеристик, результатів тематичного моделювання, аналізу тональності тексту тощо. Ці ознаки використовуються для навчання та валідації моделей машинного навчання різних типів, таких як наївний байєсів класифікатор, дерева рішень, метод опорних векторів та нейронні мережі. Головною перевагою описаного методу є можливість автоматичного генерування великих обсягів навчальних даних без потреби в ручній анотації, що дозволяє оперативно адаптувати модель до нових джерел та тематик. Водночас, істотним є ризик того, що модель може засвоювати специфічні зміщення, притаманні конкретним джерелам у навчальному наборі, замість узагальнених ознак фейкових новин.

У [11] представлено підхід до виявлення фейкових новин, який поєднує методи машинного навчання та обробки природної мови (NLP). Запропонована авторами модель базується на комбінації наївного баєсового класифікатора, методу опорних векторів (SVM) та семантичного аналізу тексту. Для навчання моделі використано датасет, завантажений з платформи Kaggle. Цей набір даних складається з двох csv-файлів (fake.csv та true.csv), які містять приклади фейкових та правдивих новин відповідно. Процес попередньої обробки даних включає очищення тексту від шуму та вилучення інформативних ознак. У рамках дослідження, ефективність запропонованої моделі порівнюється з іншими підходами до виявлення фейкових новин, такими як нейронні мережі та рекурентні нейронні мережі типу LSTM. Експериментальні результати показують, що наївний байєсів класифікатор, який є компонентом запропонованої моделі, досягає точності до 93.6% у розпізнаванні фейкових

новин. Даний підхід вирізняється комбінуванням різних методів машинного навчання та NLP, що дозволяє більш точно класифікувати текстові дані як правдиві або фейкові, проте, для практичного застосування підходу може знадобитися подолання обмежень, пов'язаних з потребою у великих обсягах розмічених даних та потенційною залежністю якості від предметної області новин.

ВИСНОВКИ ДО РОЗДІЛУ 1

У даному розділі було розглянуто теоретичні засади дослідження пропаганди в сучасному інформаційному просторі. Пропаганда визначається як навмисна спроба вплинути на сприйняття, емоції та поведінку людей шляхом поширення певних ідей, поглядів або ідеологій. Розуміння сутності пропаганди та її основних методів є необхідною передумовою для ефективної протидії маніпулятивним впливам у сучасному інформаційному просторі. Однак, зважаючи на величезні обсяги даних, які генеруються щодня, традиційні методи аналізу та виявлення пропаганди втрачають ефективність. Саме тому використання методів штучного інтелекту є потужним інструментом для автоматизованого виявлення та протидії пропаганді в текстових даних.

У даному розділі здійснено ґрунтовний огляд сучасних підходів до автоматичного розпізнавання пропаганди в текстових даних. Розглянуті дослідження засвідчують високу ефективність застосування методів машинного навчання, глибоких нейронних мереж в поєднанні з обробкою природної мови для ідентифікації пропагандистського контенту та фейкових новин. Зокрема, застосування згорткових нейронних мереж (CNN), рекурентних нейронних мереж (RNN) та їх комбінацій дозволяє успішно виявляти пропагандистський контент, враховуючи семантичні, синтаксичні та контекстуальні особливості тексту.

Окрім розробки нових архітектур моделей, значна увага приділяється методам попередньої обробки даних, вилучення ознак та формування навчальних вибірок. Дослідження показують, що автоматизація процесу розмітки даних, використання додаткових джерел інформації (метадані, візуальні ознаки, сигнали з соціальних мереж), врахування лінгвістичних і культурних особливостей текстів можуть суттєво покращити ефективність систем виявлення пропаганди.

Водночас, були виявлені певні обмеження та виклики, характерні для задачі автоматичного розпізнавання пропаганди. Зокрема, більшість підходів

вимагають значних обсягів розмічених даних для навчання моделей, процес збору, анотації та перевірки яких може вимагати значних зусиль. Крім того, ефективність розроблених методів може залежати від специфіки предметної області, жанру та стилістики текстів, що потребує додаткових зусиль для адаптації моделей до нових умов.

Проведений аналіз дозволяє окреслити перспективні напрямки подальших досліджень у сфері автоматичного розпізнавання пропаганди. Актуальною залишається розробка нових, більш ефективних архітектур моделей, які б покращили здатність до узагальнення в системах виявлення пропаганди. Важливим також є вдосконалення методів попередньої обробки та збагачення текстових даних, використання більш досконалих моделей векторного представлення слів та речень, інтеграція додаткових модальностей інформації, таких як зображення, відео та аудіо. Врахування ширшого контексту, зокрема, історичних, соціальних та культурних факторів, може також сприяти кращій адаптації систем до нових предметних областей та типів контенту.

РОЗДІЛ 2
РОЗРОБКА МЕТОДУ ВИЯВЛЕННЯ ЕЛЕМЕНТІВ ПРОПАГАНДИ В ТЕКСТОВИХ ДАНИХ

2.1. Огляд існуючих датасетів

Розробка ефективної системи автоматизованого виявлення пропаганди з використанням глибинних методів навчання вимагає ретельного підходу до формування навчальної вибірки. Першим етапом цього процесу є пошук або створення набору даних, який відповідає специфіці поставленої задачі та забезпечує достатню репрезентативність і якість даних. В рамках даної задачі датасет повинен містити збалансовану кількість екземплярів пропаганди і не пропаганди, бути достатньо великим для навчання моделей машинного навчання та мати надійну розмітку. Розглянемо кілька ключових датасетів, які використовуються в галузі роботи з пропагандою.

NLP4IF2019 - набір даних, розроблений в рамках воркшопу "Natural Language Processing for Internet Freedom". Датасет представляє собою 350 статтей для тренування моделей та 61 статтю для тестування, які загалом представляють 18 різних технік пропаганди, що були розглянуті у Розділі 1. Дані зібрані з різних новинних сайтів та охоплюють широкий спектр тем, включаючи політику, економіку, культуру тощо. Як і з багатьма іншими датасетами в цій галузі, існують певні обмеження, такі як відносно невеликий обсяг даних та фокус на англійській мові, тому подальші дослідження можуть бути спрямовані на розширення датасету, включення більшої кількості мов та врахування нових типів пропагандистських технік.

QProp Dataset є одним з найбільш масштабних та різноманітних наборів даних для виявлення пропаганди в текстових даних. На відміну від багатьох інших датасетів, які фокусуються на аналізі повних новинних статей, QProp Dataset містить близько 51,000 цитат, вилучених з приблизно 6,500 статей з різної тематики та анотованих експертами. Особливістю дататасету є наявність

додаткових атрибутів, таких як контекст цитати в межах статті, інформація про автора (спікера) та тему статті, що можуть бути використані для розробки більш складних моделей з урахуванням не лише текстових особливостей цитат, але й контекстуальної інформації. Однак, як і в попередньому датасеті, QProp Dataset містить тексти лише англійською мовою, що може обмежувати можливості його застосування для аналізу пропаганди в інших мовних середовищах. Окрім того обмеження розмітки до двох класів ("propagandistic" /"non-propagandistic") звузить можливості даної роботи до завдання бінарної класифікації.

FakeNewsNet Dataset є ще одним цінним ресурсом для дослідження та розробки методів виявлення фейкових новин та пропаганди в онлайн-медіа. Хоча основний фокус цього набору даних - фейкові новини, він може бути використаний для виявлення пропагандистського контенту, оскільки фейкові новини часто містять елементи пропаганди або самі по собі є формою пропаганди. Однією з переваг FakeNewsNet Dataset є його значний обсяг та різноманітність джерел. Він містить велику кількість новинних статей англійською мовою, зібраних з різних веб-сайтів та соціальних мереж, таких як Twitter та Facebook. однак, як і у випадку з QProp, бінарна розмітка ("Real"/"Fake") може обмежувати можливості для більш глибокого аналізу пропаганди.

TSHP-17 - датасет, що на відміну від двох попередніх, має розмітку на рівні документів (document-level) за чотирма класами: "trusted" (надійні), "satire" (сатиричні), "hoax" (містифікація) та "propaganda" (пропаганда). Особливістю TSHP-17 є використання методу віддаленого нагляду для розмітки статей. Це означає, що кожна стаття отримує мітку відповідно до класифікації джерела, з якого вона походить, а не на основі аналізу змісту самої статті. Такий підхід дозволяє швидко анотувати великі обсяги даних, але може призводити до певних неточностей, оскільки не всі статті з ненадійного джерела обов'язково містять пропаганду чи дезінформацію.

2.2. Етапи передпроцесингу текстової інформації

Процес передобробки тексту є невід'ємним та важливим етапом у задачах обробки природної мови, оскільки символи, слова та речення, визначені на цьому етапі, слугують базовими одиницями для подальших стадій обробки, таких як класифікація. Його основною метою є вилучення релевантної інформації з неструктурованих текстових даних, які часто містять шум та надлишкові елементи, такі як дати, числа та мовні одиниці без семантичного навантаження, зокрема прийменники та артиклі, а також приведення даних до формату, зручного для опрацювання алгоритмами штучного інтелекту.

У рамках цього дослідження будуть розглянуті такі методи передобробки тексту, як, нормалізація, видалення стоп-слів, стемінг та лематизація, токенізація та векторизація які дозволяють підготувати текстові дані для подальшого аналізу та використання в моделях машинного навчання.

Один з перших етапів обробки тексту – це нормалізація - процес приведення тексту до єдиної канонічної форми, що допомагає уникнути проблем при зберіганні або обробці даних завдяки узгодженості формату вводу. Нормалізація потребує чіткого розуміння того, який тип тексту підлягає обробці та як він буде оброблятися в подальшому, тому не існує універсального методу нормалізації. Популярними методами нормалізації є видалення стоп-слів, стемінг або лематизація. Крім того, процес нормалізації може включати перетворення всіх літер в нижній регістр, видалення розділових знаків, обробку чисел та інших символів, або ж більш специфічні кроки як вилучення заголовків файлів, нижніх колонтитулів, розмітки та метаданих HTML/XML, або ж витягування корисної інформації з інших джерел, таких як JSON чи база даних. У випадку мультимовних даних може виникати необхідність зведення до однієї мови або врахування специфіки кожної з наявних мов, зокрема правила морфології та синтаксису. Розглянемо деякі підходи детальніше.

Стемінг зводить слова до їхніх основних форм, видаляючи суфікси та інші морфологічні ознаки, що дозволяє зменшити кількість унікальних слів у тексті.

Наприклад, слова "running", "runs", "ran" після стемінгу були б зведені до основи "run". Лематизація, на відміну від стемінгу, знаходить базову або словникову форму слова – лему, - з урахуванням його морфологічних характеристик. Результатом завжди є дійсне слово мови. Наприклад, слово "better" після лематизації буде приведене до леми "good". (Рисунок 2.1)

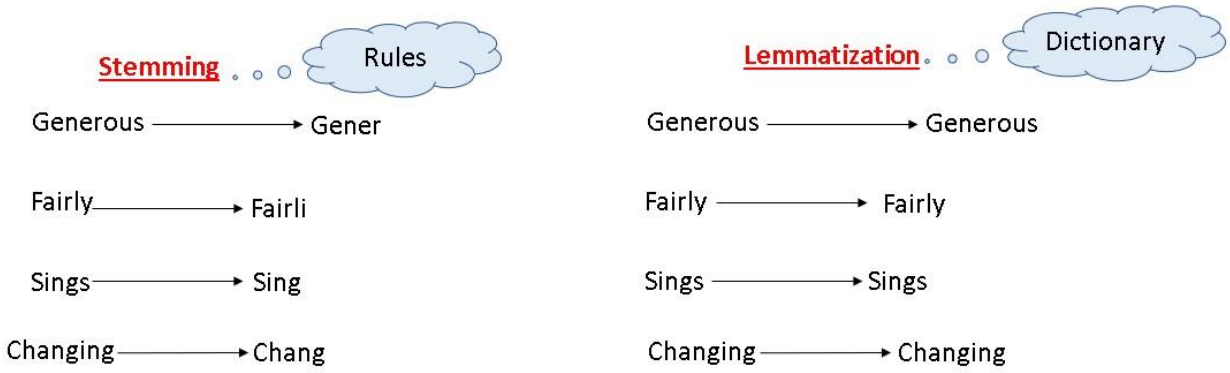

Рис. 2.1. Порівняння стемінгу і лематизації

Стемінг є простішим та швидшим методом, хоча може призвести до втрати семантичної інформації, тоді як лематизація є більш складною та повільнішою, та забезпечує більш точні результати. Вибір між стемінгом та лематизацією залежить від конкретного завдання та вимог до точності та ефективності обробки тексту. У багатьох випадках стемінг може бути достатнім і забезпечувати хороші результати, особливо для великих обсягів даних. Однак, коли необхідна висока точність та збереження семантичної інформації, лематизація може бути кращим вибором.

Видалення стоп-слів відокремлює значущу інформацію від зайвої, видаляючи слова, які не несуть значного смислового навантаження, такі як прийменники та сполучники. Стоп-слова, такі як "і", "є", "це", є загальновідомими прикладами - вони не додають інформації до контексту чи змісту тексту, проте висока частота використання ускладнює розуміння документів. Варто зауважити, що створення списку стоп-слів є непростим завданням, що вимагає врахування конкретного корпусу текстів та мети обробки.

Векторизація перетворює текстову інформацію на числові вектори, що важливо для зменшення розмірності вхідних даних і підготовки тексту до обробки алгоритмами машинного навчання. Розглянемо основні сучасні алгоритми векторизації текстів: Bag of Words, TF-IDF та Word2Vec, та порівняємо їх особливості, переваги та недоліки.

Bag of Words (BoW) - це найпростіший алгоритм векторизації текстів, який представляє документ як невпорядкований набір слів, ігноруючи граматику та порядок слів. Кожен унікальний терм у корпусі текстів становить окремий вимір вектора, а значення кожного виміру дорівнює кількості появ цього терму в документі (Рисунок 2.2). Перевагами BoW є простота реалізації та інтерпретації, а також можливість працювати з великими корпусами текстів. Недоліками є ігнорування семантичних зв'язків між словами та проблема розмірності, коли вектори мають дуже велику кількість вимірів через велику кількість унікальних термів у корпусі.

Рис. 2.2. Схема алгоритму векторизації Bag of Words

TF-IDF (Term Frequency-Inverse Document Frequency) - це вдосконалений алгоритм векторизації, який враховує не лише частоту появи термів у документі, а й їх “важливість” у контексті всього корпусу (Рисунок 2.3). TF-IDF складається з двох частин: TF (частота терму) вимірює, як часто терм з'являється в документі,

a IDF (обернена частота документа) вимірює важливість терму у всьому корпусі. Терми, які з'являються в багатьох документах, отримують меншу вагу, тоді як унікальні терми - більшу. Перевагами TF-IDF є врахування важливості термів та зменшення впливу поширених слів. Недоліками є все ще висока розмірність векторів та ігнорування семантичних зв'язків між словами.

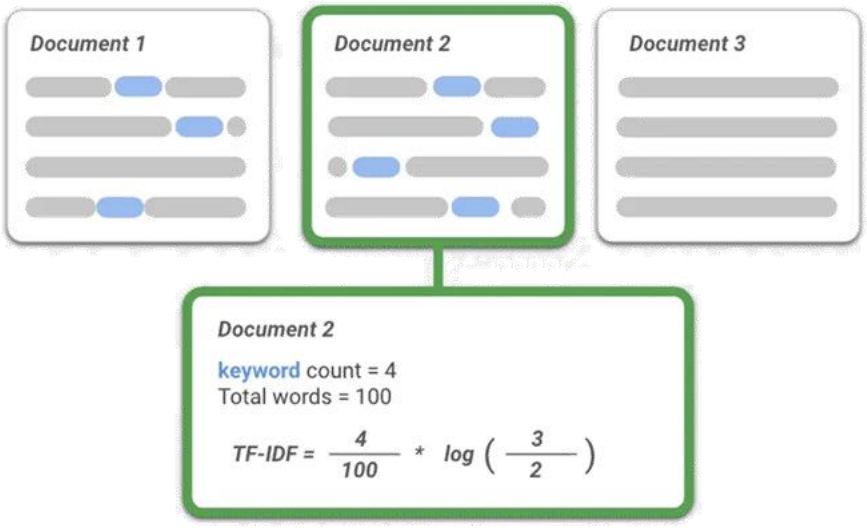

Рис. 2.3. Схема алгоритму векторизації TF-IDF

GloVe (Global Vectors for Word Representation) - це вдосконалений алгоритм векторизації, який поєднує елементи методів статистичного підходу та нейронних мереж для створення векторних представлень слів. Головною особливістю GloVe є врахування як локальні, так і глобальні властивостей слів у тексті. GloVe використовує матрицю співвідношень частоти спільного вживання слів у корпусі для побудови векторів. Алгоритм намагається навчити вектори так, щоб відношення між словами векторному просторі відображало їх семантичні зв'язки. Наприклад, вектори слів "король" і "королева" будуть близькими один до одного, а їх відмінність буде схожою на різницю між векторами "чоловік" і "жінка". Очевидними перевагами алгоритму є здатність враховувати як локальний контекст (використання слів у конкретних текстових вікнах), так і глобальний контекст (загальна частота спільного вживання слів у

всьому корпусі). Це дозволяє створювати векторні представлення слів, які добре відображають їх семантичні зв'язки. Проте недоліками такого підходу є складність і значні витрати ресурсів, а також статичність отриманих векторів – тобто кожне слово матиме одне векторне представлення незалежно від контексту, в якому воно використовується, що може бути недоліком для полісемантичних слів.

Вибір алгоритму залежатиме від специфіки задачі та доступних ресурсів. Bag of Words є найпростішим і може бути достатнім для простих задач класифікації з невеликою кількістю класів. TF-IDF є більш досконалим і може давати кращі результати, особливо коли важливість термів відіграє значну роль. GloVe також є потужним інструментом, особливо коли важливо врахувати контексти слів на різних рівнях сприйняття, але вимагає більших обчислювальних ресурсів та обсягів даних для навчання.

2.3. Обґрунтування вибору методів глибинного навчання для виявлення пропаганди в текстових даних

Виявлення пропаганди в текстових даних є комплексною проблемою, що вимагає глибокого розуміння та врахування різноманітних лінгвістичних, семантичних та контекстуальних факторів. Пропагандистські тексти характеризуються складною структурою та використанням різноманітних стилістичних прийомів, спрямованих на маніпулювання думкою читача. Вони часто містять неявні смислові конотації, емоційно забарвлену лексику та риторичні засоби, що ускладнює їх автоматичну ідентифікацію.

Традиційні підходи до машинного навчання, що базуються на ручному виділенні ознак (feature engineering), мають суттєві обмеження при роботі з такими складними текстовими даними. Ці методи вимагають значних зусиль експертів для визначення та програмування релевантних ознак, що можуть бути використані для виявлення пропаганди, однак, “ручне” виділення ознак часто не дозволяє повною мірою охопити всі нюанси та приховані патерни, притаманні

цьому типу контенту. Таким чином, ключовою перевагою глибинного навчання є його здатність автоматично виявляти та генерувати інформативні ознаки безпосередньо з необроблених текстових даних. Глибинні нейронні мережі, завдяки своїй багатошаровій архітектурі та нелінійним функціям активації, здатні вивчати абстрактні та ієрархічні репрезентації тексту, що дозволяє їм ефективно розпізнавати складні патерни та взаємозв'язки, характерні для пропагандистських текстів. Окрім того, деякі глибинні моделі, такі як рекурентні нейронні мережі (RNN), здатні враховувати послідовну природу тексту та моделювати довготривалі залежності між словами та реченнями. Це дозволяє виявляти пропагандистські патерни, які можуть бути розподілені по всьому тексту і не обмежуються лише локальними ключовими словами або фразами.

Ще однією суттєвою особливістю глибинних моделей є їх здатність до трансферного навчання (transfer learning) - підходу, що дозволяє адаптувати знання, отримані з попередньо навчених моделей, для вирішення нових, споріднених задач. До найвідоміших моделей трансформерів можна віднести BERT (Bidirectional Encoder Representations from Transformers) або GPT (Generative Pre-trained Transformer) - це мовні моделі, розроблені провідними технологічними компаніями, були треновані на надвеликих обсягах текстових даних з різноманітних джерел та тем. В процесі навчання вони набули здатності до глибокого розуміння семантики та синтаксису в текстах, що нині робить їх цінним ресурсом для широкого спектру задач обробки природної мови. Процес трансферного навчання передбачає додаткове навчання (попередньо навченої) моделі-трансформера на відносно невеликому обсязі анотованих даних, специфічних для певної проблеми, в даному випадку, ідентифікації пропагандистського контенту. Завдяки передачі знань з попереднього навчання, модель може швидко адаптуватися до особливостей пропагандистських текстів та досягати високої ефективності навіть на обмежених обсягах спеціалізованих даних. Такий підхід дозволяє суттєво зменшити потребу в масштабних анотованих датасетах.

Розглянемо детальніше деякі архітектури глибинного навчання та їх можливості.

Рекурентні нейронні мережі є особливо важливими для обробки природної мови, де значення слів та речень залежить від їх контексту. RNN мають внутрішню пам'ять, яка дозволяє їм зберігати інформацію про попередні елементи послідовності та використовувати її для прийняття рішень щодо поточного елемента. Завдяки цим властивостям, вони ефективніші в задачах аналізу тональності, машинного перекладу та генерації тексту.

Графові нейронні мережі (GNN) - це архітектура глибинного навчання, яка дозволяє обробляти дані, представлені у вигляді графів. У контексті обробки мови, GNN можуть бути використані для представлення тексту як графа, де вузли відповідають словам або реченням, а ребра відображають їх взаємозв'язки, такі як синтаксичні відношення. GNN здатні враховувати структурну інформацію тексту та моделювати складні залежності між його елементами. Це дозволяє виявляти пропагандистські прийоми, які спираються на маніпуляцію структурою аргументів або використання специфічних мовних конструкцій. GNN також ефективні для задач, де важливо враховувати взаємозв'язки між об'єктами, наприклад, для виявлення фейкових новин або аналізу соціальних мереж.

Інша архітектува, що представляє генеративні моделі це - Варіаційні автоенкодери (VAE). VAE складаються з енкодера та декодера. Енкодер стискає вхідні дані в латентний простір меншої розмірності, а декодер відновлює вхідні дані з латентного представлення. VAE навчаються мінімізувати різницю між вхідними даними та їх реконструкцією, одночасно регуляризуючи латентний простір таким чином, щоб він мав певні властивості, наприклад, був неперервним та гладким. Завдяки цьому, VAE можуть генерувати нові зразки, подібні до навчальних даних. У контексті виявлення пропаганди, вони можуть бути використані для моделювання розподілу текстових даних без пропаганди. Якщо застосувати навчений VAE до нових текстів, то пропагандистські зразки будуть погано відтворюватися декодером, що дозволить виявляти їх як аномалії.

Згорткові нейронні мережі (CNN) - тип нейронних мереж, які широко використовуються для найрізноманітніших задач, зокрема для задач обробки даних з просторовою структурою, таких як зображення або текст. CNN працюють на основі операції згортки. Згортка застосовується до матриці вхідних символів, щоб виявити локальні ознаки та шаблони. Згорткові шари можуть автоматично навчатися розпізнавати важливі ознаки, такі як ключові слова, фрази або стилістичні особливості,, і застосовуватись на різних рівнях ієрархії для виявлення ознак різного масштабу.

Застосування будь-якого з перелічених методів глибинного навчання дозволяє подолати обмеження традиційних підходів і відкриває нові можливості для автоматичного виявлення пропаганди в текстових даних.

2.4. Застосування ієрархічних механізмів уваги для задачі виявлення пропаганди

Пропагандистський контент може проявлятися на різних рівнях деталізації, від окремих слів та фраз до цілих речень, абзаців та документів. Одним з ключових компонентів сучасних моделей глибинного навчання, які дозволяють вирішити цю проблему, є механізми уваги. Застосування механізмів уваги дозволяє моделі адаптивно зосереджуватись на найбільш інформативних та релевантних частинах тексту для виявлення пропаганди.

Дана робота пропонує підхід до виявлення пропаганди в текстових даних, який полягає у застосуванні механізмів уваги на різних рівнях деталізації тексту. Ключовою особливістю запропонованого методу є використання ієрархічної структури уваги, яка дозволяє моделі аналізувати та розпізнавати пропагандистський контент на трьох основних рівнях гранулярності: на рівні окремих слів, на рівні речень та на рівні цілісних документів. Очікується, що такий багаторівневий підхід до застосування уваги надасть моделі можливість враховувати різноманітні аспекти та контексти, притаманні пропагандистським текстам.

Розглянемо детальніше теоретичні основи механізмів уваги та їх застосування на кожному з трьох рівнів запропонованої моделі. Увага базується на концепції “вирівнювання” між елементами вхідної послідовності та контекстним вектором, який представляє поточний стан моделі. Значення уваги обчислюються як зважена сума елементів вхідної послідовності, де ваги відображають релевантність кожного елемента для поточної задачі.

На найнижчому, рівні слів, модель зможе ідентифікувати специфічні лінгвістичні маркери, такі як емоційно забарвлена лексика, маніпулятивні фрази або ключові слова, характерні для пропагандистського контенту. На рівні речень модель має виявляти більш складні та контекстно-залежні патерни пропаганди, які можуть бути розподілені в межах речення або навіть охоплювати кілька речень. На рівні документів, модель аналізуватиме глобальну структуру та семантику тексту, розпізнаючи високорівневі ознаки пропаганди (загальна тональність, аргументаційна структура або наративні прийоми).

Таким чином, ключовою ідеєю даного дослідження є розробка та вивчення ієрархічної моделі глибинного навчання на основі механізмів уваги, яка дозволить виявляти пропаганду на різних рівнях деталізації, забезпечуючи більш комплексний та контекстно-залежний аналіз текстових даних. Описаний метод має потенціал значно підвищити ефективність та точність розпізнавання пропаганди в порівнянні з традиційними методами, які зазвичай зосереджуються лише на одному рівні аналізу тексту. Окрім того, застосування механізмів уваги на різних рівнях має підвищити інтерпретованість моделі: аналізуючи ваги уваги, призначені моделлю на кожному рівні, можна визначити, які саме слова, речення або фрагменти тексту мали найбільший вплив на рішення моделі щодо наявності пропаганди - це є цінним для розуміння специфічних лінгвістичних та семантичних маркерів пропаганди, а також для пояснення та обґрунтування результатів моделі, спрощення підбору її параметрів.

Поєднання ієрархічних механізмів уваги з потужними архітектурами глибинного навчання дозволить розробити комплексну та ефективну модель для виявлення пропаганди в текстових даних, таку що враховуватиме різноманітні

аспекти та контексти пропагандистського контенту, забезпечуючи високу точність та інтерпретованість результатів. Для вирішення поставленої у даній роботі задачі було обрано комбінацію BiLSTM, CNN та трансформера з застосуванням ієрархічного механізму уваги на різних рівнях обробки тексту.

BiLSTM - це різновид рекурентних нейронних мереж, які здатні обробляти послідовності даних, враховуючи інформацію як з попередніх, так і з наступних елементів. На відміну від звичайних LSTM, які обробляють послідовність тільки в прямому напрямку, BiLSTM використовують два окремі шари LSTM: для обробки послідовності в прямому та у зворотному напрямках. Таким чином, BiLSTM можуть отримати більш повне розуміння семантики та залежностей у тексті. Поєднання BiLSTM зі згортковою нейронною мережею та мережею-трансформером в ієрархічній моделі з механізмами уваги дозволить комплексно проаналізувати текстові дані на різних рівнях. BiLSTM забезпечить розуміння локального контексту, CNN виділить релевантні ознаки та шаблони, а трансформери змоделюють глобальні залежності та взаємозв'язки.

ВИСНОВКИ ДО РОЗДІЛУ 2

Даний розділ закладає теоретичну та методологічну основу для розробки ефективного методу виявлення елементів пропаганди в текстових даних з використанням глибинного навчання.

У розділі сформовано вимоги до набору даних для вирішення поставленого завдання, також зроблено огляд популярних готових датасетів, пов'язаних з темою пропаганди - NLP4IF2019, QProp Dataset, FakeNewsNet Dataset та TSHP-17.

Розглянуто ключові етапи процесу передобробки тексту, який є невід'ємною частиною підготовки даних до аналізу та використання в моделях машинного навчання. У даному пункті порівнюватись різні методи векторизації текстів, зокрема Bag of Words, TF-IDF та GloVe, з зазначенням особливостей, переваг та недоліків для вирішення задачі роботи.

Обґрунтувано вибір методів глибинного навчання для виявлення пропаганди в текстових даних через здатність автоматично виявляти та генерувати інформативні ознаки з необроблених текстових даних. Розглянуто різні архітектури глибинного навчання та їхні особливості.

Запропоновано метод ієрархічних механізмів уваги для виявлення пропаганди на різних рівнях деталізації тексту: рівні окремих слів, речень та цілісних документів. Поєднання даного механізму з потужними архітектурами глибинного навчання - BiLSTM, CNN та моделі-трансформеру, дозволить розробити комплексну та ефективну модель для виявлення пропаганди в текстових даних.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ МЕТОДУ ВИЯВЛЕННЯ ПРОПАГАНДИ НА ОСНОВІ ГЛИБИННИХ НЕЙРОННИХ МЕРЕЖ

3.1. Опис засобів реалізації

Для реалізації методу поставленої задачі використовувалась мова програмування Python, яка є однією з найпопулярніших мов для задач машинного навчання та обробки природної мови. Python має багату екосистему бібліотек та фреймворків, які спрощують розробку та прискорюють процес дослідження.

Основними технологіями та бібліотеками, використаними в даному коді, є TensorFlow, Keras, NLTK та scikit-learn.

TensorFlow - це відкритий фреймворк для машинного навчання, розроблений компанією Google. Він надає потужні інструменти для побудови та навчання глибинних нейронних мереж. TensorFlow дозволяє легко будувати складні моделі, управляти обчислювальними графами та виконувати ефективні обчислення на CPU, GPU та TPU. Вибір TensorFlow обумовлений його гнучкістю, продуктивністю та широкою підтримкою спільноти.

Keras - це високорівневий API для побудови нейронних мереж, який може працювати як оболонка TensorFlow. Keras спрощує процес створення та навчання моделей завдяки зручному та інтуїтивно зрозумілому синтаксису. Він надає багато вбудованих шарів та функцій втрат, що дозволяє швидко експериментувати з різними архітектурами. Keras також підтримує завантаження попередньо навчених моделей та легку інтеграцію з іншими бібліотеками. Значною перевагою є простота використання та можливістю швидкої розробки прототипів моделей.

NLTK (Natural Language Toolkit) - це бібліотека Python для обробки природної мови. Вона надає широкий набір інструментів для токенізації, стемінгу, лематизації та багатьох інших задач обробки тексту. NLTK має

вбудовані корпуси та лексичні ресурси, які можуть бути використані для навчання моделей.

Scikit-learn - це бібліотека машинного навчання для Python, яка надає широкий спектр алгоритмів для класифікації, регресії, кластеризації та оцінки моделей. Scikit-learn має зручний та уніфікований інтерфейс для різних алгоритмів, що дозволяє легко порівнювати моделі, вибирати найкращу. Бібліотека також надає функції для попередньої обробки даних, вибору ознак та крос-валідації. Окрім того Scikit-learn відома простотою використання, ефективністю та сумісністю з іншими бібліотеками Python.

Поєднання дозволяє будувати потужні моделі для виявлення пропаганди, обробляти та аналізувати текстові дані, а також оцінювати якість отриманих результатів. Використання Python забезпечує швидку та ефективну розробку, дозволяє експериментувати з різними підходами та архітектурами, а також спрощує інтеграцію з іншими компонентами системи.

3.2. Загальна архітектура моделі

Розглянемо детальніше архітектуру моделі, запропоновану в попередньому розділі. Побудована модель використовує складний підхід з використанням багаторівневих механізмів уваги, інтегруючи рекурентні двонапрямлені шари LSTM, шари згортки та трансформерні механізми, кожен з яких виконує специфічну роль у процесі вилучення ознак з вхідних даних (Рисунок 3.1):

```
Model: "custom_model_7"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_7 (Embedding)     multiple                  320000

 bidirectional_7 (Bidirecti  multiple                  16640
 onal)

 time_distributed_26 (TimeD  multiple                  65
 istributed)

 conv1d_7 (Conv1D)           multiple                  3104

 time_distributed_27 (TimeD  multiple                  33
 istributed)

 multi_head_attention_6 (Mu  multiple                  8416
 ltiHeadAttention)

 time_distributed_28 (TimeD  multiple                  33
 istributed)

 concatenate_6 (Concatenate  multiple                  0
 )

 time_distributed_29 (TimeD  multiple                  2451
 istributed)

=================================================================
Total params: 351103 (1.34 MB)
Trainable params: 351103 (1.34 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Рис. 3.1. Запропонована архітектура моделі

1. Embedding Layer (embedding_7) - шар перетворює індекси слів у послідовності на вектори фіксованої довжини.
2. Bidirectional LSTM Layer (bidirectional_7) - двонаправлена LSTM з двох шарів: один обробляє послідовність у прямому напрямку, інший — у зворотному. Це дозволяє враховувати контекст з обох боків.
3. TimeDistributed Layer (time_distributed_26) - застосовує шар Dense до кожного кроку вхідної послідовності окремо, що дозволяє обробляти кожен елемент послідовності окремо.

4. Conv1D Layer (conv1d_7) - Одновимірний шар згортки, що застосовується для виявлення локальних патернів у послідовності. Він використовує фільтри, що ковзають по послідовності та генерують вихідні карти ознак.
5. TimeDistributed Layer (time_distributed_27) - аналогічний попередньому шару TimeDistributed, застосовує шар Dense до кожного тимчасового кроку вхідних даних.
6. Multi-Head Attention Layer (multi_head_attention_6) - багатоголовий механізм уваги дозволяє моделі зосереджуватися на різних частинах вхідної послідовності одночасно. Шар включає в себе кілька "голів", кожна з яких виконує увагу незалежно, після чого результати об'єднуються.
7. TimeDistributed Layer (time_distributed_28) - ще один шар TimeDistributed.
8. Concatenate Layer (concatenate_6) - шар об’єднання, створює єдине представлення для виходів з декількох попередніх шарів уздовж вказаної осі.
9. TimeDistributed Layer (time_distributed_29) - останній шар TimeDistributed у моделі.

Завдяки такій комбінації шарів модель здатна ефективно обробляти складні послідовні дані, забезпечуючи високу якість результатів для задач, пов'язаних з аналізом тексту.

3.3. Алгоритм обробки тексту

Враховуючи специфіку даного дослідження, спрямованого на підвищення ефективності виявлення елементів пропаганди в текстах, було вирішено використовувати датасет NLP4IF-2019. Широкий спектр пропагандистських технік (18 різних типів) забезпечать достатнє різноманіття прикладів для навчання моделей, а розмітка на рівні фрагментів дозволить не лише класифікувати документи в цілому, але й локалізувати пропагандистські елементи всередині тексту, що має значення для практичного застосування розроблених моделей (Рисунок 3.2).

Рис. 3.2. Приклад розмітки випадкової статті з датасету NLP4IF-2019

Датасет складається з навчальної (train) та тестової (test) вибірок. Кожна вибірка містить набір текстових статей та відповідних анотацій щодо наявності пропаганди.

Розглянемо детальніше структуру набору даних:

- Кожна стаття знаходиться в окремому текстовому файлі з розширенням ".txt".
- Для кожної статті існує відповідний файл з анотаціями, з такою ж назвою, але з додатковим розширенням ".labels.tsv".
- Файл з анотаціями зберігає інформацію про наявність пропаганди у певній статті у вигляді рядків, де кожен рядок складається з чотирьох полів, розділених табуляцією: ідентифікатор статті, мітка пропаганди, початкова позиція фрагменту з пропагандою, кінцева позиція фрагменту з пропагандою (Рисунок 3.3).

article111111112.labels.tsv | article111111112.txt

```
111111112   Slogans 191 221
111111112   Black-and-White_Fallacy 476 556
111111112   Slogans 785 798
111111112   Loaded_Language 958 1015
111111112   Flag-Waving 1456    1536
111111112   Name_Calling,Labeling   1810    1824
111111112   Loaded_Language 2095    2125
111111112   Loaded_Language 911 942
111111112   Name_Calling,Labeling   1738    1787
```

Рис. 3.3. Приклад файлу анотацій

Розглянемо детальніше процес обробки даних:

1. Завантаження:
   1.1. Для кожної статті зчитується текстовий вміст файлу та зберігається в список `articles`.
   1.2. Для кожної статті зчитуються відповідні анотації з файлу ".labels.tsv" та зберігаються в список `labels`.
   1.3. Початкові та кінцеві позиції фрагментів з пропагандою зберігаються в список `label_ranges`.
   1.4. Текст кожної статті буде розділено на слова, позиції фрагментів з пропагандою з `label_ranges` переведено в формат послідовностей міток для кожної статті (таким чином щоб кожне слово статті було віднесено до певного класу)
2. Попередня обробка тексту:
   2.1. Текст кожної статті проходить через функцію `normalize_text()`, яка:
   - Виконує приведення тексту до нижнього регістру.
   - Видаляє розділові знаки.

   2.2. Після нормалізації, текст статті проходить через функцію `stem_text()`, яка виконує стемінг слів за допомогою `PorterStemmer` з бібліотеки NLTK.
3. Токенізація та створення словника:

3.1. Наступним кроком виконується токенізація текстів статей за допомогою `Tokenizer` з Keras.

3.2. Створюється словник `word_index`, який містить відображення слів на їх індекси.

4. Векторизація

4.1. Функція `load_glove_embeddings()` завантажує попередньо навчені вектори слів з файлу `glove.6B.100d.txt`.

4.2. Створюється матриця вкладень `embedding_matrix`, де кожен рядок відповідає вектору вкладення для відповідного слова зі словника.

4.3. Тексти статей перетворюються на числові послідовності за допомогою `tokenizer.texts_to_sequences()`.

4.4. Числові послідовності доповнюються або обрізаються до однакової довжини `max_sequence_length` за допомогою `pad_sequences()`.

Модель приймає на вхід послідовності слів у числовому форматі, які спочатку проходять через шар Embedding для перетворення слів у щільні вектори фіксованої розмірності. Потім ці вектори подаються на вхід трьох паралельних гілок обробки: BiLSTM, CNN та трансформера.

BiLSTM гілка (Увага на рівні слів):

- Вкладення слів подаються на вхід двонаправленої LSTM, яка обробляє послідовність в прямому та зворотному напрямках.
- Вихід BiLSTM передається через шар уваги TimeDistributed(Dense), який обчислює скалярні ваги уваги для кожного часового кроку.
- Ваги уваги нормалізуються за допомогою softmax та множаться на вихід BiLSTM для отримання зваженого контекстного вектора.

CNN гілка (Увага на рівні локальних патернів):

- Вкладення слів також подаються на вхід одновимірної згорткової нейронної мережі (Conv1D).
- Вихід CNN передається через шар уваги TimeDistributed(Dense), який обчислює скалярні ваги уваги для кожного часового кроку.

- Ваги уваги нормалізуються за допомогою softmax та множаться на вихід CNN для отримання зваженого контекстного вектора.

Гілка Transformer (Увага на рівні послідовностей):

- Вкладення слів подаються на вхід шару MultiHeadAttention, який реалізує механізм багатоголової уваги.
- Багатоголова увага дозволяє моделі зважувати внесок різних частин послідовності на основі їх релевантності.
- Вихід шару уваги передається через шар уваги TimeDistributed(Dense), який обчислює скалярні ваги уваги для кожного часового кроку.
- Ваги уваги нормалізуються за допомогою softmax та множаться на вихід шару уваги для отримання зваженого контекстного вектора.

Виходи трьох гілок (BiLSTM, CNN та шару трансформера) конкатенуються вздовж осі ознак. Об'єднаний вектор ознак потім передається через вихідний шар TimeDistributed(Dense) з активацією softmax для отримання розподілу ймовірностей по класах для кожного часового кроку (Рисунок 3.4).

Модель використовує CRF (Conditional Random Field) для обчислення функції втрат. CRF - це ймовірнісна графічна модель, яка враховує залежності між мітками класів у послідовності та забезпечує узгодженість передбачень.

Модель приймає передбачені ймовірності класів (y_pred), справжні мітки класів (y_true), довжини послідовностей та параметри переходів між мітками класів і обчислює негативний логарифм правдоподібності для послідовності міток, враховуючи передбачені ймовірності та параметри переходів. Ця функція втрат заохочує модель призначати вищі ймовірності правильним міткам класів та враховувати залежності між сусідніми мітками. Параметри переходів ініціалізуються як змінна моделі та оптимізуються під час навчання разом з іншими параметрами моделі.

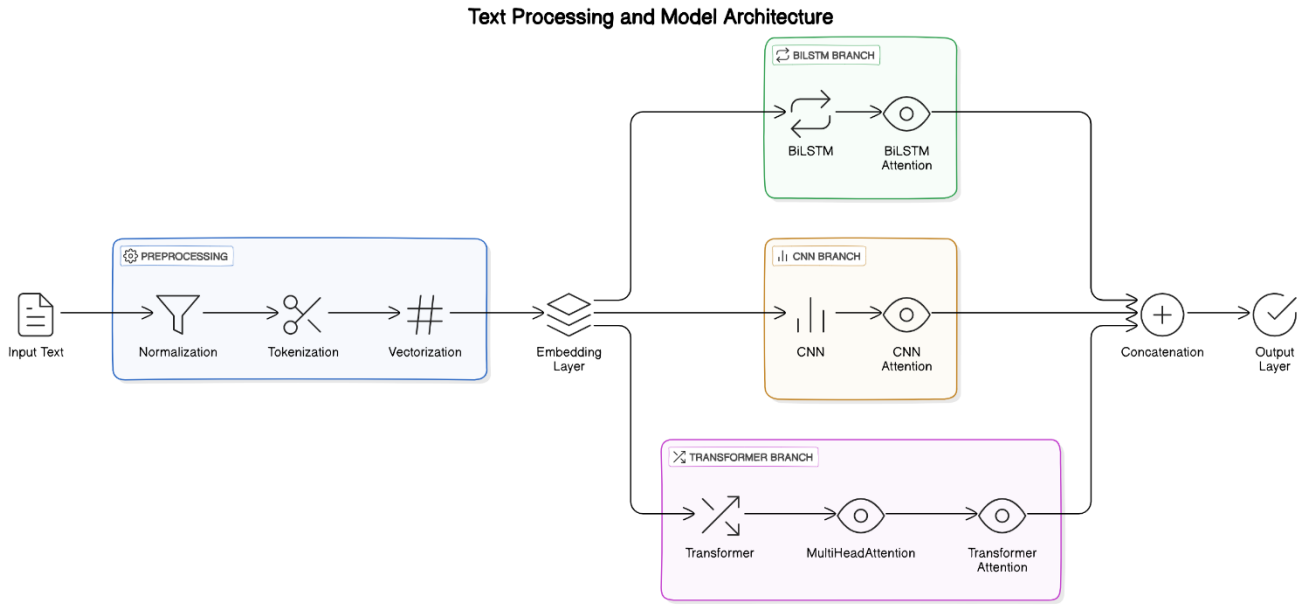

Рис. 3.4. Схема обробки текстових даних

Для оптимізації параметрів моделі використовується оптимізатор Adaptive Moment Estimation - Adam. Adam - це популярний алгоритм оптимізації, який адаптивно налаштовує швидкість навчання для кожного параметра моделі на основі оцінок першого та другого моментів градієнтів. Він поєднує переваги інших оптимізаторів, таких як Adagrad та RMSprop, і має хорошу збіжність та ефективність на практиці, дозволяє швидше досягти оптимуму та уникнути застрягання в локальних мінімумах.

Для запобігання перенавчанню та покращення узагальнюючої здатності моделі використовуються 2 методи регуляризації: EarlyStopping та Learning Rate Reduction.

Метод ранньої зупинки EarlyStopping зупиняє процес навчання, якщо значення функції втрат на валідаційному наборі даних не покращується протягом певної кількості епох.

Метод зменшення швидкості навчання ReduceLROnPlateau зменшує швидкість навчання, якщо значення функції втрат на валідаційному наборі даних не покращується протягом певної кількості епох.

Таким чином, запропонована модель реалізує механізми уваги на різних рівнях обробки тексту (BiLSTM, CNN та шар уваги трансформера) для ефективного виявлення пропаганди. Ці механізми уваги дозволяють моделі зважувати внесок різних частин послідовності та враховувати контекстну інформацію. Комбінація різних архітектур та механізмів уваги дозволяє моделі виявляти складні пропагандистські прийоми та покращувати точність передбачень.

ВИСНОВКИ ДО РОЗДІЛУ 3

У даному розділі було детально розглянуто реалізацію методу виявлення пропаганди на основі глибинних нейронних мереж з використанням мови програмування Python та її екосистеми бібліотек. Запропонований підхід грунтується на поєднанні потужних архітектур глибинного навчання, таких як Bidirectional LSTM, CNN та механізмів уваги, з метою ефективного виявлення пропагандистських технік у текстових даних.

Для реалізації моделі було обрано фреймворк TensorFlow та його високорівневий API Keras, які надають гнучкі та ефективні інструменти для побудови та навчання нейронних мереж. Використання цих технологій дозволило швидко експериментувати з різними архітектурами та гіперпараметрами моделі, а також забезпечило високу продуктивність обчислень. Для запобігання перенавчанню та покращення узагальнюючої здатності моделі було застосовано методи регуляризації, такі як рання зупинка (EarlyStopping) та зменшення швидкості навчання (ReduceLROnPlateau). Ці підходи дозволили зупинити процес навчання у випадку погіршення якості на валідаційному наборі даних та адаптивно налаштовувати швидкість навчання для досягнення оптимальних результатів.

Запропонована модель окрім класифікації тексту на той що містить і не містить пропаганду, здатна розпізнавати 18 різних технік пропаганди, зокрема "Loaded Language", "Name Calling,Labeling", "Appeal to fear-prejudice" та інших. Використання механізмів уваги дозволило моделі зосереджуватись на найбільш релевантних фрагментах тексту та враховувати контекстну інформацію при прийнятті рішень.

РОЗДІЛ 4
ОЦІНКА ЕФЕКТИВНОСТІ РОЗРОБЛЕНОЇ СИСТЕМИ

4.1. Метрики оцінки ефективності

Для оцінки ефективності розробленої системи виявлення пропаганди в текстових даних розглянемо детальніше основні метрики оцінки якості.

Accuracy - частка правильно класифікованих текстів серед усіх текстів. Не є повністю репрезентативною метрикою, оскільки при незбалансованих класах, коли кількість зразків одного класу значно перевищує інший, може давати викривлене уявлення про ефективність моделі.

Precision - для кожного класу розраховується як відношення правильно класифікованих зразків цього класу до всіх зразків, які модель віднесла до цього класу. Для отримання загальної Precision використовується макро-, мікро- або зважене усереднення:

- Macro Precision: середнє арифметичне Precision для кожного класу.
- Micro Precision: відношення суми правильно класифікованих зразків усіх класів до суми зразків, які модель віднесла до будь-якого класу.
- Weighted Precision: середнє зважене Precision для кожного класу, де вагами є кількість зразків у кожному класі.

Recall - для кожного класу розраховується як відношення правильно класифікованих зразків цього класу до всіх зразків, які насправді належать до цього класу. Загальна Recall отримується аналогічно до Precision за допомогою макро-, мікро- або зваженого усереднення.

F1-score - гармонійне середнє між Precision та Recall. Для небінарної класифікації використовується макро-, мікро- або зважене усереднення F1-score:

- Macro F1: середнє арифметичне F1-score для кожного класу.
- Micro F1: розраховується на основі мікро-усереднених Precision та Recall.

– Weighted F1: середнє зважене F1-score для кожного класу, де вагами є кількість зразків у кожному класі.

Матриця помилок (Confusion Matrix) - таблиця розміру NxN (де N - кількість класів), в якій елемент (i, j) показує кількість зразків фактичного класу i, які були класифіковані моделлю як клас j. Аналіз матриці помилок дає інформацію про правильність класифікації для кожного класу, найбільш поширені помилки та незбалансованість класів.

Матриця помилок є основою для розрахунку різних метрик оцінки ефективності моделі, таких як точність, повнота, точність та F1-score, її аналіз дає багато корисної інформації:

1. Правильність класифікації для кожного класу: діагональні елементи показують, наскільки добре модель розпізнає кожен окремий тип пропаганди.
2. Найбільш поширені помилки: найбільші значення поза діагоналлю вказують на типи пропаганди, які модель найчастіше плутає між собою. Це може свідчити про схожість цих типів або про недостатню здатність моделі розрізняти їх.
3. Незбалансованість класів: якщо сума елементів у кожному рядку (тобто, реальна кількість текстів кожного типу) сильно відрізняється, це вказує на незбалансованість класів у навчальних даних, що може впливати на ефективність моделі.
4. Загальна якість класифікації: сума діагональних елементів, поділена на загальну суму всіх елементів матриці, дає загальну точність класифікації для всіх класів.

На основі цієї інформації можна робити висновки про сильні та слабкі сторони моделі, і відповідно планувати кроки для її вдосконалення.

При виборі метрик для оцінки ефективності небінарної класифікації пропаганди, особливу увагу слід приділити використанню зваженого усереднення (Weighted Precision, Weighted Recall, Weighted F1), яке враховує кількість зразків у кожному класі. Це дозволить отримати більш збалансовану

оцінку ефективності моделі у випадку незбалансованих класів, що часто зустрічається в задачах виявлення пропаганди. Комплексний аналіз усіх вищезазначених метрик, в тому числі матриці помилок, дає змогу всебічно оцінити сильні та слабкі сторони розробленої системи, та окреслити шляхи її подальшого вдосконалення.

Процес оцінки ефективності моделі складатиметься з кількох етапів та включатиме порівняння з базовою моделлю, а також розділений аналіз якості розпізнавання пропаганди та якості класифікації пропагандистських технік.

4.2. Оцінка моделі в порівнянні з базовою моделлю

Розглянемо процес навчання моделі на перших 25 епохах, зображений на Рисунках 4.1 та 4.2 у вигляді графіків точності та втрат на тренувальному та валідаційному наборах даних.

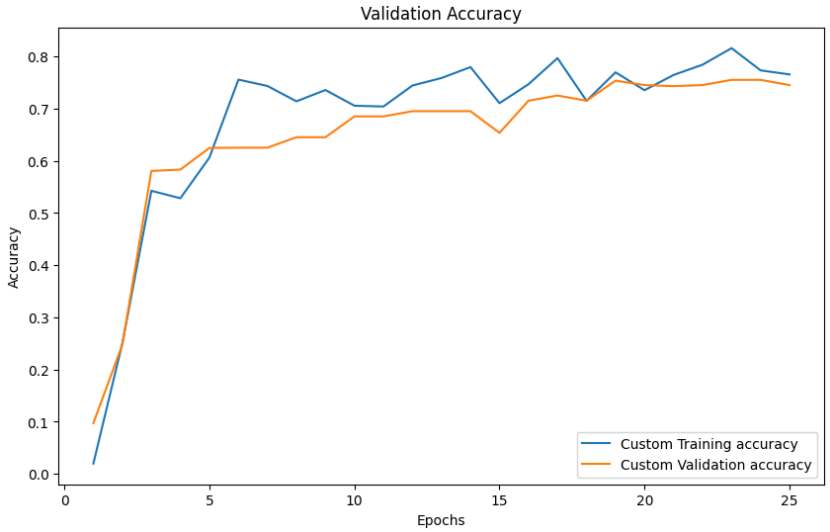

Рис. 4.1. Зміна точності Accuracy в процесі навчання

Графік точності демонструє стабільне зростання протягом перших 10-12 епох та вихід на плато з досягненням рівня точності близько 0.79-0.81, що вказує на досягнення оптимального рівня ефективності на даному наборі даних. Незначні коливання точності після виходу на плато можуть бути обумовлені

стохастичністю процесу оптимізації та наявністю складних прикладів у валідаційному наборі.

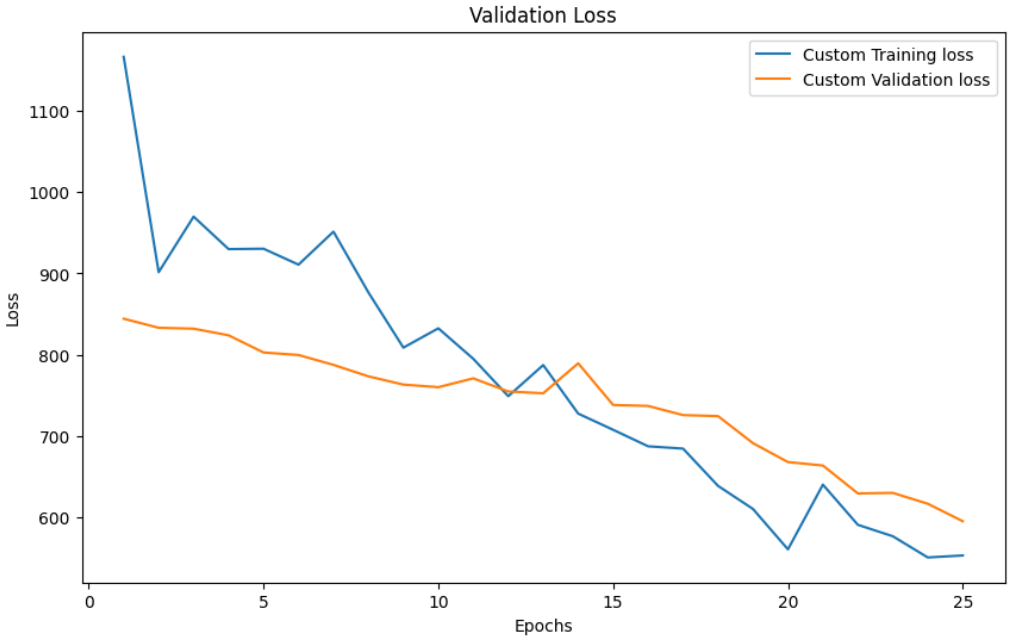

Рис. 4.2. Зміна втрат Loss в процесі навчання

Аналіз графіка втрат надає додаткову інформацію про процес оптимізації моделі. Значення втрат стабільно зменшуються протягом перших 15 епох, що свідчить про ефективну мінімізацію розбіжності між передбаченнями моделі та справжніми мітками класів. Після початкового зниження, втрати стабілізуються та демонструють повільніше зменшення, що узгоджується з виходом графіка точності на плато.

Важливо відзначити, що графіки точності та втрат не демонструють ознак перенавчання моделі, коли точність на валідаційному наборі починає знижуватись, а втрати зростати при подальшому навчанні. Це свідчить про здатність Custom моделі узагальнювати отримані знання та ефективно працювати на нових даних. Разом з тим, відносно високий рівень втрат після стабілізації може вказувати на потенціал для подальшого вдосконалення моделі. Загалом, аналіз графіків точності та втрат свідчить про успішне навчання Custom моделі для виявлення пропаганди в текстах. Модель демонструє здатність

ефективно покращувати свою точність та мінімізувати помилки в процесі навчання, досягаючи високого рівня ефективності на валідаційному наборі даних. Узгодженість динаміки точності та втрат, а також відсутність ознак перенавчання, підтверджують придатність обраної архітектури та методів оптимізації для поставленої задачі.

Для повнішої оцінки ефективності побудованої моделі порівняємо її результати з результатами базової моделі. У якості базової використано модель з простішою архітектурою, в даному випадку – Bidirectional LSTM з TimeDistributed шаром (Рисунок 4.3). Модель навчена на тому ж наборі даних та з використанням однакових параметрів (розмір словника, максимальна довжина послідовності, розмірність ембедінгів тощо).

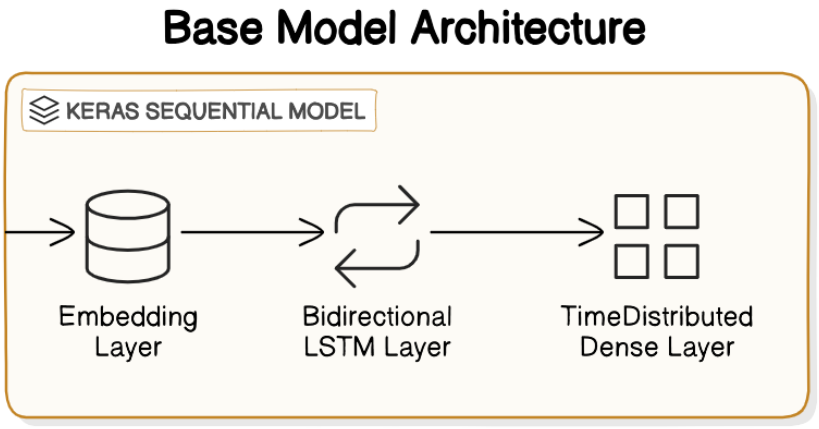

Рис. 4.3. Архітектура базової моделі

Порівняння якості CustomModel з базовою моделлю, наведене в Таблиці 4.1 дозволяє оцінити доцільність використання більш складної архітектури та її вплив на ефективність розпізнавання пропаганди. Отримані показники метрик дозволяють зробити висновок про значне покращення ефективності виявлення пропаганди в текстах завдяки використанню механізмів уваги та вдосконаленої архітектури.

Таблиця 4.1

Порівняння результатів розробленої моделі з базовою

	Accuracy	Presicion	Recall	F1-score
Base Model	0.7462	0.8738	0.5920	0.5617
Custom Model	0.8668	0.9747	0.6839	0.6532

Custom модель демонструє вищу загальну точність класифікації (86.68%) порівняно з базовою моделлю (74.62%), що свідчить про її здатність більш ефективно розпізнавати та класифікувати пропаганду. Крім того, має вищу точність (97.47%) порівняно з базовою моделлю (87.38%), що означає меншу кількість хибно позитивних результатів. Повнота Custom моделі (68.39%) також дещо вища за повноту базової моделі (59.20%), вказуючи на її здатність виявляти більшу частку текстів з пропагандою. Вища F1-міра Custom моделі (65.32%) порівняно з базовою моделлю (56.17%) підтверджує її кращу загальну ефективність у виявленні пропаганди.

Незважаючи на покращення, відносно невисокі значення повноти для обох моделей свідчать про потенціал для подальшого вдосконалення. Додатковий аналіз результатів Custom моделі для окремих класів пропаганди та розгляд помилок класифікації на конкретних прикладах дозволить виявити потенційні проблемні місця та напрямки для подальшого вдосконалення моделі.

4.3. Оцінка якості розпізнавання пропаганди та не-пропаганди

Наступна задача оцінки ефективності полягає в оцінюванні здатності моделі розрізняти пропагандистські та непропагандистські статті. Оскільки запропонований метод виявлення пропаганди у даних не включає окремого етапу бінарної класифікації на наявність або відсутність пропаганди, то для перевірки цієї здатності моделі було вирішено проаналізувати результати розпізнавання класу 18 ("non-propaganda") у порівнянні з агрегованими результатами для класів 0-17, кожен з яких представляє певну маніпулятивну

техніку. Таким чином, класи 0-17 були умовно об'єднані в один клас "propaganda" для проведення бінарної оцінки ефективності моделі.

```
4/4 [==============================] - 2s 498ms/step
Metrics for class 18:
Accuracy: 0.8403
Precision: 0.8847
Recall: 0.9454
F1-score: 0.9034
```

Рис. 4.4. Результат бінарної класифікації на наявність пропаганди

Результати на Рисунку 4.4 оцінки ефективності виявлення пропаганди на тестовій вибірці показують, що модель досягає точності 84% та F1-міри 90%, що свідчить про гарну результативність моделі у виявленні пропаганди.

4.4. Оцінка якості класифікації методів пропаганди

Розглянемо ефективність моделі в класифікації конкретних типів пропагандистських технік. Дана оцінка спрямована на визначення точності, з якою модель може виявляти та диференціювати різноманітні види пропаганди в текстових фрагментах. Для кожного класу маніпулятивних прийомів (класи 0-17) були окремо розраховані та проаналізовані метрики якості класифікації. Крім того, було проведено порівняльний аналіз результатів між різними типами пропаганди. Отримані значення метрик для кожного класу представлені в Таблиці 4.2.

Таблиця 4.2

Порівняння метрик якості класифікації для різних типів пропаганди

Номер класу	Назва класу	Accuracy	Precision	Recall	F1-score
0	'Black and White Fallacy'	0.9753	0.4860	0.9990	0.4860
1	'Reductio ad hitlerum',	0.9969	0.9999	0.7700	0.7700
2	'Slogans',	0.9977	0.9999	0.7800	0.7800
3	'Red Herring',	0.9992	0.9999	0.9340	0.9340
4	'Bandwagon',	0.9999	0.9999	0.9999	0.9999
5	'Obfuscation, Intentional Vagueness, Confusion',	0.9996	0.9998	0.9640	0.9640
6	'Whataboutism',	0.9927	0.9987	0.4460	0.4460
7	'Loaded Language',	0.9841	0.9899	0.3060	0.3060
8	'Appeal to fear-prejudice',	0.9776	0.9779	0.4140	0.4140
9	'Appeal to Authority',	0.9925	0.9989	0.4860	0.4860
10	'Exaggeration, Minimisation',	0.9912	0.9879	0.6000	0.6000
11	'Name Calling, Labeling',	0.9925	0.9999	0.5660	0.5660
12	'Thought-terminating Cliches',	0.9990	0.9985	0.9180	0.9180
13	'Straw Men',	0.9995	0.9999	0.9480	0.9480
14	'Doubt',	0.9825	0.9999	0.2280	0.2280
15	'Causal Oversimplification',	0.9930	0.9958	0.5780	0.5780
16	'Flag-Waving',	0.9908	0.9969	0.3940	0.3940
17	'Repetition',	0.9927	0.9999	0.6960	0.6960
18	'Non-propaganda'	0.8662	0.8835	0.9775	0.9274

Таким чином, значення метрик Accuracy, Precision, Recall та F1-score пораховано окремо для кожного класу. Проаналізуємо отримані результати. Модель демонструє достатньо високі середні показники за кожною з метрик, що свідчить про здатність розрізняти більшість наведених типів пропаганди та клас відсутності пропаганди. Для деяких класів, таких як 'Bandwagon', 'Red Herring',

'Thought-terminating Cliches' та 'Straw Men', значення усіх чотирьох метрик перевищують 0.9, що свідчить про високу повноту розпізнавання цих типів пропаганди. В той же час, для деяких класів 'Loaded Language', 'Appeal to fear-prejudice', 'Doubt' показники Recall та F1-score значно нижчі за середні, що вказує на проблеми з виявленням цих типів пропаганди та потребу в покращенні ефективності моделі. Це може бути пов'язано з недостатньою кількістю тренувальних даних для цих класів або з особливостями самих технік.

Розглянемо більш детально результати класифікації кожного класу на основі матриці плутанини представленої на Рисунку 4.5. Матриця демонструє, як часто випадки одного класу помилково класифікуються як інші класи, що дозволяє краще зрозуміти специфічні труднощі моделі. Аналізуючи матрицю разом з метриками якості з Таблиці 4.2, можна виділити такі спостереження:

- Діагональні елементи матриці, що представляють правильні передбачення для кожного класу, мають високі значення для більшості типів пропаганди. Це підтверджує високу точність (Accuracy) моделі, про яку свідчать метрики в таблиці.
- Класи з низькими показниками Recall та F1-score в таблиці, такі як 'Loaded Language', 'Appeal to fear-prejudice' та 'Doubt', мають значну кількість неправильних передбачень в матриці.
- Є певна плутанина між семантично пов'язаними термінами, наприклад "Repetition" частково передбачається як "Doubt", а "'Exaggeration, Minimisation'" як "Loaded_Language".
- Загалом, модель краще справляється з деякими чітко визначеними логічними помилками та прийомами пропаганди, але має труднощі з більш тонкими або неоднозначними категоріями.
- Клас 'Non-propaganda' має високу точність (Precision) і повноту (Recall), що підтверджується невеликою кількістю помилкових спрацьовувань в матриці. Однак, деякі типи пропаганди іноді неправильно класифікуються як 'Non-propaganda', що знижує загальну точність моделі.

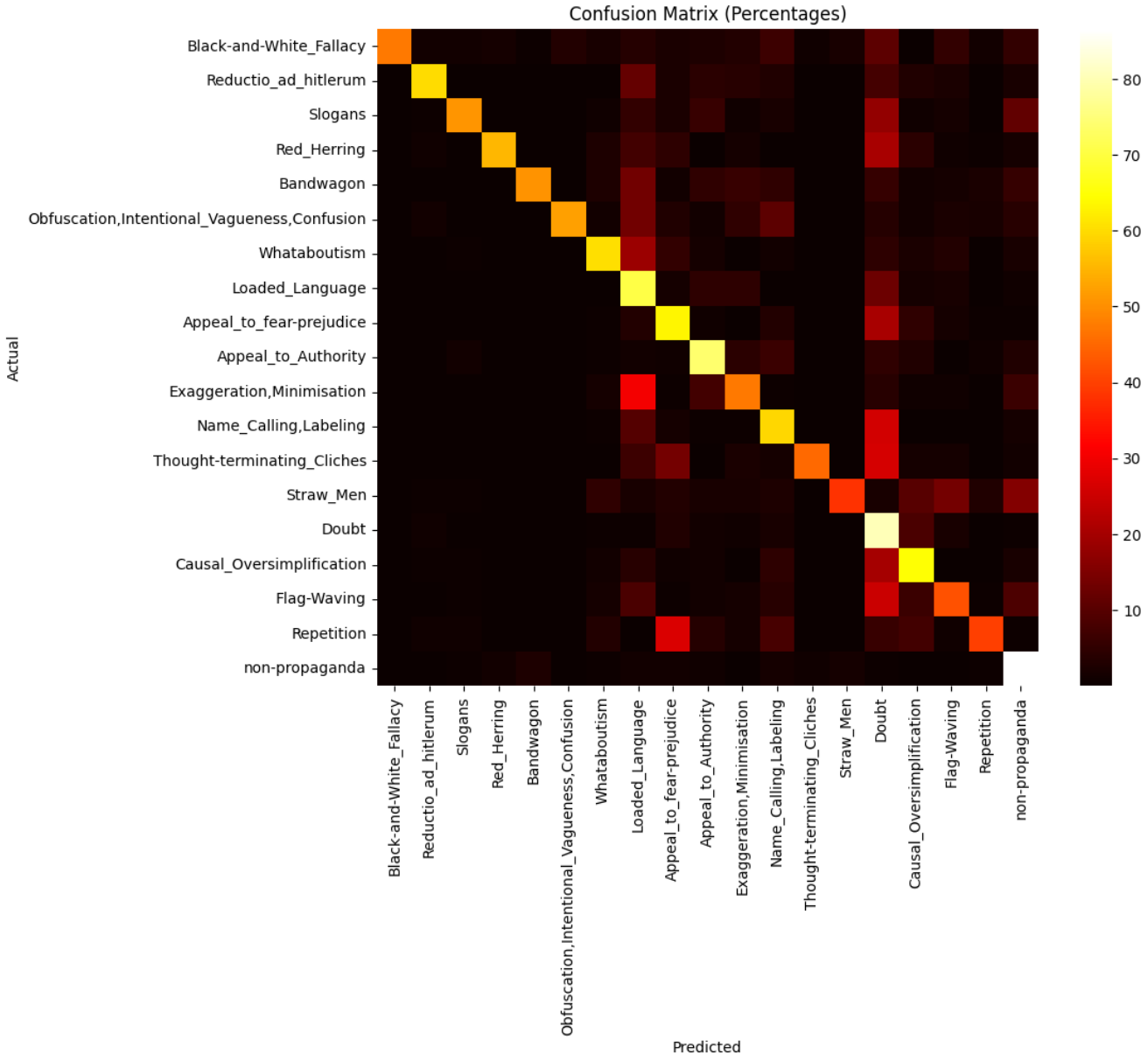

Рис. 4.5. Матриця плутанини для розробленої моделі

Загалом модель демонструє високу загальну ефективність у виявленні пропагандистських технік, але потребує покращення для деяких класів з низькою повнотою та F-мірою. Можливі шляхи вдосконалення можуть включати збільшення кількості навчальних даних для цих класів, зокрема з застосуванням методів аугментації даних, таких як перефразування або синонімізація, застосування методів боротьби з дисбалансом класів.

4.5. Результати застосування 3-рівневого механізму уваги

Одна з ключових компонент розробленої моделі CustomModel – механізм уваги, який працює на трьох різних рівнях обробки текстових даних: рівні окремих токенів (слів), рівні локальних патернів та рівні глобальних залежностей між токенами. Механізм BiLSTM-CNN-трансформер уваги додає власний унікальний контекст до обробки тексту на кожному з трьох рівнів. Розглянемо детальніше результати застосування запропонованого методу.

На рівні окремих токенів механізм уваги, реалізований через шар Bidirectional LSTM (self.bilstm_attention), дозволяє моделі зважувати важливість кожного слова в контексті послідовності. Цей шар забезпечує захоплення як попередньої, так і наступної контекстної інформації для кожного слова. Після проходження через двонапрямлену LSTM, кожен токен представляється вектором прихованого стану, який містить інформацію про його контекст з обох сторін. Шар уваги перетворює ці вектори в скалярні значення, які відображають релевантність кожного токена для задачі розпізнавання пропаганди. Це дає моделі можливість виділяти ключові слова та фрази, що вказують на наявність маніпулятивних технік, та підсилювати їх вплив на остаточне рішення. Результат застосування механізму уваги на словесному рівні до частини вхідної послідовності представлено на Рисунку 4.6.

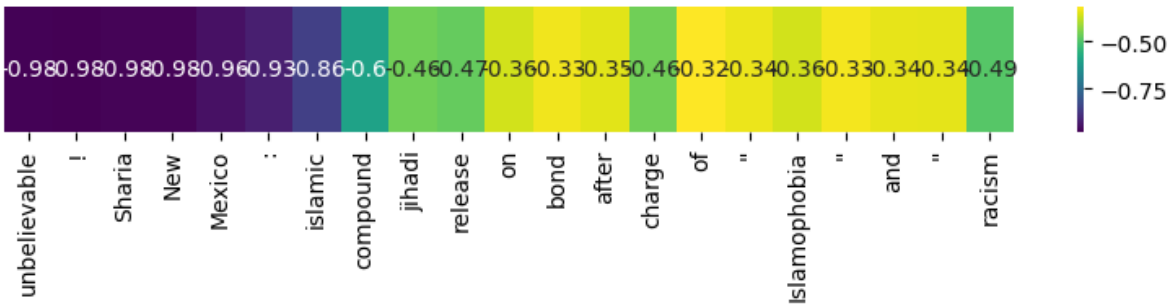

Рис. 4.6. Ваги уваги на рівні окремих токенів

На рівні локальних патернів механізм уваги (self.cnn_attention) застосовується до виходу згорткового шару (Conv1D), який виділяє локальні ознаки з послідовності. Кожен фільтр в згортковому шарі може спеціалізуватися на виявленні певних комбінацій слів або фраз, характерних для різних типів пропаганди. Увага на цьому рівні дозволяє моделі визначати, які з цих локальних патернів є найбільш інформативними для поточного прикладу, та посилювати їх вплив на класифікацію. Приклад результату механізму уваги рівня локального контексту зображено на Рисунку 4.7.

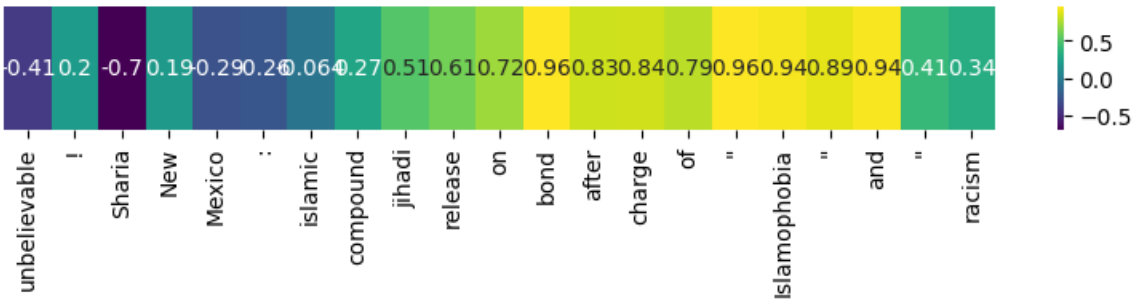

Рис. 4.7. Ваги уваги на рівні локальних патернів

На рівні глобальних залежностей між токенами механізм уваги реалізований через шар Transformer (self.transformer_layer). На відміну від LSTM та CNN, які обробляють послідовність локально, Transformer дозволяє моделі враховувати зв'язки між будь-якими парами токенів, незалежно від їх позиції в тексті. Увага на цьому рівні дозволяє моделі виявляти глобальні патерни та взаємозв'язки в тексті, які можуть вказувати на наявність маніпулятивних технік, навіть якщо ключові слова та фрази розкидані по послідовності. Рисунок 4.8 ілюструє результат застосування до вхідного тексту механізму уваги рівня глобальних патернів.

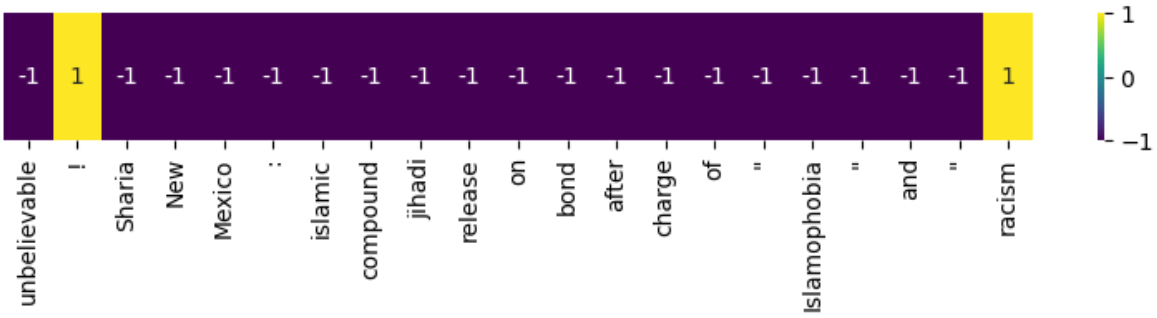

Рис. 4.8. Ваги уваги на рівні глобальних залежностей

Після обробки послідовності різними компонентами моделі (LSTM, CNN, Transformer), їх виходи об'єднуються в шарі Concatenate. Механізм уваги на цьому фінальному рівні, реалізований як Dense шар з softmax активацією, дозволяє моделі зважувати важливість ознак, отриманих з кожного компонента, для остаточної класифікації. Ця увага може інтерпретуватися як вибір найбільш інформативних представлень послідовності, отриманих на різних рівнях абстракції. Інтеграція цих представлень з урахуванням їх значимості дозволяє моделі приймати більш обґрунтовані рішення щодо наявності та типу пропаганди в тексті.

ВИСНОВКИ ДО РОЗДІЛУ 4

У даному розділі проведено експериментальне дослідження розробленої моделі CustomModel для виявлення пропаганди в текстових даних. Метою дослідження було оцінити ефективність запропонованої архітектури з використанням механізмів уваги на різних рівнях обробки тексту та порівняти її результати з базовою моделлю.

Аналіз процесу навчання CustomModel показав стабільне зростання точності та мінімізацію помилок на валідаційному наборі даних. Відсутність ознак перенавчання свідчить про здатність моделі узагальнювати отримані знання на нових даних.

Порівняння результатів розробленої моделі з базовою моделлю на тестовому наборі даних показало значне покращення ефективності виявлення пропаганди за всіма основними метриками. CustomModel досягла на 10.09% вищої точності Presicion. Показники повноти Recall (68.39% проти 59.20%) та F1-міри зросли на 9.19% та 9.15% відповідно, що підтверджує доцільність використання механізмів уваги та вдосконаленої архітектури для підвищення якості розпізнавання пропаганди.

Оцінка ефективності моделі в задачі бінарної класифікації на наявність пропаганди показала високу точність (84%) та F1-міру (90%), що свідчить про здатність моделі успішно відрізняти пропагандистські тексти від непропагандистських.

Аналіз результатів класифікації окремих типів пропаганди виявив, що модель демонструє високу ефективність для більшості класів. Особливо якісно запропонований метод спрацював для технік 'Bandwagon', 'Red Herring', 'Thought-terminating Cliches' та 'Straw Men', з показниками вище 0.9 за всіма метриками. Для деяких класів, таких як 'Loaded Language', 'Appeal to fear-prejudice' та 'Doubt', значення повноти та F1-міри є значно нижчими, що дає простір для подальших досліджень в покращенні ефективності моделі для цих типів пропаганди.

Дослідження запропонованого 3-рівневого механізму уваги показало, що він дозволяє моделі ефективно виявляти та зважувати важливість різних аспектів тексту на рівні окремих слів, локальних патернів та глобальних залежностей. Для аналізу роботи механізмів уваги було проведено візуалізацію та інтерпретацію ваг уваги на кожному з трьох рівнів. Ці візуалізації дозволили виявити, на яких словах, фразах та залежностях модель зосереджується при прийнятті рішень, а також порівняти патерни уваги для різних типів пропаганди та текстів без пропаганди. Результати аналізу показали, що увага на рівні токенів дозволяє моделі виділяти ключові слова та фрази, специфічні для кожного типу пропаганди, тоді як увага на рівні локальних патернів та глобальних залежностей допомагає враховувати більш складні та абстрактні ознаки маніпулятивних технік.

Загалом, результати експериментального дослідження підтверджують ефективність розробленої моделі глибинного навчання з використанням механізмів уваги для виявлення пропаганди в текстових даних. Модель продемонструвала значне покращення якості розпізнавання пропаганди в порівнянні з базовою моделлю та показала високі результати як в задачі бінарної класифікації, так і в розрізненні окремих типів пропагандистських технік.

Отримані результати свідчать про перспективність використання розроблених підходів та архітектури для автоматизованого виявлення пропаганди в текстових даних. Запропонована модель може стати основою для створення ефективних інструментів боротьби з дезінформацією та маніпуляціями в онлайн-просторі.

ВИСНОВКИ

У рамках даної дисертаційної роботи було проведено комплексне дослідження проблеми виявлення пропаганди в текстових даних з використанням методів глибинного навчання. У ході виконання роботи було досягнуто поставленої мети - розроблено метод виявлення пропаганди в текстових даних на основі глибинного навчання. Для досягнення мети було виконано всі поставлені завдання дослідження.

Проведено ґрунтовний аналіз теоретичних засад дослідження пропаганди та її основних методів, в ході якого було виявлено, що традиційні методи втрачають ефективність через великий обсяг даних, що генеруються щоденно. Здійснено огляд сучасних підходів до автоматичного розпізнавання пропаганди в текстових даних, виявлено їх переваги, недоліки та перспективні напрямки вдосконалення.

Аналіз існуючих датасетів (NLP4IF2019, QProp Dataset, FakeNewsNet Dataset, TSHP-17) показав необхідність якісної попередньої обробки тексту для ефективного навчання моделей.

Метод виявлення елементів пропаганди в текстових даних було розроблено на основі комбінації потужних архітектур глибинного навчання – Bidirectional LSTM, CNN, Transformer model – та ієрархічних механізмів уваги. Запропонована модель здатна розпізнавати 18 різних технік пропаганди та відділяти їх від фрагментів “не-пропаганди”, використовуючи механізми уваги для зосередження на найбільш 3 рівнях деталізації тексту: рівні окремих слів, речень та цілісних документів. Розроблена модель була реалізована за допомогою фреймворків TensorFlow та Keras, які дозволили ефективно експериментувати з архітектурами та гіперпараметрами моделей. Використання методів регуляризації, таких як рання зупинка та зменшення швидкості навчання, дозволило уникнути перенавчання та покращити узагальнюючу здатність моделі. Таким чином, розроблена модель досягла на 10.09% вищої точності та на 9.15% F1-міри порівняно з базовою моделлю.

Особливо високу ефективність модель продемонструвала у виявленні таких технік, як "Bandwagon", "Red Herring", "Thought-terminating Cliches" та "Straw Men", з показниками вище 0.9 за всіма метриками. Для деяких класів, таких як "Loaded Language", "Appeal to fear-prejudice" та "Doubt", значення повноти та F1-міри були нижчими, що вказує на необхідність подальшого вдосконалення моделі в цих напрямках.

Дослідження та візуалізація роботи запропонованого 3-рівневого механізму уваги окреслила ключові переваги його використання:

1. Запропонований механізм дозволяє моделі зосереджуватись на важливих частинах послідовності, що сприяє кращому захопленню релевантних особливостей і покращує продуктивність моделі.
2. Ваги уваги дають уявлення про те, які частини вхідних даних мають найбільший вплив на виходи моделі, що підвищує інтерпретованість моделі.
3. Увага допомагає ефективніше обробляти довгі послідовності, дозволяючи моделі зосереджуватись на ключових частинах послідовності, а не намагатися врахувати всю інформацію рівномірно.

Візуалізація та інтерпретація ваг уваги показали, що механізм дозволяє моделі ефективно виявляти та зважувати важливість різних аспектів тексту на рівні окремих слів, локальних патернів та глобальних залежностей.

Таким чином, розроблений метод виявлення пропаганди на основі глибинного навчання з використанням 3-рівневих механізмів уваги демонструє високу ефективність та перспективність використання для автоматизованого розпізнавання пропаганди в текстових даних. Запропонована модель може стати основою для створення потужних інструментів боротьби з дезінформацією та маніпуляціями в онлайн-просторі.

РЕКОМЕНДАЦІЇ ЩОДО ПОДАЛЬШИХ ДОСЛІДЖЕНЬ ТА ВИКОРИСТАННЯ РЕЗУЛЬТАТІВ

Враховуючи отримані результати та виявлені обмеження було визначено перелік рекомендацій щодо подальших досліджень та практичного застосування розробленого методу.

Рекомендації щодо подальших досліджень проблеми включають:

1. Розширення та збагачення навчальних даних:

   Обмеження обраного датасету, такі як фокус на англійській мові та відносно невелика кількість повних статей, можуть впливати на узагальненість результатів. Тому в майбутніх дослідженнях пропонується розглянути можливість розширення та доповнення навчальних даних зокрема шляхом аугментації, залучення текстів іншими мовами та врахування нових типів пропагандистських технік, залучення експертів для анотації додаткових даних, особливо для класів пропаганди, які потребують покращення ефективності моделі.

2. Вдосконалення архітектури моделі:

   Незважаючи на високу ефективність розробленої моделі, існує потенціал для подальшого покращення її архітектури. Рекомендується дослідити та порівняти ефективність інших архітектур глибинного навчання для виявлення пропаганди. Також доцільно експериментувати з різними конфігураціями та гіперпараметрами моделі, такими як розмір емебедінгів, кількість шарів, розмір вікна уваги тощо.

3. Інтеграція додаткових джерел інформації:

   Для підвищення точності виявлення пропаганди пропонується розглянути можливість включення метаданих, таких як автор, джерело, дата публікації, в процес навчання моделі. Це може надати додатковий контекст та покращити ефективність моделі. Крім того, пропонується дослідити інтеграцію мультимодальних даних, таких як зображення, відео, аудіо, що супроводжують текст. Врахування цих додаткових джерел інформації

може допомогти моделі краще зрозуміти контекст та підвищити точність виявлення пропаганди.

4. Адаптація до нових доменів та мов:

   Важливою є завдання розширення дослідження на інші мови, враховуючи лінгвістичні та культурні особливості при виявленні пропаганди. Це дозволить застосовувати модель до ширшого спектру текстових даних та підвищить її практичну цінність.

5. Інтерпретація та пояснення результатів:

   Для покращення довіри та прозорості моделі необхідно вдосконалити методи візуалізації та інтерпретації механізмів уваги. Це допоможе краще зрозуміти, як модель приймає рішення та на які аспекти тексту вона звертає увагу при виявленні пропаганди. Крім того, розробка підходів для генерації пояснень щодо класифікації тексту як пропагандистського, висвітлюючи ключові фрази та патерни, що вплинули на рішення моделі, підвищить довіру користувачів до результатів та полегшить їх інтерпретацію.

6. Інтеграція в системи моніторингу медіа:

   Для ефективного використання результатів дослідження пропонується впровадити розроблену модель в існуючі системи моніторингу новин, соціальних медіа та онлайн-контенту. Це дозволить автоматично виявляти та позначати потенційно пропагандистські матеріали, полегшуючи роботу модераторів та факт-чекерів. Налаштування сповіщень про виявлені випадки пропаганди допоможе оперативно реагувати та вживати необхідних заходів.

7. Розробка інструментів для користувачів:

   Для підвищення обізнаності користувачів про проблему пропаганди та надання їм можливості критично оцінювати інформацію, рекомендується створити плагіни для веб-браузерів або мобільні додатки, які використовують розроблену модель для аналізу текстового контенту в режимі реального часу. Ці інструменти можуть попереджати користувачів

про потенційну пропаганду та надавати додаткову інформацію для перевірки фактів. Інтеграція моделі в існуючі платформи факт-чекінгу та верифікації інформації також може допомогти користувачам приймати більш обґрунтовані рішення щодо достовірності контенту.

8. Застосування в освітніх та дослідницьких цілях:

   Розроблена модель та набори даних можуть бути цінними ресурсами для навчання студентів та дослідників методам виявлення пропаганди та критичного аналізу медіа. Створення інтерактивних інструментів та візуалізацій, які демонструють роботу моделі та механізмів уваги, може підвищити обізнаність про техніки пропаганди та сприяти розвитку навичок медіаграмотності.

Ці рекомендації та пропозиції охоплюють різні аспекти подальшого розвитку та застосування розробленого методу виявлення пропаганди в текстових даних. Їх реалізація сприятиме вдосконаленню моделі, розширенню її можливостей та підвищенню ефективності у реальних умовах. Це дозволить зробити вагомий внесок у боротьбу з дезінформацією та маніпуляціями в інформаційному просторі, сприяючи створенню більш здорового та достовірного медіа-середовища.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Shlok Gilda. 2017. Evaluating machine learning algorithms for fake news detection. In 2017 IEEE 15th Student Conference on Research and Development (SCOReD), pages 110–115. IEEE.
2. Alberto Barron-Cedeno, Giovanni Da San Martino, Israa Jaradat, and Preslav Nakov. 2019. Proppy: A system to unmask propaganda in online news. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 9847–9848.
3. Tundis, A.; Mukherjee, G., Mühlhäuser, M. 2021.An Algorithm for the Detection of Hidden Propaganda in Mixed-Code Text over the Internet. In Appl. Sci., [електронний ресурс] https://doi.org/10.3390/app11052196
4. Vitalii Danylyk, Victoria Vysotska. 2024. Information Technology for Detecting Fakes and Propaganda Based on Machine Learning and Sentiment Analysis
5. Francisco-Javier Rodrigo-Ginés, Jorge Carrillo-de-Albornoz and Laura Plaza. 2023. Hierarchical Modeling for Propaganda Detection: Leveraging Media Bias and Propaganda Detection Datasets.
6. JUSTDeep at NLP4IF 2019 Shared Task: Propaganda Detection using Ensemble Deep Learning Models
7. Edson C Tandoc Jr, Zheng Wei Lim, and Richard Ling. 2018. Defining fake news a typology of scholarly definitions. Digital journalism
8. Michael Orlov and Marina Litvak. 2018. Using behavior and text analysis to detect propagandists and misinformers on twitter. In Annual International Symposium on Information Management and Big Data
9. Akshay Jain and Amey Kasbe. 2018. Fake news detection. In 2018 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS), pages 1–5. IEEE
10. Stefan Helmstetter and Heiko Paulheim. 2018. Weakly supervised learning for fake news detection on twitter. In 2018 IEEE/ACM International Conference on

Advances in Social Networks Analysis and Mining (ASONAM), pages 274–277. IEEE.

11. Mykhailo Granik and Volodymyr Mesyura. 2017. Fake news detection using naive bayes classifier. In 2017 IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON)
12. Olena Gavrilenko, Yurii Oliinyk, and Hanna Khanko. 2019. Analysis of propaganda elements detecting algorithms in text data. In International Conference on Computer Science, Engineering and Education Applications, pages 438–447. Springer.
13. SuthanthiraDevi P., Karthika S., Sowmya K., Srinidhi S., Pavithra S. 2020. International Journal of Advanced Trends in Computer Science and Engineering [електронний ресурс] http://www.warse.org/IJATCSE/static/pdf/file/ijatcse155932020.pdf
14. Joachims T. “Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms” [Текст]: / T. Joachims // Kluwer Academic Publishers Norwell, MA, USA. – 2002. pp. 86
15. Ballard D.H. Computer Vision [Текст]: / D.H. Ballard, C.M. Brown C.M. // Prentice Hall Inc., 1982. – 539p
16. Mikolov T. Distributed Representations of Words and Phrases and their Compositionality [Текст]: / T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean // Proceeding NIPS 13 Proceedings of the 26th International Conference on Neural Information Processing Systems – Volume 1. – 2012. – pp. 3221-2340.
17. Mikolov T. Linguistic Regularities in Continuous Space Word Representations [Текст]: / T. Mikolov, W. Yih, G. Zweig // Proceedings of NAACLHLT 2013 – Atlanta, Georgia – 2013. Pp.520–523.
18. Zhang W. A comparative study of TF*IDF, LSI and multi-words for text classification [Текст]: / W. Zhang, T. Yoshida, X. Tang // Expert Systems with Applications - Volume 38 Issue 3 - 2011, pp. 2002-2012.

19. Ng, A. Y. and Jordan, M. I. (2002). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In NIPS 14, pp. 841–848

20. Sahlgren M. Usong bag-of-concept to improve the performance of support vector machines in text categorization [Текст]: / M. Sahlgren, R. Coster // Proc. of the 20th Int. Conf. on Computational Linguistics, Association for Computational Linguistics. 2004. p. 320.

ДОДАТОК

Методи глибинного навчання для виявлення елементів пропаганди у текстових даних

ЛІСТИНГ ПРОГРАМНГО КОДУ

Аркушів 15

Київ – 2024

```
import os
import requests
import zipfile

def download_glove_embeddings(destination='glove.6B.100d.txt'):
    if not os.path.exists(destination):
        print("Downloading GloVe embeddings...")
        url = "http://nlp.stanford.edu/data/glove.6B.zip"
        response = requests.get(url)
        zip_path = "glove.6B.zip"

        with open(zip_path, "wb") as f:
            f.write(response.content)

        with zipfile.ZipFile(zip_path, "r") as zip_ref:
            zip_ref.extractall()

        os.remove(zip_path)
        print("Download complete.")

download_glove_embeddings()

import numpy as np

def load_glove_embeddings(file_path):
    embeddings_index = {}
    with open(file_path, 'r', encoding='utf-8') as f:
        for line in f:
            values = line.split()
            word = values[0]
```

```
            coefs = np.asarray(values[1:], dtype='float32')
            embeddings_index[word] = coefs
    return embeddings_index

# Load GloVe embeddings
glove_embeddings = load_glove_embeddings('glove.6B.100d.txt')

def create_embedding_matrix(word_index, embeddings_index, embedding_dim):
    embedding_matrix = np.zeros((len(word_index) + 1, embedding_dim))
    for word, i in word_index.items():
        embedding_vector = embeddings_index.get(word)
        if embedding_vector is not None:
            embedding_matrix[i] = embedding_vector
    return embedding_matrix

import os
import nltk
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.corpus import stopwords
# from nltk.stem import WordNetLemmatizer
import numpy as np
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer

nltk.download('punkt')
# nltk.download('stopwords')
nltk.download('wordnet')
```

```
    stemmer = PorterStemmer()
    # stop_words = set(stopwords.words('english'))

    import re
    import string

    # def lemmatize_text(text):
    #     words = word_tokenize(text)
    #     # words = [lemmatizer.lemmatize(word.lower()) for word in words if
word.isalnum() and word.lower() not in stop_words]
    #     words = [lemmatizer.lemmatize(word.lower()) for word in words if
word.isalnum()]
    #     return ' '.join(words)
    uniq_labels =['Black-and-White_Fallacy',
      'Reductio_ad_hitlerum',
      'Slogans',
      'Red_Herring',
      'Bandwagon',
      'Obfuscation,Intentional_Vagueness,Confusion',
      'Whataboutism',
      'Loaded_Language',
      'Appeal_to_fear-prejudice',
      'Appeal_to_Authority',
      'Exaggeration,Minimisation',
      'Name_Calling,Labeling',
      'Thought-terminating_Cliches',
      'Straw_Men',
      'Doubt',
      'Causal_Oversimplification',
      'Flag-Waving',
```

```
    'Repetition',
    'non-propaganda']

def normalize_text(text):
    # Convert to lowercase and remove non-printable characters
    text = text.lower()
    # Remove newline characters
    # text = text.replace("\n", " ").replace("\r", " ")
    # Remove punctuation
    text = re.sub(f"[{re.escape(string.punctuation)}]", "", text)
    # # Remove extra spaces
    # text = re.sub(r'\s+', ' ', text).strip()
    # print(text)
    # if (text == ''):
    #   print(text, '0')
    #   return '0'
    # if (text == ' '):
    #   print(text, '0')
    #   return '0'
    return text

def stem_text(word):
    return stemmer.stem(word)

def lemmatize_text(word):
    words = word_tokenize(word)
    # words = [lemmatizer.lemmatize(word.lower()) for word in words if
word.isalnum() and word.lower() not in stop_words]
    return lemmatizer.lemmatize(word.lower())
```

```
    def uniq(arr):
      unique_elements = list(set(element for sublist in arr for element in sublist))
      return unique_elements

    class NLP4IF_Dataset:
        def __init__(self, data_path):
            self.data_path = data_path
            self.articles, self.labels, self.label_ranges, self.row_articles, self.y =
self.load_data()
            self.tokenizer = Tokenizer()
            self.word_index = self.create_tokenizer()
            self.max_sequence_length = 100  # or any suitable length
            # self.embeddings_index = self.load_glove_embeddings()
            # self.embedding_matrix = self.create_embedding_matrix()

        def load_data(self):
            row_articles = []
            articles = []
            labels = []
            # row_label_ranges = []
            label_ranges = []

            for filename in os.listdir(self.data_path):
                if filename.endswith(".txt"):
                    article_id = filename.split(".")[0]
                    labels_file = article_id + ".labels.tsv"
                    with open(os.path.join(self.data_path, filename), "r", encoding="utf-8")
as f:
                        article_text = f.read()
                        article_text = article_text.replace("\n", " ").replace("\r", " ")
```

```
                row_articles.append(article_text)
                article_text_preproc = ' '.join([stem_text(t) for t in
article_text_normal])
                articles.append(article_text_preproc)
            with open(os.path.join(self.data_path, labels_file), "r", encoding="utf-
8") as f:
                article_labels = []
                article_label_ranges = []
                for line in f:
                    _, label, start, end = line.strip().split("\t")
                    article_labels.append(label)
                    article_label_ranges.append((int(start), int(end)))
                labels.append(article_labels)
                label_ranges.append(article_label_ranges)
        y = []
        for i, article in enumerate(row_articles):
          indexes = label_ranges[i]
          words_set = article.split(' ')
          yi = [uniq_labels.index('non-propaganda') for _ in range(len(words_set))]
          for ind, label in zip(indexes, labels[i]):
            start, end = ind
            symbol_count = 0
            for wi, word in enumerate(words_set):
              if int(symbol_count + len(word)/2) in range(start, end+1): yi[wi] =
uniq_labels.index(label)
              symbol_count += len(word)
              symbol_count += 1
          y.append(yi)
        return articles, labels, label_ranges, row_articles, y
```

```
    def create_tokenizer(self):
        self.tokenizer.fit_on_texts(self.articles)
        return self.tokenizer.word_index

    def load_glove_embeddings(self, glove_path='glove.6B.100d.txt'):
        embeddings_index = {}
        with open(glove_path, 'r', encoding='utf-8') as f:
            for line in f:
                values = line.split()
                word = values[0]
                coefs = np.asarray(values[1:], dtype='float32')
                embeddings_index[word] = coefs
        return embeddings_index

    def create_embedding_matrix(self, embedding_dim=100):
        embedding_matrix = np.zeros((len(self.word_index) + 1, embedding_dim))
        for word, i in self.word_index.items():
            embedding_vector = self.embeddings_index.get(word)
            if embedding_vector is not None:
                embedding_matrix[i] = embedding_vector
        return embedding_matrix

    def preprocess_data(self):
        sequences = self.tokenizer.texts_to_sequences(self.articles)
        word_data = pad_sequences(sequences, maxlen=self.max_sequence_length)
        return word_data

def load_dataset(train_path, test_path):
    train_dataset = NLP4IF_Dataset(train_path)
    test_dataset = NLP4IF_Dataset(test_path)
```

```
    return train_dataset, test_dataset

# Load training and test datasets
train_path = 'train'
test_path = 'test'
train_dataset, test_dataset = load_dataset(train_path, test_path)

import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input, Embedding, Bidirectional, LSTM,
TimeDistributed, Dense, Conv1D, Concatenate, Multiply, MultiHeadAttention
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow_addons.text.crf import crf_log_likelihood
import matplotlib.pyplot as plt


Update the model definition
class CustomModel(Model):
    def __init__(self, embedding_matrix, max_len, n_classes, *args, **kwargs):
        super(CustomModel, self).__init__(*args, **kwargs)
        self.embedding_dim = embedding_matrix.shape[1]
        self.max_len = max_len
        self.n_classes = n_classes

        self.embedding = Embedding(input_dim=embedding_matrix.shape[0],
                                   output_dim=self.embedding_dim,
                                   weights=[embedding_matrix],
                                   input_length=max_len,
```

```
                                trainable=False,
                                mask_zero=True)
    # class CustomModel(Model):
    #     def __init__(self, vocab_size, embedding_dim, max_len, n_classes, *args,
**kwargs):
    #         super(CustomModel, self).__init__(*args, **kwargs)
    #         self.vocab_size = vocab_size
    #         self.embedding_dim = embedding_dim
    #         self.max_len = max_len
    #         self.n_classes = n_classes

    #         self.embedding = Embedding(input_dim=vocab_size,
output_dim=embedding_dim, mask_zero=True)
            self.bilstm = Bidirectional(LSTM(units=64, return_sequences=True,
recurrent_dropout=0.2))
            self.bilstm_attention = TimeDistributed(Dense(1, activation='tanh'))
            self.cnn = Conv1D(filters=64, kernel_size=9, padding='same',
activation='relu')
            self.cnn_attention = TimeDistributed(Dense(1, activation='tanh'))
            self.transformer_layer = MultiHeadAttention(num_heads=4, key_dim=64)
            self.transformer_attention = TimeDistributed(Dense(1, activation='tanh'))
            self.concatenate = Concatenate(axis=2)
            self.output_layer = TimeDistributed(Dense(n_classes, activation="softmax"))

            self.transition_params = None

        def build(self, input_shape):
            self.transition_params = self.add_weight(shape=(self.n_classes,
self.n_classes),
                                                     initializer='glorot_uniform',
```

```
                                    name='transition_params')
        super(CustomModel, self).build(input_shape)

    def call(self, inputs):
        x = self.embedding(inputs)
        bilstm_output = self.bilstm(x)
        bilstm_attention = self.bilstm_attention(bilstm_output)
        bilstm_attention = tf.nn.softmax(bilstm_attention, axis=1)
        bilstm_attention = Multiply()([bilstm_output, bilstm_attention])

        cnn_output = self.cnn(x)
        cnn_attention = self.cnn_attention(cnn_output)
        cnn_attention = tf.nn.softmax(cnn_attention, axis=1)
        cnn_attention = Multiply()([cnn_output, cnn_attention])

        transformer_output = self.transformer_layer(x, x)
        transformer_attention = self.transformer_attention(transformer_output)
        transformer_attention = tf.nn.softmax(transformer_attention, axis=1)
        transformer_attention = Multiply()([transformer_output,
transformer_attention])

        combined = self.concatenate([bilstm_attention, cnn_attention,
transformer_attention])
        outputs = self.output_layer(combined)

        return outputs, bilstm_attention, cnn_attention, transformer_attention

    def compute_loss(self, y_true, y_pred):
        y_true = tf.cast(y_true, dtype=tf.int64)
```

```
            sequence_lengths = tf.reduce_sum(tf.cast(tf.not_equal(y_true, 0), tf.int64), axis=-1)
            loss, _ = crf_log_likelihood(y_pred, y_true, sequence_lengths, transition_params=self.transition_params)
            return tf.reduce_mean(-loss)

        def train_step(self, data):
            x, y_true = data
            with tf.GradientTape() as tape:
                y_pred, bilstm_attention, cnn_attention, transformer_attention = self(x, training=True)
                loss = self.compute_loss(y_true, y_pred)
            gradients = tape.gradient(loss, self.trainable_variables)
            self.optimizer.apply_gradients(zip(gradients, self.trainable_variables))

            y_pred_labels = tf.argmax(y_pred, axis=-1)
            y_true = tf.cast(y_true, tf.int64)
            mask = tf.cast(tf.not_equal(y_true, 0), tf.float32)
            accuracy = tf.reduce_sum(tf.cast(tf.equal(y_true, tf.cast(y_pred_labels, tf.int64)), tf.float32) * mask) / tf.reduce_sum(mask)

            return {'loss': loss, 'accuracy': accuracy}

        def test_step(self, data):
            x, y_true = data
            y_pred, bilstm_attention, cnn_attention, transformer_attention = self(x, training=False)
            loss = self.compute_loss(y_true, y_pred)

            y_pred_labels = tf.argmax(y_pred, axis=-1)
```

```
        y_true = tf.cast(y_true, tf.int64)
        mask = tf.cast(tf.not_equal(y_true, 0), tf.float32)
        accuracy = tf.reduce_sum(tf.cast(tf.equal(y_true, tf.cast(y_pred_labels, tf.int64)), tf.float32) * mask) / tf.reduce_sum(mask)

        return {'loss': loss, 'accuracy': accuracy}

    # Build the model
    max_words = 10000
    max_len = 500  # Maximum length of the word sequences
    embedding_dim = 100  # Dimension of word embeddings

    # Tokenize texts and create sequences
    tokenizer = keras.preprocessing.text.Tokenizer(num_words=max_words)

    # Create embedding matrix
    embedding_matrix = create_embedding_matrix(tokenizer.word_index, glove_embeddings, embedding_dim)

    # tokenizer.fit_on_texts(train_dataset.articles)
    # X_train_seq = tokenizer.texts_to_sequences(train_dataset.articles)
    tokenizer.fit_on_texts(X)
    X_train_seq = tokenizer.texts_to_sequences(X)
    X_train_pad = keras.preprocessing.sequence.pad_sequences(X_train_seq, maxlen=max_len)

    # Encode labels
    # label_encoder = LabelEncoder()
    # label_encoder.fit(range(len(uniq_labels)))
    # y_train_encoded = [label_encoder.transform(labels) for labels in train_dataset.y]
```

```
    # y_train_pad = keras.preprocessing.sequence.pad_sequences(train_dataset.y, maxlen=max_len)
    y_train_pad = keras.preprocessing.sequence.pad_sequences(y, maxlen=max_len)

    # model = CustomModel(max_words, embedding_dim=64, max_len=max_len, n_classes=19)
    model = CustomModel(embedding_matrix, max_len=max_len, n_classes=19)

    optimizer = Adam(learning_rate=0.0007)
    model.compile(optimizer=optimizer, loss='sparse_categorical_crossentropy', metrics=['accuracy'])



    # Add callback functions for EarlyStopping and ReduceLROnPlateau
    early_stopping = EarlyStopping(monitor='val_loss', patience=3, restore_best_weights=True)
    reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.1, patience=2, min_lr=0.0005)

    # Train the model
    history = model.fit(
        X_train_pad,
        y_train_pad,
        batch_size=128,
        epochs=25,
        validation_split=0.1,
        verbose=1,
        callbacks=[early_stopping, reduce_lr]
    )
```

```
import matplotlib.pyplot as plt

# Визначення та навчання базової моделі (Bidirectional LSTM)
base_model = keras.Sequential([
    Embedding(input_dim=max_words, output_dim=embedding_dim, mask_zero=True),
    Bidirectional(LSTM(units=32, return_sequences=True, recurrent_dropout=0.1)),
    TimeDistributed(Dense(19, activation="softmax"))
])

optimizer = Adam(learning_rate=0.0007)
base_model.compile(optimizer=optimizer, loss='sparse_categorical_crossentropy', metrics=['accuracy'])
# base_model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

# X_train_seq = tokenizer.texts_to_sequences(X)
# X_train_pad = keras.preprocessing.sequence.pad_sequences(X_train_seq, maxlen=max_len)
# y_train_pad = keras.preprocessing.sequence.pad_sequences(y, maxlen=max_len)

base_history = base_model.fit(
    X_train_pad,
    y_train_pad,
    batch_size=128,
    epochs=25,
    validation_split=0.1,
    verbose=1,
```

```
    callbacks=[early_stopping, reduce_lr]
)

def plot_history(histories, names, metric='accuracy'):
    plt.figure(figsize=(10, 6))

    for history, name in zip(histories, names):
        train_metric = history.history[metric]
        val_metric = history.history[f'val_{metric}']
        epochs = range(1, len(train_metric) + 1)

        plt.plot(epochs, train_metric, label=f'{name} Training {metric}')
        plt.plot(epochs, val_metric, label=f'{name} Validation {metric}')

    plt.title(f'Training and Validation {metric.capitalize()}')
    plt.xlabel('Epochs')
    plt.ylabel(metric.capitalize())
    plt.legend()

    plt.show()

plot_history([base_history, history], names=[' Base', ' Custom'], metric='accuracy')

plot_history([base_history, history], names=[' Base', ' Custom'], metric='loss')
```