МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра інформаційно-обчислювальних систем і управління

СОЯ Мар'яна Юріївна

Модуль виявлення фейкових новин на основі ключових слів /Module for fake news detection based on keywords

Спеціальність 122 – Комп'ютерні науки
Освітньо-професійна програма – Комп'ютерні науки

Кваліфікаційна робота

Виконав студент групи КН-42
М.Ю Соя

_____

Науковий керівник:
д.т.н., доцент Х.В. Ліп'яніна-Гончаренко

_____

Кваліфікаційну роботу допущено до захисту
«___»_____ 2025 р.

В.о. завідувача кафедри
_____ Н.М. Васильків

Тернопіль – 2025

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

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ Н.М. Васильків
«____» _____2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Соя Мар'яна Юріївна
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: Модуль виявлення фейкових новин на основі ключових слів /Module for fake news detection based on keywords, керівник роботи к.т.н., доцент Х.В. Ліп'яніна-Гончаренко
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від 20 грудня 2024 р. № 938.

2. Строк подання студентом закінченої кваліфікаційної роботи 25 травня 2025 р.

3. Вихідні дані до кваліфікаційної роботи: завдання на кваліфікаційну роботу студента, наукові статті, технічна література.

4. Основні питання, які потрібно розробити:

- Аналіз існуючих методів виявлення фейкових новин та оцінка їх переваг і недоліків.
- Формалізація математичної моделі для виявлення фейкових новин на основі ключових слів.
- Розробка алгоритму для видобутку ключових слів та їх класифікації.
- Реалізація програмного модуля для виявлення фейкових новин із застосуванням сучасних методів машинного навчання.
- Тестування та оцінка точності моделі на основі реальних даних.
- Оцінка ефективності запропонованого рішення та аналіз можливих шляхів його вдосконалення.

5. Перелік графічного матеріалу в роботі:

- діаграма архітектури системи виявлення фейкових новин;
- схема алгоритму попередньої обробки текстів;
- скріншоти результатів класифікації новин, графіків точності та таблиць ефективності моделей.

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 20 грудня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Затвердження теми кваліфікаційної роботи, ознайомлення з літературними джерелами та складання плану роботи	до 01.01. 2025 р.	
2	Написання 1 розділу кваліфікаційної роботи	до 01.02. 2025 р.	
3	Написання 2 розділу кваліфікаційної роботи	до 01.04.2025 р.	
4	Написання 3 розділу кваліфікаційної роботи	до 01.05. 2025 р.	
5	Представлення попереднього варіанту кваліфікаційної роботи, перевірка та внесення змін керівником	до 15.05.2025 р.	
6	Опрацювання зауважень та представлення завершеного варіанту кваліфікаційної роботи. Підготовка супроводжуючих документів	до 25.05.2025 р.	
	Перевірка кваліфікаційної роботи на оригінальність тексту	до 30.05.2025 р.	
	Оформлення кваліфікаційної роботи та отримання допуску до захисту	до 10.06.2025 р.	
	Подання кваліфікаційної роботи до захисту на засіданні атестаційної комісії	до 10.06. 2025 р.	

Студент_____ М. Ю. Соя
(підпис) (прізвище та ініціали)

Керівник кваліфікаційної роботи_____ Х. В. Ліп’яніна-Гончаренко
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота на тему «Модуль виявлення фейкових новин на основі ключових слів» на здобуття освітнього ступеня «бакалавр» за спеціальністю 122 «Комп'ютерні науки» освітньої програми «Комп'ютерні науки» викладена на 66 сторінках і містить 21 ілюстрацій, 1 таблиця і 28 використаних джерел.

Метою роботи є розробка комбінованого підходу до виявлення фейкових новин, що поєднує методи витягу ключових слів (TF-IDF, RAKE, YAKE!, LDA, LSA, TextRank) та алгоритми машинного навчання з використанням ансамблевих класифікаторів (Voting, Stacking).

Методи дослідження включають: аналіз сучасних підходів до класифікації текстів, побудову інформаційної моделі, попередню обробку тексту (токенізація, лематизація, видалення стоп-слів), векторизацію текстів, витяг ключових слів, навчання моделей класифікації (Random Forest, Logistic Regression, SVM) та оцінку точності класифікації за допомогою метрик точності, повноти та F1-міри.

У результаті реалізовано програмний модуль на мові Python із використанням бібліотек scikit-learn, NLTK, pandas, matplotlib, що дозволяє завантажувати новини у форматі CSV, автоматично витягувати ключові ознаки та здійснювати класифікацію з точністю понад 92%. Результати виводяться у вигляді графіків, таблиць і звітів, що підвищує зручність використання.

Результати дослідження можуть бути застосовані в галузях інформаційної безпеки, журналістики, фактчекінгу та розробки інструментів для боротьби з дезінформацією.

Ключові слова: ВИЯВЛЕННЯ ФЕЙКОВИХ НОВИН, ОБРОБКА ПРИРОДНОЇ МОВИ, КЛАСИФІКАЦІЯ ТЕКСТУ, TF-IDF, RAKE, YAKE!, АНСАМБЛІ КЛАСИФІКАТОРІВ, FAKE NEWS DETECTION, NLP, MACHINE LEARNING.

ANNOTATION

The qualification thesis titled "Module for Fake News Detection Based on Keywords" for obtaining the Bachelor's degree in specialty 122 "Computer Science", educational program "Computer Science", is presented on 66 pages and contains 21 illustrations, 1 table, and 28 references.

The purpose of the thesis is to develop a combined approach for fake news detection that integrates keyword extraction methods (TF-IDF, RAKE, YAKE!, LDA, LSA, TextRank) and machine learning algorithms using ensemble classifiers (Voting, Stacking).

The research methods include: analysis of modern approaches to text classification, construction of an information model, preprocessing of text (tokenization, lemmatization, stop word removal), text vectorization, keyword extraction, training of classification models (Random Forest, Logistic Regression, SVM), and evaluation of classification accuracy using precision, recall, and F1-score metrics.

As a result, a software module was implemented in Python using scikit-learn, NLTK, pandas, matplotlib libraries, which enables loading news in CSV format, automatic extraction of key features, and classification with an accuracy exceeding 92%. The results are displayed in the form of graphs, tables, and reports, which enhances usability.

The research results can be applied in the fields of information security, journalism, fact-checking, and development of tools for combating disinformation.

Keywords: FAKE NEWS DETECTION, NATURAL LANGUAGE PROCESSING, TEXT CLASSIFICATION, TF-IDF, RAKE, YAKE!, ENSEMBLE CLASSIFIERS, NLP, MACHINE LEARNING.

ЗМІСТ

ВСТУП

У сучасному інформаційному середовищі фейкові новини стали однією з найбільших загроз для суспільства, оскільки вони здатні маніпулювати громадською думкою, підривати довіру до органів влади та сприяти політичним кризам. З розвитком цифрових технологій і соціальних медіа фейкові новини розповсюджуються з великою швидкістю, часто досягаючи значної аудиторії. Це створює серйозні виклики для інформаційної безпеки, особливо в контексті маніпуляцій під час виборчих кампаній та політичних криз. Зважаючи на це, необхідність створення автоматизованих систем для виявлення фейкових новин стає актуальним завданням для забезпечення достовірності інформації та стабільності соціальних і політичних процесів.

Визначення та класифікація фейкових новин є складною задачею через їх схожість з реальними новинами, що включають маніпуляції з фактами, перебільшення або неповні дані. Для боротьби з дезінформацією важливо використовувати сучасні технології, такі як машинне навчання та обробка природної мови, які дозволяють автоматизувати процес виявлення фейкових новин на основі лексичних, граматичних та семантичних характеристик текстів. Важливою проблемою є також розробка ефективних методів, які б дозволяли досягти високої точності класифікації навіть у разі змін в стилістиці та контексті новин.

Особливу увагу в дослідженні слід приділити комбінованим підходам, що поєднують кілька методів для покращення точності класифікації новин, зокрема використання різних технік видобутку ключових слів, векторизації тексту та алгоритмів машинного навчання. Інтеграція таких методів дозволить створити адаптивну систему, здатну ефективно реагувати на нові форми фейкових новин. Такі системи мають великий потенціал для автоматизації процесу перевірки фактів і зниження впливу фейкових новин на громадську свідомість.

Розробка ефективних методів виявлення фейкових новин не лише сприятиме покращенню інформаційної безпеки, але й допоможе створити

інструменти для навчання громадян медіаграмотності, що є важливим кроком у боротьбі з дезінформацією. Завдяки використанню новітніх технологій у поєднанні з аналітичними підходами можна створити ефективні системи, які будуть активно сприяти покращенню загальної ситуації в інформаційному просторі, забезпечуючи більш високу точність класифікації новин.

Метою роботи є розробка нового комбінованого підходу до виявлення фейкових новин на основі ключових слів.

Завдання дослідження:

1. Аналіз існуючих методів виявлення фейкових новин та оцінка їх переваг і недоліків.
2. Формалізація математичної моделі для виявлення фейкових новин на основі ключових слів.
3. Розробка алгоритму для видобутку ключових слів та їх класифікації.
4. Реалізація програмного модуля для виявлення фейкових новин із застосуванням сучасних методів машинного навчання.
5. Тестування та оцінка точності моделі на основі реальних даних.
6. Оцінка ефективності запропонованого рішення та аналіз можливих шляхів його вдосконалення.

Предмет роботи – це процес виявлення фейкових новин на основі аналізу текстових даних.

Об'єкт роботи – модуль для автоматизованого виявлення фейкових новин, який використовує методи машинного навчання та текстового аналізу.

Методи дослідження включають аналіз існуючих підходів до виявлення фейкових новин, застосування методів машинного навчання (зокрема Random Forest), видобуток ключових слів за допомогою методів TF-IDF, RAKE, Yake!, а також використання латентного семантичного аналізу (LSA) та тематичного моделювання (LDA).

Апробація результатів дослідження. Основні теоретичні та практичні результати дослідження були представлені на міжнародних наукових конференціях, матеріали яких опубліковано та проіндексовано в базі Scopus:

1. In Proceedings of the 2024 5th IEEE KhPI Week on Advanced Technology (KhPIWeek 2024), Kharkiv, Ukraine, 7–11 October 2024. IEEE.
2. 1st International Workshop on Intelligent and CyberPhysical Systems (ICyberPhyS 2024), Khmelnytskyi, Ukraine, 28 June 2024.
3. 7th International Workshop on Computer Modeling and Intelligent Systems (CMIS 2024), Zaporizhzhia, Ukraine, 3 May 2024.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧІ РІШЕННЯ

1.1 Визначення фейковості новин та їх класифікація

Визначення фейковості новин є складною задачею, оскільки фейкові новини часто маскуються під правдиві, використовуючи подібний стиль, мову та структуру. Зазвичай фейкові новини можуть мати елементи перебільшення, маніпулювання фактами або навмисного викривлення подій, що сприяє дезінформації. Оскільки достовірність джерел і точність інформації можуть бути важко оцінити вручну, на допомогу приходять сучасні технології аналізу текстів, які дозволяють автоматично виявляти фейкові новини на основі певних характеристик тексту. Для цього використовуються методи, що враховують лексичні, граматичні, а також семантичні особливості тексту [1].

Одним із основних підходів до класифікації новин є застосування алгоритмів машинного навчання, таких як класифікація текстів. Вони дозволяють автоматично навчатися на великих наборах даних, щоб визначити різницю між правдивими та фейковими новинами. Основними особливостями, на які спираються ці методи, є лексичний аналіз, видобуток ключових слів, а також аналіз стилістичних характеристик тексту. Наприклад, фейкові новини часто містять конкретні фрази або терміни, що порушують логічну послідовність або емоційно забарвлені, що робить їх більш помітними для алгоритмів [2].

Методи машинного навчання, зокрема такі як Random Forest, допомагають класифікувати новини шляхом визначення ознак, які характерні для фейкових або реальних новин. Для виявлення фейкових новин використовуються методи видобутку ключових слів, як-от TF-IDF, RAKE, Yake! та інші, які дозволяють виділяти значущі терміни, що є індикаторами фейковості. Вони аналізують текст і виявляють слова або фрази, які найчастіше зустрічаються в фейкових новинах. Оскільки ці методи надають важливу інформацію щодо структури та вмісту новини, їх комбінація з алгоритмами машинного навчання дозволяє покращити точність класифікації [3].

Крім того, важливим аспектом є використання таких методів, як латентний семантичний аналіз (LSA) та латентне розподілене зображення (LDA), що дозволяють глибше зрозуміти семантичні зв'язки між словами в тексті. Вони дозволяють виявити латентні теми та тренди, що можуть бути характерними для фейкових новин і тим самим покращити класифікацію. Ці методи дозволяють створити більш точні прогнози щодо надійності новини та можуть бути важливими інструментами для подальшого вдосконалення моделі виявлення фейкових новин [4].

Система виявлення фейкових новин, побудована на основі таких підходів, дозволяє автоматизувати процес аналізу та класифікації новин, що значно знижує потребу в людському втручанні. Завдяки інтеграції різних методів машинного навчання та обробки природної мови, такі системи можуть адаптуватися до змін у стилістиці та мовних характеристиках новин, що забезпечує їхню ефективність і точність у боротьбі з фейковою інформацією [5].

Фейкові новини можуть мати різні форми, від емоційно забарвлених повідомлень до маніпуляцій фактами, що робить їх складними для автоматичного виявлення. Вони часто спричиняють великі соціальні та політичні наслідки, що обумовлює потребу в створенні ефективних систем для їх автоматичної класифікації. Одним із важливих кроків у розробці таких систем є використання підходів, що включають комбінацію традиційних методів текстової обробки та новітніх технологій машинного навчання. Ці системи дозволяють не лише виділяти найбільш значущі лексичні одиниці, а й аналізувати контекст, у якому вони з'являються, що особливо важливо для виявлення маніпуляцій чи перебільшення в текстах.

Підходи, які застосовуються для класифікації фейкових новин, включають в себе техніки, що дозволяють аналізувати лексичні характеристики та стилістичні ознаки тексту. Наприклад, застосування таких методів, як TruncatedSVD для латентного семантичного аналізу або Latent Dirichlet Allocation (LDA), дозволяє з'ясувати, які теми або терміни найчастіше зустрічаються в текстах, що класифікуються як фейкові. Використання цих методів у поєднанні з

алгоритмами машинного навчання, такими як Random Forest, SVM або глибокі нейронні мережі, дозволяє досягти високої точності в класифікації новин та мінімізувати вплив людського фактору на процес перевірки достовірності інформації [6].

Ще одним важливим аспектом є використання алгоритмів, що дозволяють працювати з великими обсягами даних у реальному часі, зокрема технік онлайн-навчання, які можуть адаптуватися до змін у структурі текстів та стилістиці мовлення. Це дає змогу своєчасно реагувати на нові тенденції у створенні фейкових новин та забезпечує ефективність системи на довгостроковій основі. До таких методів належать різні алгоритми для оновлення моделей в режимі реального часу, що дозволяють зберігати точність класифікації навіть за змінних умов.

Інтеграція таких підходів дозволяє створити гнучкі та адаптивні системи для виявлення фейкових новин, які можна застосовувати в широкому спектрі сфер, включаючи онлайн-медіа, соціальні мережі, а також для автоматизації процесів перевірки фактів. Системи на основі машинного навчання можуть ефективно виявляти новини, які містять ознаки маніпуляції чи дезінформації, що дозволяє значно знизити вплив фейкової інформації на суспільство та допомогти користувачам отримувати достовірну інформацію з перевірених джерел [7].

1.2 Вплив фейкових новин на суспільство та інформаційну безпеку

Фейкові новини мають великий вплив на суспільство, оскільки вони можуть викривляти реальність, маніпулюючи громадською думкою та сприяючи політичним і соціальним кризам. Особливо важливо це в умовах сучасного інформаційного середовища, де багато людей отримують новини через соціальні мережі, які значною мірою не підлягають контролю та перевірці. У таких умовах фейкові новини можуть поширюватися миттєво, набуваючи широкого розголосу. В Україні поширення фейкових новин набуло особливого значення через інформаційну війну, що почалася з анексією Криму та збройним конфліктом на

сході країни. Фальшиві новини використовуються як інструмент маніпуляцій не лише на внутрішньому рівні, але й на міжнародному, що становить загрозу для інформаційної безпеки держави [8].

На політичному рівні фейкові новини активно використовуються для маніпулювання виборчими процесами. В Україні, як і в інших країнах, спостерігається поширення дезінформації під час виборчих кампаній. Це може включати як фальшиві повідомлення про кандидатів, так і вигадані скандали чи маніпулювання електоральними настроями. Наприклад, під час президентських виборів 2019 року було зафіксовано багато фальшивих новин, що стосувалися як кандидатів, так і певних подій, що мали на меті змінити ставлення виборців. Це показує, як фейкові новини можуть бути використані для підриву демократичних процесів, порушення виборчої справедливості і зниження рівня довіри до політичної системи [9].

Фейкові новини також є значною загрозою для інформаційної безпеки, оскільки вони можуть створювати соціальну напругу та сприяти поглибленню конфліктів. Зокрема, під час війни на сході України, фейкові новини активно використовувалися для створення недовіри між різними групами населення та політичними силами, а також для розпалювання ненависті та ворожнечі. Поширення неправдивої інформації про діяльність армії, гуманітарну ситуацію чи політичну ситуацію може призвести до загострення внутрішньополітичної ситуації, погіршення міжетнічних стосунків та загрози національній безпеці. Такі інформаційні атаки стали важливою складовою гібридної війни, яка ведеться не лише на полі бою, а й в інформаційному просторі [10].

На глобальному рівні фейкові новини мають здатність впливати на міжнародні відносини, сприяючи дезінформації, що підриває стабільність держав. Країни, які використовують фальшиві новини як інструмент зовнішньої політики, можуть через них маніпулювати світовими поглядами на конфлікти чи економічні проблеми. Важливим аспектом цієї проблеми є не лише національна, але й міжнародна безпека, оскільки неправдиві новини можуть спричиняти економічні чи політичні санкції, міжнародні кризи та навіть воєнні конфлікти.

Ось чому боротьба з фейковими новинами є важливою частиною не тільки внутрішньої інформаційної безпеки, але й глобальної стабільності [11].

Для ефективної протидії фейковим новинам необхідні комплексні підходи, включаючи розробку технологій автоматичного виявлення дезінформації, підтримку медіаграмотності серед населення та створення законодавчих ініціатив, що регулюють діяльність медіа та інформаційних платформ. В Україні, зокрема, активно розвиваються платформи для перевірки фактів, такі як StopFake, які покликані допомогти громадянам перевіряти достовірність новин та запобігати поширенню фальшивої інформації. Водночас важливою є і боротьба з використанням соціальних мереж для розповсюдження дезінформації. За допомогою автоматичних алгоритмів, які аналізують тексти новин, можна зменшити масштаби поширення фейкових новин та зберегти інформаційну безпеку на національному та міжнародному рівнях [12].

1.3 Дослідження існуючих методів виявлення фейкових новин на основі ключових слів

У контексті боротьби з дезінформацією важливу роль відіграють методи автоматичного виявлення фейкових новин, зокрема підходи, що базуються на аналізі ключових слів. Ключові слова відображають основні змістові аспекти тексту та можуть слугувати ефективними предикторами правдивості повідомлення. У цьому підрозділі розглянуто результати актуальних досліджень, присвячених порівнянню ефективності методів виділення ключових слів (TF-IDF, RAKE, YAKE!, KeyBERT, LSA, LDA, TextRank) для задач класифікації фейкових новин. Аналіз охоплює точність, швидкодію, релевантність витягнутих слів та здатність до обробки коротких текстів. На основі узагальнення емпіричних результатів сформовано порівняльну таблицю, яка дозволяє візуалізувати переваги та недоліки кожного з методів.

Lipianina-Honcharenko et al. провели комплексну оцінку ефективності методів виділення ключових слів для класифікації фейкових новин, таких як TF-

IDF, RAKE, Yake!, KeyBERT, LSA, LDA та TextRank. Найкращі результати було отримано при використанні Yake! та KeyBERT — точність класифікації перевищила 85% [13].

Mishchenko et al. удосконалили алгоритм TF-IDF для використання з наївним байєсівським класифікатором. У дослідженні зазначено, що розширений TF-IDF забезпечив точність 89% при класифікації новин, перевищуючи стандартний підхід на 4% [14].

Sriram порівнював TF-IDF із Word2Vec та Sentence Embeddings у задачі виявлення фейків. Використання TF-IDF із лінійним SVM забезпечило точність 86.3%, що виявилось найвищим серед усіх розглянутих методів [15].

Li & Qu розглядали застосування ключових слів, згенерованих за допомогою TF-IDF, YAKE та RAKE, для виявлення фейкових кампаній на краудфандингу. TF-IDF забезпечив Recall на рівні 78%, а Yake! — найнижчу похибку у 9% [16].

Dubey et al. запропонували фреймворк класифікації фейкових новин, використовуючи TF-IDF як основний засіб векторизації. В експериментах точність моделей з використанням TF-IDF сягала 91.3% при використанні SVM [17].

Farinha зосередився на застосуванні RAKE і YAKE! для аналізу Twitter-повідомлень. RAKE продемонстрував кращу якість витягу ключових слів у коротких текстах, тоді як YAKE! був значно швидшим за інші методи [18].

Landu et al. протестували RAKE і TF-IDF на новинних статтях Сенегалу. RAKE перевершив TF-IDF за точністю (81% проти 75%) та кращою релевантністю витягнутих слів [19].

Nadim et al. провели порівняльну оцінку інструментів RAKE, YAKE, TF-IDF та graph-based моделей. RAKE та YAKE показали найкращу швидкодію (менше 1 сек на документ) з конкурентною точністю — понад 80% [20].

Campos et al. представили оригінальний опис алгоритму YAKE!, який базується на локальних ознаках документа. У дослідженні YAKE! виявився на 60% швидшим за TF-IDF та RAKE при подібній точності (~82%) [21].

Jafari et al. адаптували TF-IDF та YAKE! для рекомендації хештегів у соцмережах. YAKE! показав найвищу релевантність — 0.78 за метрикою NDCG, перевищивши TF-IDF на 12% [22].

Як видно з таблиці 1.1, найвищу точність демонструють моделі на основі TF-IDF (до 91%) та KeyBERT (до 88%), однак останній значно поступається в швидкодії, що обмежує його використання в реальному часі.

Таблиця 1.1 - Порівняльна таблиця методів виявлення фейкових новин на основі ключових слів

Метод	Середня точність (%)	Швидкодія	Переваги	Недоліки
TF-IDF	86–91	Середня	Стабільність, простота реалізації	Менша релевантність у коротких текстах
RAKE	81–83	Висока	Добра якість у коротких текстах	Потребує ручної оптимізації
YAKE!	82–85	Дуже висока	Локальні ознаки, швидке виконання	Нижча точність на великих корпусах
KeyBERT	85–88	Низька	Семантична релевантність	Висока обчислювальна вартість
TextRank	~80	Середня	Графова інтерпретація	Вразливість до параметрів вікна
LSA	~78	Середня	Виявлення латентних тем	Складність налаштування
LDA	~80	Низька	Тематична узагальненість	Чутливість до розміру корпусу

Методи YAKE! та RAKE є найбільш ефективними для коротких текстів, зокрема у соціальних мережах, завдяки своїй швидкодії та незалежності від корпусу. TextRank, LSA та LDA мають переваги у контекстному моделюванні, однак поступаються за точністю і є більш ресурсоємними. Зважаючи на наведений аналіз, вибір методу має базуватись на специфіці задачі: для швидких

рішень у режимі реального часу доцільно використовувати YAKE!, тоді як для точного тематичного аналізу — TF-IDF або KeyBERT з відповідною оптимізацією.

1.4 Постановка задачі дослідження

Актуальність проблеми фейкових новин зростає через їхній негативний вплив на інформаційне суспільство, політичні процеси, економіку та національну безпеку. Фейкові новини часто маніпулюють громадською думкою, підривають довіру до державних інститутів, створюють конфлікти в суспільстві та сприяють поширенню дезінформації. Особливо важливим є виявлення та блокування фейкових новин у реальному часі, оскільки ці новини можуть поширюватися з високою швидкістю через соціальні мережі та інші онлайн платформи. Сучасні інформаційні технології, зокрема аналіз тексту за допомогою алгоритмів машинного навчання, є важливими інструментами для боротьби з цією проблемою. Однак існуючі методи виявлення фейкових новин стикаються з труднощами через динамічні зміни в стилістичних характеристиках новин, відсутність єдиних стандартів для класифікації новин та складність виявлення контекстуальних маніпуляцій.

Існуючі підходи, такі як методи машинного навчання та видобуток ключових слів, забезпечують певну ефективність у розпізнаванні фейкових новин. Алгоритми машинного навчання, зокрема класифікатори, такі як Random Forest і Naive Bayes, використовуються для класифікації новин на основі тренувальних даних. Однак точність цих методів може бути значно знижена через відсутність достатньої кількості маркованих даних для тренування, а також через складність виявлення нових патернів фейкових новин, які змінюються з часом. Крім того, методи видобутку ключових слів, такі як TF-IDF, RAKE та Yake!, добре справляються з виявленням значущих термінів, але вони не завжди враховують контекст новини, що може призводити до помилкових класифікацій.

Урахування всіх цих факторів обумовлює необхідність удосконалення існуючих підходів та розробки нових методів, здатних поєднати переваги декількох підходів для досягнення більш високої точності виявлення фейкових новин. Одна з таких стратегій полягає в інтеграції методів видобутку ключових слів із сучасними методами машинного навчання для більш точного аналізу новин. Впровадження комбінованого підходу, який використовує кілька методів обробки текстів, дозволить не лише покращити точність класифікації новин, а й зробити систему більш гнучкою та адаптивною до нових форм фейкових новин.

Метою роботи є розробка нового комбінованого підходу до виявлення фейкових новин на основі ключових слів.

Завдання дослідження:

1. Аналіз існуючих методів виявлення фейкових новин та оцінка їх переваг і недоліків.
2. Формалізація математичної моделі для виявлення фейкових новин на основі ключових слів.
3. Розробка алгоритму для видобутку ключових слів та їх класифікації.
4. Реалізація програмного модуля для виявлення фейкових новин із застосуванням сучасних методів машинного навчання.
5. Тестування та оцінка точності моделі на основі реальних даних.
6. Оцінка ефективності запропонованого рішення та аналіз можливих шляхів його вдосконалення.

2 ПРОЕКТУВАННЯ МОДУЛЯ ВИЯВЛЕННЯ ФЕЙКОВИХ НОВИН

2.1 Проектування архітектури модуля виявлення фейкових новин

Актуальність розвитку систем для автоматичного виявлення фейкових новин зростає через їхній значний вплив на інформаційне середовище, політичні та соціальні процеси. Тому розробка ефективних методів для автоматичної класифікації новин є важливою задачею. Задля досягнення високої точності і ефективності моделі виявлення фейкових новин, пропонується використання комбінованих підходів, що включають попередню обробку текстів, видобуток ознак, машинне навчання для класифікації та оцінку результатів. Архітектура запропонованої системи виявлення фейкових новин побудована з використанням різноманітних методів обробки тексту та машинного навчання, що дозволяє підвищити точність класифікації.

Компоненти архітектури системи:

- Користувач: Користувач є ініціатором всіх основних процесів системи. Його роль полягає в завантаженні даних та запуску процесу обробки. Основні функції користувача включають:

1) uploadCSV() — завантаження CSV файлів, що містять текстові дані для подальшої обробки;

2) startProcessing() — ініціює обробку даних і подальший аналіз, запускаючи всі наступні етапи обробки.

- DataLoader (Завантажувач даних): Компонент DataLoader відповідає за завантаження текстових даних, які підлягають подальшій обробці. Цей компонент забезпечує необхідну підготовку даних до аналізу:

1) loadData() — завантажує текстові дані із зовнішніх джерел, зокрема з CSV файлів;

2) cleanText() — виконується очищення тексту від небажаних символів, що можуть спотворювати аналіз;

3) tokenize() — перетворює текст на окремі токени (слова чи фрази);

4) removeStopWords() — видаляє стоп-слова, що не несуть суттєвої

інформації для класифікації та аналізу.

- KeywordExtractor (Виділення ключових слів): Компонент KeywordExtractor призначений для виділення важливих слів або фраз із тексту, що дозволяє скоротити розмір оброблюваних даних і зосередитися на найбільш релевантних частинах тексту:

1) extractTFIDF() — метод для виділення найбільш важливих слів на основі їх частоти в документі;

2) extractRAKE() — алгоритм для виділення ключових слів, який використовує співвідношення частоти та важливості слів у тексті;

3) extractTextRank() — метод, що застосовує графовий підхід для аналізу взаємозв'язків слів і виділення найбільш значущих ключових слів.

- Classifier (Класифікатор): Classifier є центральним компонентом для класифікації текстів на основі виділених ознак. Цей компонент відповідає за перетворення тексту в числові вектори та використання їх для навчання моделі:

1) vectorize() — перетворює текстові дані в числову форму для подальшого аналізу;

2) train() — навчає модель на основі векторизованих даних, що містять текстові ознаки;

3) compareResults() — порівнює результати різних моделей для вибору найбільш ефективної.

- Visualizer (Візуалізатор): Компонент Visualizer відповідає за візуалізацію результатів класифікації та аналізу. Це дозволяє оцінити ефективність моделей і порівняти різні методи класифікації:

1) plotAccuracy() — створює графік, що відображає точність роботи кожного методу;

2) highlightBestMethod() — підсвічує метод, який демонструє найкращі результати;

3) showAccuracyTable() — виводить таблицю точності для кожного методу, що використовується в системі.

- ReportGenerator (Генератор звітів): ReportGenerator автоматизує процес

створення підсумкових звітів, надаючи користувачам важливу інформацію про ефективність класифікації та точність моделі:

1) generateMetrics() — створює основні метрики точності, ефективності та інших важливих параметрів;

2) showFinalReport() — виводить остаточний звіт, що містить підсумки роботи системи, точність моделей і рекомендації для оптимізації.

На рисунку 2.1 зображено архітектуру системи виявлення фейкових новин, яка включає ключові компоненти для обробки текстових даних, видобутку ознак, навчання моделі та оцінки результатів.

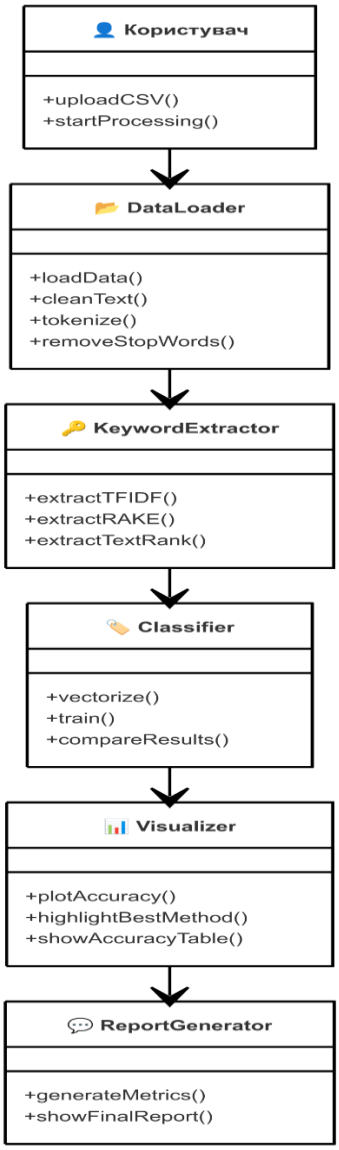


Рисунок 2.1 - Архітектура системи виявлення фейкових новин

Взаємодія між компонентами організована таким чином, що після завантаження даних DataLoader передає їх до TextPreprocessor для попередньої обробки. Після цього дані передаються до Feature Extractor, де за допомогою методів TF-IDF, RAKE, Yake!, LDA та інших здійснюється видобуток ключових слів та ознак.

Отримані ознаки передаються в Classifier, який здійснює класифікацію новин за допомогою моделей машинного навчання. Після цього Evaluator оцінює ефективність класифікації, а Visualizer надає графічне представлення результатів для подальшого аналізу.

Цей підхід дозволяє ефективно комбінувати методи, підвищуючи точність виявлення фейкових новин, а також забезпечує адаптивність системи до нових типів дезінформації, що може з'являтися в інформаційному просторі.

2.2 Методи обробки та підготовки текстових даних

Обробка та підготовка текстових даних є критичним етапом у задачах аналізу тексту, включаючи виявлення фейкових новин. Для ефективного використання методів машинного навчання необхідно провести попередню обробку текстових даних, щоб знизити складність і шум, а також забезпечити зручний формат для подальшого аналізу. Основні етапи цієї обробки включають токенізацію, видалення стоп-слів, лематизацію, а також векторизацію текстів.

Токенізація є процесом розбиття тексту на окремі одиниці (токени), зазвичай слова або фрази. Токенізація дозволяє представити текст у вигляді, зручному для подальшого аналізу. Токенізація може бути реалізована через просте розбиття за пробілами або за більш складними алгоритмами, що враховують знаки пунктуації.

Математично цей процес можна описати як функцію:

$$Tokenize(T) = \{t_1, t_1, \dots, t_n\}\,, \tag{2.1}$$

де T — вхідний текст;

$\{t_1, t_1, \ldots, t_n\}$ — набір токенів, що утворюються в результаті токенізації.

Стоп-слова — це слова, які не несуть значної інформації і часто зустрічаються в текстах, але не впливають на результати аналізу. До таких слів зазвичай відносяться артиклі, сполучники, прийменники тощо. Видалення стоп-слів знижує шум у текстових даних, зберігаючи лише найбільш значущі слова для класифікації чи інших завдань.

Математично процес видалення стоп-слів можна описати як:

$$RemoveStopWords(T) = T' \text{ where } T' = \frac{T}{S}, \tag{2.2}$$

де S — множина стоп-слів;

T' — текст після видалення стоп-слів.

Приведення всіх слів до одного регістру (зазвичай до нижнього) дозволяє уникнути дублювання одних і тих самих слів через різні варіанти написання (наприклад, "Python" і "python"). Це спрощує подальший аналіз і векторизацію тексту.

Математично цей процес можна виразити як:

$$LowerCase(T) = \{lower(t) \mid t \in T\}, \tag{2.3}$$

де *lower(t)* — функція, яка переводить слово *t* до нижнього регістру.

Лематизація — це процес приведення слів до їх основної форми (леми), що дозволяє зменшити кількість варіантів одних і тих самих слів. Наприклад, "running" та "ran" перетворюються в "run". Лематизація дозволяє знизити кількість відмінків і форм слів, що підвищує ефективність аналізу.

Математично лематизацію можна описати як:

$$Lemmatize(t) = lemma(t), \tag{2.4}$$

де *lemma(t)* — це лема слова *t*.

Векторизація тексту — це процес перетворення тексту в числові представлення, які можна використовувати для аналізу з допомогою алгоритмів машинного навчання. Найпоширенішими методами векторизації є TF-IDF (Term Frequency-Inverse Document Frequency) та Bag of Words (BoW).

Метод BoW полягає в представленні тексту як набору слів, де кожне слово розглядається незалежно від контексту, і вектор містить лише інформацію про кількість появи кожного слова в тексті. Формально це можна виразити як:

$$BoW(T) = \{count(t_i) \mid t_i \in T\}, \tag{2.5}$$

де *count(t_i)* — це кількість появи слова t_i у тексті *T.*

Обробка та підготовка текстових даних є важливим етапом в задачах автоматичного виявлення фейкових новин, оскільки дозволяє знизити шум і зберегти лише важливу інформацію для подальшого аналізу. Приведення тексту до єдиного формату, зокрема через токенізацію, лематизацію, видалення стоп-слів та векторизацію, дозволяє значно покращити ефективність подальших етапів аналізу, таких як виявлення ключових слів, класифікація та оцінка результатів. Таким чином, правильна підготовка текстових даних є критично важливою для успішного виконання задач виявлення фейкових новин.

2.3 Методи виводу ключових слів

Видобуток ключових слів з тексту є важливим етапом у задачах текстової класифікації, зокрема при виявленні фейкових новин. Ключові слова представляють основні теми або ідеї в тексті і можуть слугувати важливими ознаками для класифікації. Існує кілька підходів до видобутку ключових слів, зокрема статистичні методи, такі як TF-IDF (Term Frequency-Inverse Document Frequency), RAKE (Rapid Automatic Keyword Extraction), Yake! та методи на

основі графів, наприклад TextRank. Кожен з цих методів має свої переваги та обмеження в залежності від контексту задачі та типу текстових даних.

TF-IDF (Term Frequency-Inverse Document Frequency) [23] — це метод для оцінки важливості терміну в документі стосовно всієї колекції документів або корпусу. Він складається з двох основних частин: TF (частота терміна) та IDF (зворотна частота документа), що разом допомагають виявити, наскільки важливий певний термін у конкретному документі, порівняно з усім набором документів.

Частота терміна показує, як часто певний термін зустрічається в конкретному документі. Формула для розрахунку TF виглядає так:

$$TF(t,d) = \frac{f(t,d)}{\sum_k f(k,d)}, \tag{2.6}$$

де $f(t,d)$ — кількість появ терміна *t* у документі *d;*

$\sum_k f(k,d)$ — загальна кількість всіх термінів у документі *d.*

Таким чином, TF показує частку певного терміна від загальної кількості термінів у документі.

Зворотна частота документа вказує на важливість терміна у всьому корпусі документів. Якщо термін зустрічається у багатьох документах, то його важливість знижується. Формула для IDF виглядає так:

$$IDF(t,D) = \frac{loglog\,(|D|)}{(|d \in D: t \in d|)}, \tag{2.7}$$

де $|\ D\ |$ — загальна кількість документів у корпусі;

$|\ \{d \in D: t \in d\}\ |$ — кількість документів, які містять термін t.

Чим рідше зустрічається термін у документах, тим вищим буде його IDF-значення.

Остаточна формула TF-IDF поєднує обидва ці показники:

$$TF - IDF(t, d, D) = TF(t, d) \cdot IDF(t, D) \quad (2.8)$$

Це означає, що важливість терміна залежить як від його частоти в документі, так і від його загальної поширеності в корпусі документів. Термін, що часто зустрічається в одному документі, але рідко в інших, матиме високе TF-IDF значення. Навпаки, терміни, що зустрічаються у багатьох документах, матимуть низьке TF-IDF значення, навіть якщо вони часто зустрічаються в конкретному документі.

RAKE (Rapid Automatic Keyword Extraction) [24] — це метод для автоматичного вилучення ключових слів з тексту на основі частоти термінів та їх співвідношення з іншими термінами в документі. Він не потребує попереднього навчання на даних або лінгвістичних ресурсів (таких як словники або лексичні бази), що робить його ефективним та швидким для виявлення релевантних ключових слів.

Текст поділяється на фрази, що складаються з одного або декількох слів. Для цього використовуються розділові знаки (.,!?), стоп-слова (загальні слова, які не несуть конкретного смислового навантаження) та інші символи, що не є частиною фраз, які можуть бути ключовими словами. Ці фрази можуть складатися з одного або кількох слів, що стоять поруч.

Для кожного терміна в фразах-кандидатах обчислюються два основні показники:

- Частота терміна f(t): кількість поява терміна у всьому тексті;
- Ступінь терміна d(t): кількість усіх термінів, з якими даний термін знаходиться в одній фразі. Це можна розглядати як суму кількості слів у всіх фразах, що містять цей термін.

Оцінка важливості терміна обчислюється на основі відношення його ступеня до частоти:

$$Score(t) = \frac{f(t)}{d(t)} \quad (2.9)$$

Це показує відношення між кількістю термінів, що з'являються разом з даним терміном, і кількістю його появ у тексті. Чим більше ступінь терміна відносно його частоти, тим важливіше це слово для формування ключової фрази.

Для кожної фрази-кандидата обчислюється її загальна оцінка як сума оцінок всіх термінів, що входять у фразу:

$$Score(Phrase) = \sum_{t \in Phrase} Score(t) \quad (2.10)$$

Таким чином, фрази, що містять важливі терміни, отримують високі оцінки і розглядаються як потенційні ключові фрази для тексту.

YAKE! (Yet Another Keyword Extractor) [25]— це метод для автоматичного вилучення ключових слів з тексту, який базується на статистичних характеристиках термінів у тексті без необхідності навчання на зовнішніх корпусах або використання лінгвістичних ресурсів. Основна ідея YAKE! полягає в тому, що він використовує кілька показників для оцінки важливості кожного терміна в тексті з урахуванням локального контексту, позиції та поширеності терміна.

Спочатку текст розбивається на терміни та фрази-кандидати за допомогою розділових знаків, стоп-слів, та інших символів, що не належать до ключових фраз.

Розраховується частота терміна у тексті, тобто кількість разів, коли термін з'являється в тексті. Чим частіше термін з'являється в документі, тим вища його частотна оцінка:

$$TF(t) = f(t, d)\ , \quad (2.11)$$

де $f(t, d)$— кількість появ терміна *t* у документі d.

YAKE! враховує позицію, на якій з’являється термін у тексті. Термін, що з'являється ближче до початку тексту, може мати іншу вагу, ніж той, що з'являється ближче до кінця.

YAKE! враховує, чи з'являється термін у різних контекстах (фразах). Якщо термін зустрічається лише в обмеженій кількості контекстів, його важливість зменшується.

Оцінюється співвідношення терміна з іншими термінами у фразах-кандидатах. Якщо термін часто з’являється з одними й тими ж іншими термінами, його унікальність зменшується.

Підсумкова формула для оцінки ключової фрази включає зважене врахування частоти терміна, позиції, поширеності та унікальності:

$$Score(t) = W1 \cdot f(t,d) + W2 \cdot Pos(t,d) + W3 \cdot Spread(t)\ , \qquad (2.12)$$

де $f(t,d)$ — частота терміна *t;*

$Pos(t,d)$— позиція терміна *t* в документі;

$Spread(t)$ — кількість контекстів, у яких термін з'являється;

$W1, W2, W3$ — вагові коефіцієнти для налаштування впливу кожного компонента.

LSA (Latent Semantic Analysis) [26] — це метод обробки тексту, який використовується для виявлення прихованих семантичних відношень між термінами в текстових документах. Основною метою LSA є зменшення розмірності простору документів і слів, щоб виявити взаємозв'язки між термінами, що не є очевидними на поверхневому рівні.

Текстовий корпус спочатку перетворюється у матрицю термінів-документів. Кожен рядок матриці відповідає певному терміну, а кожен стовпчик — документу. Значення в комірках можуть бути кількістю появ термінів у документах або зваженими значеннями, наприклад, TF-IDF. Таким чином, початкове представлення тексту є високорозмірним і розрідженим.

Формально, нехай A — матриця термінів-документів розміром m×n, де mmm — кількість термінів, а n — кількість документів.

LSA застосовує сингулярний розклад матриці (SVD), щоб зменшити розмірність початкової матриці. SVD розкладає матрицю A на три матриці:

$$A = U\Sigma V^T \tag{2.13}$$

де U — ортогональна матриця термінів;

Σ — діагональна матриця сингулярних значень, що містить ранжовані сингулярні значення (від найбільшого до найменшого);

V^T — ортогональна матриця документів.

Сингулярні значення в матриці Σ визначають важливість кожного з векторів у U і V^T. Менші сингулярні значення можна відкинути, зменшуючи розмірність простору, залишивши лише найзначніші компоненти.

Після виконання SVD, обираються лише найбільші сингулярні значення та відповідні їм компоненти в матрицях U і V^T. Це дозволяє зменшити розмірність матриці, залишивши тільки основні латентні семантичні структури тексту.

Таким чином, матриця зменшеної розмірності дає можливість аналізувати документи і терміни в новому латентному просторі, де терміни, що мають схожі контексти, будуть близькими один до одного.

У зменшеному просторі латентних змінних терміни, які часто зустрічаються в подібних контекстах, але можуть не бути очевидними на поверхневому рівні, стають ближчими один до одного. Це допомагає виявити приховані семантичні відношення між термінами та документами, що дозволяє LSA ефективно працювати в задачах тематичного моделювання, пошуку інформації та класифікації текстів.

LDA (Latent Dirichlet Allocation) [27] — це статистичний метод тематичного моделювання, який використовується для автоматичного вилучення прихованих тем із великої колекції текстових документів. Основна ідея LDA полягає в тому, що кожен документ розглядається як суміш декількох тем, а кожна

тема — як розподіл термінів. Метою алгоритму є виявлення цих тем і розподіл термінів серед них.

Теми в LDA визначаються як розподіли термінів. Це набори слів, які мають високу ймовірність виникнення в межах певної теми. Наприклад, якщо тема стосується політики, то найбільш імовірними словами в ній можуть бути «вибори», «президент», «парламент» тощо.

Кожен документ у корпусі розглядається як суміш кількох тем. Наприклад, документ про економічну політику може бути сумішшю двох тем: економіки та політики.

LDA припускає, що розподіл тем у кожному документі має заздалегідь визначений розподіл, наприклад, Dirichlet-розподіл. Це параметризований розподіл, що дозволяє контролювати, наскільки сильно одна тема домінує в документі. Параметр α\alphaα визначає цей розподіл тем для документів.

$$\theta_d \sim Dirichlet(\alpha)\ , \tag{2.14}$$

де θ_d — вектор розподілу тем для документа d;

α — гіперпараметр, що контролює густоту тем у документах.

Кожна тема також має власний розподіл термінів, що теж моделюється за допомогою Dirichlet-розподілу. Параметр β визначає цей розподіл термінів для кожної теми.

$$\phi_k \sim Dirichlet(\beta)\ , \tag{2.15}$$

де ϕ_k — вектор розподілу термінів для теми k;

β — гіперпараметр, що контролює густоту термінів у темах.

Для кожного терміна в документі спочатку обирається тема на основі розподілу тем для цього документа. Потім термін обирається на основі розподілу термінів для обраної теми.

$$z_{d,n} \sim Multinomial(\theta_d) \quad (2.16)$$

$$w_{d,n} \sim Multinomial(\phi_{z_{d,n}}) \quad (2.17)$$

де $z_{d,n}$ — тема для n-го терміна в документі d;

$w_{d,n}$ — сам термін, обраний відповідно до теми $z_{d,n}$;

θ_d — розподіл тем для документа d;

$\phi_{z_{d,n}}$ — розподіл термінів для обраної теми $z_{d,n}$.

Основне завдання LDA — знайти розподіл тем для кожного документа і розподіл термінів для кожної теми. Це здійснюється за допомогою методів оцінки параметрів, таких як Метод очікування-максимізації (EM) або Метод варіаційного висновку.

Модель LDA описується наступною імовірнісною функцією:

$$P(\alpha, \beta) = P(\alpha) \prod_{k=1}^{K} P(\beta) \prod_{n=1}^{N} P(\theta) P(z_n, \phi) \,, \quad (2.18)$$

де w — терміни в документах;

z — теми для кожного терміна;

θ— розподіл тем для кожного документа;

ϕ— розподіл термінів для кожної теми;

α, β — гіперпараметри Dirichlet-розподілів.

TextRank [28] — це алгоритм для вилучення ключових слів і автоматичної побудови анотацій тексту, заснований на методах ранжування графів. TextRank є адаптацією алгоритму PageRank, який використовується в пошукових системах для ранжування веб-сторінок. У випадку TextRank, замість веб-сторінок, алгоритм працює з словами або реченнями тексту.

Текст представлений у вигляді графа, де вершини — це слова або речення, а ребра встановлюються між тими вершинами, які є "близькими" в контексті. Для задач вилучення ключових слів вершинами є окремі слова, для задач сумаризації — цілі речення.

Для вилучення ключових слів:

- Термін вважається вершиною графа;
- Ребро з'єднує два терміни, якщо вони з'являються в межах одного вікна слів (зазвичай вікно має фіксований розмір, наприклад 2 або 3);
- Для кожної вершини v_i, вагу зв'язків з іншими вершинами можна описати через матрицю подібності, яка враховує частоту зустрічі слів разом у вікні.

Для задач сумаризації:

- Вершинами є цілі речення;
- Ребра з'єднують речення на основі семантичної схожості, яка може бути обчислена, наприклад, за допомогою косинусної подібності між векторними представленнями речень.

Косинусна подібність між реченнями S_i і S_j визначається так:

$$cosine_{similarity(S_i,S_j)} = \frac{S_i \cdot S_j}{\|S_i\|\|S_j\|} \tag{2.19}$$

Речення, що мають високий ступінь подібності, отримують сильніші зв'язки в графі.

Алгоритм TextRank ітеративно обчислює вагу кожної вершини (слова або речення) за наступною формулою, подібною до формули PageRank:

$$R(v_i) = (1-d) + d\sum_{v_j \in In(v_i)} \frac{R(v_j)}{(|Out(v_j)|)}, \tag{2.20}$$

де $R(v_i)$— ранг вершини v_i;

d — коефіцієнт загасання (звичайно d=0.85);

$In(v_i)$ — сукупність вершин, які мають зв'язки з v_i;

$Out(v_j)$ — кількість вихідних ребер з вершини v_j.

Ранг кожної вершини розраховується ітеративно до досягнення збіжності.

Ключові слова: Після того як вершини (слова) отримають ранжування, обираються слова з найвищим рангом як ключові.

Сумаризація тексту: Речення з найвищими рангами обираються для побудови анотації тексту.

2.4 Ансамбль класифікаторів виявлення фейкових новин

Ансамблеві методи машинного навчання засновані на ідеї, що поєднання кількох моделей (кожна з яких може бути досить різною за підходами) часто дає точніший прогноз, ніж використання окремої найкращої моделі. У цій реалізації використано два основних ансамблевих підходи: VotingClassifier (голосування) та StackingClassifier (стекінг). Обидві методики дозволяють поєднати кілька сильних моделей — наприклад, LogisticRegression, RandomForestClassifier, Support Vector Machine (SVC) тощо — в одну метамодель. Завдяки цьому отримуємо більш стійкий та точний результат.

На початку відсортовуємо результати (попередньо обчислені й збережені в змінній results) за точністю у спадаючому порядку. Беремо п’ять найкращих моделей і зберігаємо їх у списку top_methods. Концентруємося лише на тих векторах ознак (TF-IDF, RAKE, Yake!, LSA, LDA, TextRank), які показали найвищі показники точності. Пізніше ці методи стануть базою для ансамблевих модельних композицій.

$$\text{TopMethods} = \{m_1, m_2, \ldots, m_5\} \quad \text{де} \quad \max(\text{Accuracy}(m_i)) \tag{2.21}$$

Для кожного з можливих методів перетворення тексту (TF-IDF, RAKE, Yake!, LSA, LDA, TextRank) заздалегідь створюємо матрицю ознак. Кожен рядок відповідає документу (тексту), а кожен стовпець — певному атрибуту (наприклад, слову або латентному виміру теми). У випадку TF-IDF матриця будується на основі частоти слів, зваженої на обернену частоту документів. Для

RAKE, Yake! та TextRank використовуються ключові слова як ознаки (через CountVectorizer). LSA (Latent Semantic Analysis) та LDA (Latent Dirichlet Allocation) формують зміщений простір ознак зі зменшеною кількістю вимірів або тематичних розподілів:

$$X_{\text{TF-IDF}} \in \mathbb{R}^{N \times d_{\text{TF-IDF}}}, \quad X_{\text{RAKE}} \in \mathbb{R}^{N \times d_{\text{RAKE}}}, \tag{2.22}$$

де N — кількість документів;

d — розмірність простору ознак після відповідного методу.

Отримавши топ-5 методів, для кожного з них витягуємо відповідну матрицю ознак (наприклад, $X_{\text{TF-IDF}}, X_{\text{RAKE}}, \cdots$) і створюємо об'єкти класифікаторів. У прикладі використовується RandomForestClassifier з кількістю дерев n_estimators=100. Кожна пара (метод, модель) додається до списку classifiers, а матриці ознак — до списку X_ensemble. Це означає, що кожен метод матиме власні ознаки та власну випадковий ліс як модель.

Оскільки VotingClassifier вимагає один спільний набір ознак, у прикладі обрано найкращий метод із top_methods[0] (тобто метод, який дав найвищу точність). Відповідно, взято матрицю ознак саме цього методу як X_{final}. Цільова змінна y_{final} — це мітки класу з колонки data['Label'].

Математично, нехай $X_{\text{final}} \in \mathbb{R}^{N \times d_{\text{final}}}$ є найкращою матрицею ознак, а $y_{\text{final}} \in \{0, 1, \ldots, K-1\}^N$ — вектор міток (припустимо, задача класифікації на K класів).

За допомогою функції train_test_split з бібліотеки scikit-learn ділимо X_{final} та y_{final} на тренувальну $(X_{\text{train}}, y_{\text{train}})$ і тестову $(X_{\text{test}}, y_{\text{test}})$ вибірки у пропорції *80%:20%*. Це важливо, щоб мати об'єктивну оцінку продуктивності моделі на «нових» (не бачених під час навчання) даних.

Основна ідея VotingClassifier полягає в тому, що кожна модель-учасник ансамблю робить власний прогноз $\hat{y}_i$, і потім використовується правило

голосування для визначення фінального результату. У режимі voting='hard' фінальний прогноз виходить через мажоритарне голосування:

$$\hat{y}_{\text{final}} = \text{mode}(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_M), \tag{2.23}$$

де M — загальна кількість моделей у голосуванні;

mode повертає найчастіший клас серед усіх прогнозів.

Створюється voting_clf = VotingClassifier(estimators=classifiers, voting='hard') і викликається метод fit(X_train, y_train), який навчає кожну з наших п’яти моделей на одному і тому ж наборі ознак (X_{train}). Потім викликається voting_clf.predict(X_test), щоб отримати фінальні прогнози $\hat{y}_{\text{ensemble}}$. Кожна модель робить свій внесок у голосування, і результат узгоджується за більшістю.

Для вимірювання точності обчислюємо:

$$\text{Accuracy} = \frac{1}{N_{\text{test}}} \sum_{j=1}^{N_{\text{test}}} \mathbb{I}\Big(\hat{y}_{\text{ensemble}}(x_j) = y_{\text{test},j}\Big), \tag{2.24}$$

де N_{test} — кількість зразків у тестовому наборі;

$\hat{y}_{\text{ensemble}}(x_j)$ — прогноз ансамблю для j-го зразка;

$y_{\text{test},j}$ — істинна мітка.

Результат виводиться через accuracy_score(y_test, y_pred_ensemble). Також виконується classification_report для більш детальної статистики (точність, повнота, F1-міра тощо) за кожним класом.

Після отримання точності ансамблю голосування (Accuracy(Ensemble Top 5)) записується це значення в словник results['Ensemble Top 5'], щоб надалі мати змогу порівняти його з іншими моделями та з іншими типами ансамблів.

Стекінг (Stacking) — це більш складний ансамблевий підхід, де прогнози базових моделей стають вхідними ознаками для метамоделі (final_estimator). Тобто, спочатку кожна з базових моделей $(m_1, m_2, \ldots, m_M)$ навчається на

власних особливих ознаках або на спільному наборі ознак, а потім метамодель (наприклад, логістична регресія) навчається на передбаченнях цих базових моделей.

Використовуємо LabelEncoder, щоб перетворити текстові мітки data['Label'] на цілочисельні (0, 1, 2, ...). Така перекодування корисна при побудові більшості моделей, особливо коли є ймовірність, що алгоритми потребують числове уявлення класів. Вектор *y=LabelEncoder(data['Label']).*

Для стекінгу знову звертаюсь до top_methods і для кожного методу створюю відповідний набір ознак *X* та модель RandomForestClassifier. Ці пари записуються в список base_estimators у форматі (method_name, estimator_object). Окрім цього, всі матриці ознак зберігаються у змінну vectorizers_stacking. Ідея полягає в тому, що кожна модель працюватиме з «своїми» ознаками під час крос-валідації.

Тут, у коді, як загальний набір ознак для метамоделі, обрано TF-IDF. Інакше кажучи, $\mathbf{X}_{\text{meta}} = X_{\text{TF-IDF}}$. Саме на основі цих ознак розбито дані на тренувальну та тестову частини для мета-рівня. Проте слід розуміти, що базові моделі (для стекінгу) зазвичай генерують проміжні прогнози (логіти, імовірності чи класи) через крос-валідацію, і вже вони подаються як ознаки до метамоделі.

Знову використовую train_test_split із тими ж параметрами, щоб X_{meta}, y розбити у співвідношенні 80% на тренування та 20% на тест. У такий спосіб зберігається погодженість методів оцінки якості.

Під час тренування стекінгу відбувається така послідовність:

1. Для базової моделі m_k (де $k = 1, \ldots, M$) виконується крос-валідація. Кожне «поділення» дає передбачення $\hat{y}_k^{(train)}$ на валідаційному фолді;

2. Потім усі ці передбачення об’єднуються у вектор (або матрицю, якщо кілька класів) для метамоделі;

3. Залежно від параметра passthrough, до метамоделі також можуть додаватися оригінальні ознаки X_{meta}. У прикладі passthrough=False, отже, метамодель отримує лише передбачення базових моделей.

Формально, метамодель розв’язує задачу:

$$\hat{y} = g(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_M), \tag{2.25}$$

де g — функція, яку вчимо на основі логістичної регресії або іншого фінального естиматора.

Отже, далі представлено запропонований алгоритм у вигляді псевдокоду (рисунок 2.2).

```
Алгоритм: Ансамблеві методи класифікації фейкових новин

Вхід:
    - data: набір текстових новин із мітками 'Label'
    - results: словник із точністю попередніх моделей
    - vectorizers: словник із попередньо створеними матрицями ознак для методів (TF-IDF, RAKE, ...)

Вихід:
    - accuracy_ensemble: точність VotingClassifier
    - accuracy_stack: точність StackingClassifier
    - Оновлений словник results із новими точностями ансамблів

1. Відбір і підготовка моделей:
    - Визначити топ-5 методів векторизації за точністю з results
    - Для кожного методу: отримати матрицю ознак, створити RandomForestClassifier

2. Побудова VotingClassifier:
    - Вибрати ознаки найкращого методу для X_final, взяти y_final із data['Label']
    - Розбити X_final та y_final на тренувальний і тестовий набори
    - Створити та навчити VotingClassifier на X_train
    - Оцінити точність на X_test, зберегти у results['Ensemble Top 5']

3. Підготовка для стекінгу:
    - Закодувати мітки у через LabelEncoder
    - Створити список (method, RandomForestClassifier) для кожного з топ-5 методів

4. Побудова StackingClassifier:
    - Взяти X_meta на основі TF-IDF, розбити на X_train та X_test
    - Ініціалізувати StackingClassifier із LogisticRegression як фінальним класифікатором
    - Навчити стекінг, зробити прогноз, оцінити точність

5. Збереження результатів:
    - Додати accuracy_stack до results['StackingClassifier']
    - Повернути: accuracy_ensemble, accuracy_stack, results
```

Рисунок 2.2 - Псевдокод алгоритму

У цьому алгоритмі описано два підходи ансамблевого навчання для класифікації фейкових новин: голосування (VotingClassifier) та стекінг (StackingClassifier). Спочатку з-поміж попередньо протестованих методів векторизації тексту (TF-IDF, RAKE, YAKE, LSA, LDA, TextRank) обираються п’ять найефективніших за точністю. Для кожного з них створюється відповідний набір ознак і класифікатор RandomForestClassifier. Потім будується VotingClassifier, який працює на основі голосування цих моделей, навчається на тренувальному наборі та оцінюється на тестовому, після чого обчислюється точність ансамблю.

У другій частині алгоритму реалізовано стекінг, у якому кожна базова модель (із топ-5) навчається окремо, а її прогнози передаються метамоделі — логістичній регресії. Мітки класів кодуються за допомогою LabelEncoder, а для мета-рівня використовується спільний набір ознак (TF-IDF). Після навчання стекінг-моделі виконується прогнозування та оцінка точності. У фіналі обидві точності (Voting та Stacking) записуються у словник результатів, що дозволяє порівняти ефективність підходів.

Підсумовуючи, реалізація ансамблевих методів Voting та Stacking у задачі виявлення фейкових новин дозволила ефективно поєднати результати декількох моделей на основі різних векторизацій тексту. VotingClassifier забезпечив просту, але надійну агрегацію результатів шляхом голосування, тоді як StackingClassifier продемонстрував гнучкість і здатність адаптуватися до особливостей базових моделей, використовуючи метамодель для остаточного рішення. Такий підхід суттєво підвищує точність класифікації та дозволяє зменшити ризик переобучення, що робить його доцільним для практичного застосування в системах автоматичного виявлення недостовірної інформації.

3 РЕАЛІЗАЦІЯ МОДУЛЯ ВИЯВЛЕННЯ ФЕЙКОВИХ НОВИН

3.1 Опис середовища програмування

Програмне середовище, яке було використано для реалізації даного проекту, надає користувачеві всі необхідні інструменти для обробки великих текстових даних, зокрема для задач виявлення фейкових новин. Всі операції обробки, аналізу та тестування моделей були виконані в середовищі Google Colab. Це хмарне середовище дозволяє ефективно працювати з великими обсягами даних та використовувати потужні обчислювальні ресурси, такі як графічні процесори (GPU) та тензорні процесори (TPU). Окрім цього, Google Colab інтегрується з Google Drive, що дає можливість зберігати та отримувати доступ до даних без необхідності використання локального сховища.

Для реалізації задачі виявлення фейкових новин були використані різноманітні бібліотеки Python, які дозволяють виконувати всі необхідні етапи обробки та аналізу текстових даних. Зокрема, були застосовані такі бібліотеки:

- NLTK (Natural Language Toolkit) — це одна з найпопулярніших бібліотек для обробки природної мови. Вона надає набір інструментів для токенізації, стемінгу, лематизації, а також для проведення частинно-мовного аналізу. Для даної задачі використовувались функції для токенізації тексту, що дозволяють розділити великий текст на окремі слова або речення, що є важливим для подальших етапів обробки.

- Scikit-learn — бібліотека, що містить набір алгоритмів машинного навчання, а також інструменти для попередньої обробки даних (векторизація тексту за допомогою TF-IDF або Bag of Words), побудови моделей класифікації, а також оцінки їх ефективності. У даному проекті Scikit-learn використовувалася для створення та тестування моделей класифікації.

- Pandas — бібліотека для маніпулювання даними, що дозволяє ефективно працювати з даними у вигляді таблиць (DataFrame). Вона була використана для зчитування текстових даних з CSV файлів, очищення їх від непотрібних

символів, а також для організації даних перед їх векторизацією та подальшою обробкою.

- Matplotlib та Seaborn — бібліотеки для візуалізації даних, що дозволяють створювати графіки, гістограми, діаграми та інші візуальні елементи, які використовуються для порівняння результатів тестування моделей та оцінки ефективності класифікації фейкових новин.

Для забезпечення коректної роботи з бібліотеками та їх ресурсами було здійснено кілька важливих кроків налаштування середовища програмування.

Оскільки бібліотека NLTK використовує певні ресурси для токенізації та інших операцій, важливо забезпечити їх наявність у системі. Для цього в середовищі виконання було передбачено перевірку наявності каталогу для збереження цих ресурсів та їх створення у разі відсутності. Додатково було налаштовано шлях до директорії, де зберігаються ці ресурси, що гарантує їх коректну роботу протягом виконання програми.

Цей етап, що включає перевірку наявності та створення каталогу для ресурсів, полягає в тому, що для забезпечення безперебійної роботи бібліотеки NLTK створюється каталог, якщо він відсутній, і додається шлях до цього каталогу для забезпечення доступу до всіх необхідних ресурсів.

Цей етап гарантує, що всі необхідні ресурси будуть збережені в правильно налаштованому каталозі, і система не зіткнеться з помилками через відсутність цього каталогу в процесі виконання.

Для того щоб бібліотека NLTK змогла коректно працювати з усіма своїми ресурсами, необхідно вказати правильний шлях до директорії, де ці ресурси зберігаються. Тому в системному шляху nltk.data.path додається шлях до цього каталогу.

Цей етап також згадано в попередньому розділі 2.4, де показано, як додати шлях до збережених ресурсів NLTK, щоб забезпечити їх доступність.

Це налаштування дозволяє бібліотеці NLTK знаходити необхідні файли на всіх етапах виконання без виникнення помилок, пов'язаних із відсутністю доступу до ресурсів.

Одним із основних етапів обробки тексту є токенізація, для якої бібліотека NLTK використовує ресурс punkt. Для того щоб цей процес виконувався коректно, необхідно завантажити відповідний ресурс punkt, який відповідає за розбиття тексту на окремі слова або речення.Щоб забезпечити коректну роботу токенізації, достатньо завантажити ресурс

Завантаження ресурсу punkt є необхідним для правильного виконання токенізації тексту, що є важливим етапом при подальшому аналізі та обробці даних.

Для забезпечення коректної обробки українських текстів було використано власний файл зі списком стоп-слів (рисунок 3.1):

```
with open('/content/drive/MyDrive/source/stopwords_ua.txt', 'r', encoding='utf-8') as f:
    ukrainian_stop_words = set(f.read().splitlines())
```

Рисунок 3.1 – Зчитування українських стоп-слів із файлу

Цей список використовується для фільтрації слів, що не несуть значної інформації і можуть негативно впливати на ефективність аналізу. Завантаження та використання такого списку дає змогу покращити точність моделей.

Були розглянуті ключові етапи налаштування середовища для роботи з бібліотекою NLTK, що є основою для подальшої обробки текстових даних. Зокрема, було детально описано процес перевірки наявності та створення каталогу для збереження ресурсів NLTK, налаштування шляху до цих ресурсів та завантаження необхідного ресурсу punkt для токенізації текстів.

Ці етапи є критично важливими для забезпечення коректної роботи з текстовими даними. Наявність правильно налаштованого середовища гарантує безперебійну роботу бібліотеки NLTK, що дозволяє здійснювати ефективну обробку текстів на різних етапах аналізу, таких як токенізація, векторизація, класифікація та екстракція ключових слів. Забезпечення доступу до всіх необхідних ресурсів та налаштування середовища дозволяє уникнути помилок,

пов'язаних з відсутністю ресурсів чи неправильними шляхами до них, що значно підвищує ефективність обробки даних.

Таким чином, налаштування середовища для роботи з NLTK є важливим кроком для подальшого успішного виконання задач обробки текстів, зокрема для виявлення фейкових новин, що підвищує точність і надійність моделей машинного навчання.

3.2 Підготовка та обробка текстових даних

Підготовка та обробка даних є основними етапами в процесі побудови ефективних моделей машинного навчання, зокрема для задачі виявлення фейкових новин. Ці етапи дозволяють підготувати дані для подальшого використання в моделях, які здійснюють класифікацію текстів. У даному розділі описано, як дані обробляються та підготовлюються для тестування на основі наданого коду, включаючи етапи завантаження, очищення, токенізації, векторизації, а також поділу на навчальну та тестову вибірки.

Процес обробки даних починається з завантаження текстових даних у програму. Це необхідний етап, оскільки текстові дані зазвичай зберігаються у файлах різних форматів, таких як CSV, JSON тощо. У даному випадку для завантаження даних використовується бібліотека Pandas**,** що дозволяє ефективно працювати з таблицями даних у форматі DataFrame**.**

У рамках даного дослідження було проведено розвідковий аналіз даних (Exploratory Data Analysis, EDA) текстового набору, що містить повідомлення українською мовою з відповідними мітками фейковості. Метою аналізу є виявлення структурних характеристик, особливостей розподілу текстів, семантичних закономірностей та підготовка основи для подальшої побудови моделей машинного навчання.

Набір даних було завантажено у форматі CSV, де присутні два основних стовпці: Text — текст повідомлення, та Label — мітка, що вказує на його істинність (значення “Fake” або “True”). Усього в датасеті — 73 721 запис, проте

лише 8 043 з них мають вказану мітку класу, а 6 записів містять відсутні значення в полі тексту.

Для забезпечення коректності аналізу було проведено очищення даних: видалено рядки з пропущеними значеннями та дублікатами. У результаті залишилося 7 804 унікальні повністю заповнені записи. Для кожного запису було обчислено додаткову метрику — довжину тексту в символах.

На рисунку 3.2 зображено кількість текстів у кожному класі. Спостерігається незначна перевага записів, що мають мітку "True" (4 668), у порівнянні з "Fake" (3 375). Це свідчить про помірну незбалансованість класів, яка може бути врахована під час моделювання.

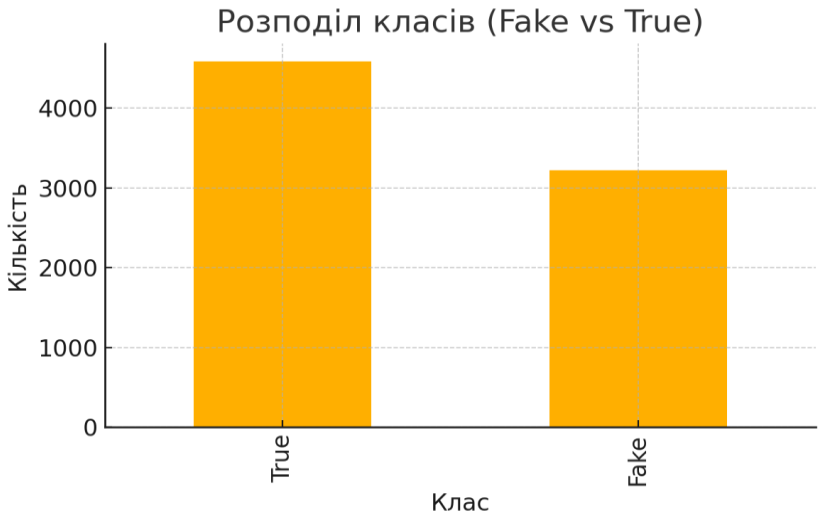


Рисунок 3.2 - Розподіл кількості записів у класах "True" та "Fake".

На рисунку 3.3 представлено гістограму довжини текстів. Вона демонструє, що більшість повідомлень мають довжину від 500 до 1500 символів, що вказує на наративний, а не короткий формат (наприклад, заголовків).

Рисунок 3.4 демонструє середню довжину текстів залежно від мітки. Повідомлення з класом “True” в середньому довші, ніж “Fake”, що може свідчити про більшу інформативність або ширше викриття теми у достовірних джерелах.

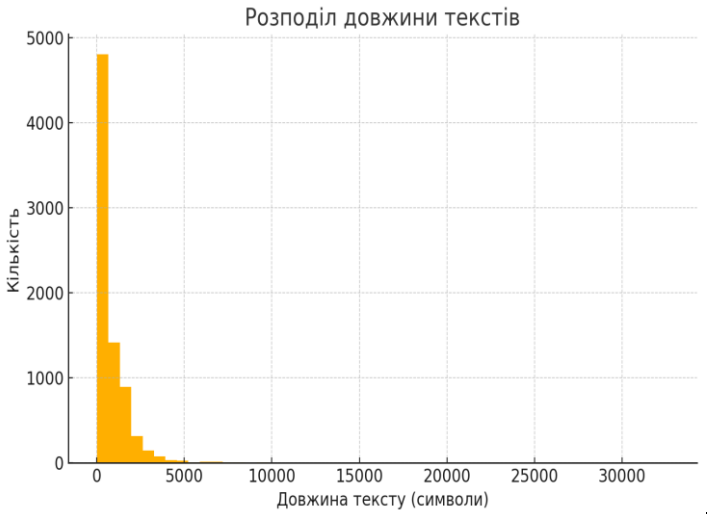


Рисунок 3.3 - Гістограма довжини текстів у символах.

Рисунок 3.4 - Середня довжина тексту залежно від класу.

Для кожного з класів було проведено підрахунок найчастіших слів (уніграми). Враховуючи, що в українській мові переважають функціональні слова (прийменники, сполучники тощо), до топ-20 найпоширеніших слів увійшли: “на”, “у”, “в”, “що”, “з”.

На рисунку 3.5 та рисунку 3.6 зображено графіки частоти найпоширеніших слів у кожному класі. Видно, що набір функціональних слів є подібним, однак частотність деяких відмінна між класами. Це відкриває можливість для побудови дискримінативних ознак при векторизації текстів.

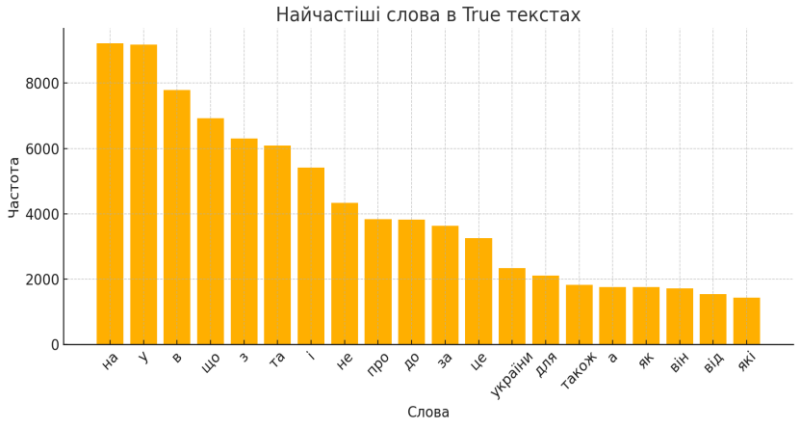


Рисунок 3.5 - Частотність найпопулярніших слів у "True" текстах

Рисунок 3.6 - Частотність найпопулярніших слів у "Fake" текстах

Для кращого уявлення лексичного поля кожного з класів було згенеровано WordCloud для “True” та “Fake” повідомлень (рисунки 3.7 та 3.8). У візуалізаціях видно повторення загальних термінів, однак можна помітити акценти на певних емоційних або політичних словах у фейкових текстах.

Рисунок 3.7 - WordCloud для "True" новин

Рисунок 3.8 - WordCloud для "Fake" новин

Аналіз показав, що фейкові тексти відрізняються від достовірних як за довжиною, так і за лексичними характеристиками. Спостерігається певна текстова економія у “Fake” повідомленнях. Частотний та візуальний аналіз лексем дав змогу виявити потенційно інформативні ознаки для подальшої класифікації. Попри домінування стоп-слів, відмінності в контекстах є помітними.

Після проведення розвідкового аналізу було здійснено повну попередню обробку текстів, яка включала нормалізацію, очищення та лемматизацію. Зокрема, всі тексти було приведено до нижнього регістру, очищено від пунктуаційних знаків, а також видалено українські стоп-слова — функціональні слова, які не несуть значного смислового навантаження (в, на, до, і тощо). Завдяки цьому значно зменшився словниковий запас (вектор ознак), що дозволило уникнути перенасичення моделі непотрібною інформацією. Останнім етапом було застосування лемматизації — перетворення слів до їх базової (словникової) форми, що сприяло зведенню схожих морфологічних форм до єдиного представлення (наприклад: війна, війною, війни → війна).

З метою кращого врахування контекстної інформації було побудовано n-грамні моделі: уніграми (окремі слова), біграми (пари слів) та триграми (три слова поспіль). На прикладі 1000 випадкових текстів було виявлено найчастіші n-грамні конструкції (див. рисунки 3.9). Наприклад, серед уніграм домінують загальні терміни, тоді як біграми та триграми виявляють більш інформативні фрази, характерні для журналістських текстів або фейкових повідомлень. Це дозволяє використовувати такі послідовності як ефективні ознаки для майбутнього класифікаційного моделювання.

Рисунки 3.9 демонструють топ-10 уніграм, біграм та триграм, які найчастіше зустрічаються у вибірці. Як видно, n-грамна модель значно збагачує репрезентацію текстів та дозволяє виявляти шаблони, притаманні фейковим повідомленням.

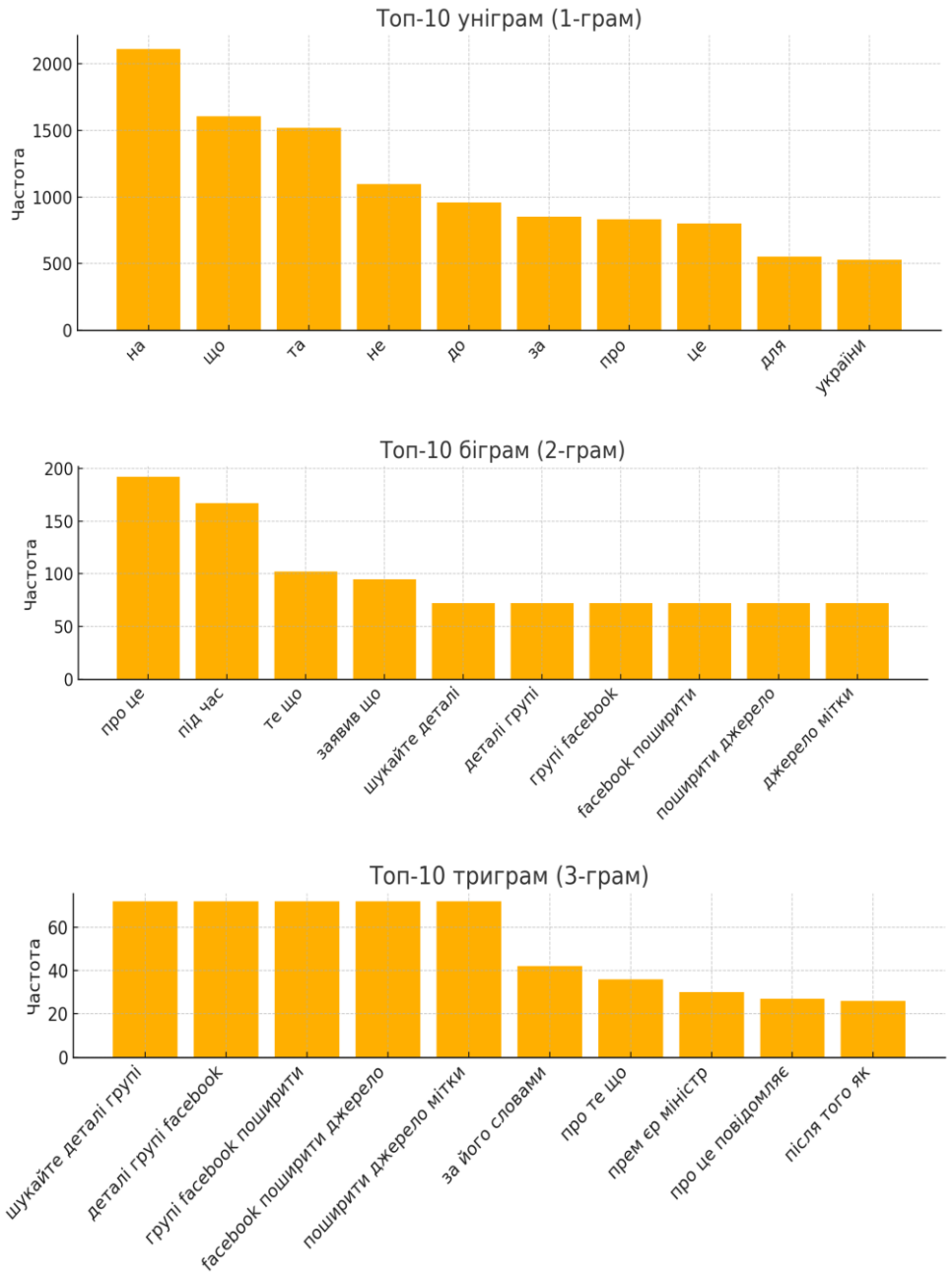


Рисунок 3.9 - Топ-10 уніграм, біграм та триграм

Отже, в результаті повного EDA та попередньої обробки текстів було виявлено ключові лексичні та структурні особливості фейкових і достовірних новин, а також сформовано інформативні n-грамні ознаки. Це створює надійну основу для побудови моделей машинного навчання з метою автоматичної класифікації фейкових повідомлень.

3.3 Оцінка ефективності виявлення фейкових новин

Тестування та оцінка ефективності є невід'ємною частиною процесу розробки будь-якої моделі машинного навчання, зокрема для задач класифікації текстових даних, таких як виявлення фейкових новин. Оцінка ефективності моделі дозволяє не тільки перевірити точність її роботи, але й визначити ступінь її здатності до узагальнення на нові дані, що є критично важливим для реальних застосунків.

Аналіз класифікаційного звіту для моделі, що використовує метод TF-IDF (рисунок 3.10), вказує на високий рівень ефективності моделі для класифікації новин як "Fake" або "True". Загальна точність (accuracy) моделі становить 0.89 або 89%, що свідчить про значну кількість правильних передбачень серед усіх тестових зразків. Загальний показник f1-score для класу "Fake" становить 0.86, що вказує на хорошу ефективність моделі в класифікації фейкових новин. Для класу "True" значення f1-score дорівнює 0.91, що є дуже високим результатом і свідчить про високу ефективність у класифікації правдивих новин. Макро-середнє значення f1-score для двох класів становить 0.88, що свідчить про добре збалансовану роботу моделі, хоча з деякою перевагою в класифікації правдивих новин. Зважене середнє значення f1-score дорівнює 0.89, що підтверджує високий рівень ефективності моделі в загальному, зокрема завдяки перевазі класу "True" в тестовому наборі, оскільки кількість правдивих новин значно перевищує кількість фейкових.

```
Accuracy (TF-IDF): 0.889372280919826
Classification Report (TF-IDF):
              precision    recall  f1-score   support

        Fake       0.92      0.81      0.86       686
        True       0.87      0.95      0.91       923

    accuracy                           0.89      1609
   macro avg       0.90      0.88      0.88      1609
weighted avg       0.89      0.89      0.89      1609
```

Рисунок 3.10 - Звіт про класифікацію фейкових та правдивих новин із використанням TF-IDF

Аналіз класифікаційного звіту для моделі, що використовує метод RAKE (рисунок 3.11), вказує на досить низьку ефективність моделі для класифікації новин як "Fake" або "True". Загальна точність (accuracy) моделі становить 0.58 або 58%, що свідчить про низький відсоток правильних передбачень серед усіх тестових зразків. Загальний показник f1-score для класу "Fake" становить 0.01, що вказує на дуже низьку ефективність моделі в класифікації фейкових новин. Модель майже не класифікує фейкові новини, що вимагає значних покращень для підвищення чутливості до цього класу. Для класу "True" значення f1-score дорівнює 0.73, що є досить високим результатом і свідчить про хорошу ефективність моделі у класифікації правдивих новин. Модель добре класифікує правдиві новини, але вимагає значних покращень для підвищення точності для фейкових новин. Макро-середнє значення f1-score для двох класів становить 0.37, що вказує на значну диспропорцію в ефективності моделі між двома класами. Це свідчить про необхідність вдосконалення моделі для досягнення більш збалансованих результатів. Зважене середнє значення f1-score дорівнює 0.43, що підтверджує низький рівень ефективності моделі в загальному, через значну перевагу класу "True" в тестовому наборі, оскільки кількість правдивих новин значно перевищує кількість фейкових.

```
Accuracy (RAKE): 0.5767557489123679
Classification Report (RAKE):
              precision    recall  f1-score   support

        Fake       1.00      0.01      0.01       686
        True       0.58      1.00      0.73       923

    accuracy                           0.58      1609
   macro avg       0.79      0.50      0.37      1609
weighted avg       0.76      0.58      0.43      1609
```

Рисунок 3.11 - Звіт про класифікацію фейкових та правдивих новин із використанням RAKE

Аналіз класифікаційного звіту для моделі, що використовує метод Yake! (рисунок 3.12), вказує на досить хороший рівень ефективності моделі для класифікації новин як "Fake" або "True". Загальна точність (accuracy) моделі становить 0.77 або 77%, що свідчить про значну кількість правильних

передбачень серед усіх тестових зразків. Загальний показник f1-score для класу "Fake" становить 0.73, що вказує на добрий рівень ефективності моделі в класифікації фейкових новин. Модель має деякі труднощі при класифікації фейкових новин, оскільки значення recall для цього класу є трохи меншим — 0.74. Для класу "True" значення f1-score дорівнює 0.79, що є хорошим результатом і свідчить про достатньо високу ефективність моделі у класифікації правдивих новин. Макро-середнє значення f1-score для двох класів становить 0.76, що свідчить про добре збалансовану роботу моделі, з деякою перевагою в класифікації правдивих новин. Це вказує на те, що модель здатна добре класифікувати обидва класи, але дещо ефективніше працює з класом "True". Зважене середнє значення f1-score дорівнює 0.77, що підтверджує високий рівень ефективності моделі в загальному, завдяки вищому представленню класу "True" в тестовому наборі.

```
Accuracy (Yake!): 0.7656929770043506
Classification Report (Yake!):
              precision    recall  f1-score   support

        Fake       0.72      0.74      0.73       686
        True       0.80      0.79      0.79       923

    accuracy                           0.77      1609
   macro avg       0.76      0.76      0.76      1609
weighted avg       0.77      0.77      0.77      1609
```

Рисунок 3.12 - Звіт про класифікацію фейкових та правдивих новин із використанням Yake!

Аналіз класифікаційного звіту для моделі, що використовує метод LSA (рисунок 3.13), вказує на високий рівень ефективності моделі для класифікації новин як "Fake" або "True". Загальна точність (accuracy) моделі становить 0.84 або 84%, що свідчить про значну кількість правильних передбачень серед усіх тестових зразків. Загальний показник f1-score для класу "Fake" становить 0.79, що вказує на добрий рівень ефективності моделі в класифікації фейкових новин. Для класу "True" значення f1-score дорівнює 0.87, що є високим результатом і свідчить про хорошу ефективність моделі у класифікації правдивих новин.

Макро-середнє значення f1-score для двох класів становить 0.83, що свідчить про добре збалансовану роботу моделі, з помірною перевагою в класифікації правдивих новин. Зважене середнє значення f1-score дорівнює 0.84, що підтверджує загальний високий рівень ефективності моделі. Це зумовлено тим, що модель демонструє добрі результати як для фейкових, так і для правдивих новин.

Цей результат свідчить про хорошу здатність моделі класифікувати обидва класи з високою точністю, хоча її ефективність дещо краща для класифікації правдивих новин.

```
Accuracy (LSA): 0.8377874456183965
Classification Report (LSA):
              precision    recall  f1-score   support

        Fake       0.86      0.74      0.79       686
        True       0.82      0.91      0.87       923

    accuracy                           0.84      1609
   macro avg       0.84      0.82      0.83      1609
weighted avg       0.84      0.84      0.84      1609
```

Рисунок 3.13 - Звіт про класифікацію фейкових та правдивих новин із використанням LSA

Аналіз класифікаційного звіту для моделі, що використовує метод LDA (рисунок 3.14), показує наявність деяких проблем у класифікації класу "Fake", хоча загальна точність класифікації є задовільною. Загальна точність (accuracy) моделі становить 0.68 або 68%, що свідчить про помірну кількість правильних передбачень серед усіх тестових зразків. Для класу "Fake" значення f1-score становить 0.61, що вказує на середній рівень ефективності моделі в класифікації фейкових новин. Точність цього класу складає 0.64, а повнота — 0.59, що вказує на проблеми з класифікацією фейкових новин, зокрема з точністю і чутливістю до цього класу. Для класу "True" модель демонструє більш високі показники, з f1-score 0.73, що свідчить про добру ефективність у класифікації правдивих новин. Точність для цього класу становить 0.71, а повнота — 0.76, що також свідчить про добру здатність моделі класифікувати правдиві новини. Макро-

середнє значення f1-score для двох класів становить 0.67, що вказує на помірну збалансованість здатності моделі класифікувати обидва класи. Зважене середнє значення f1-score дорівнює 0.68, що підтверджує загальний рівень ефективності моделі, але також вказує на необхідність поліпшення чутливості до класу "Fake" для покращення її точності.

Цей результат свідчить про те, що модель добре класифікує правдиві новини, але потребує удосконалення для покращення ефективності на класі "Fake".

```
Accuracy (LDA): 0.6842759477936606
Classification Report (LDA):
              precision    recall  f1-score   support

        Fake       0.64      0.59      0.61       686
        True       0.71      0.76      0.73       923

    accuracy                           0.68      1609
   macro avg       0.68      0.67      0.67      1609
weighted avg       0.68      0.68      0.68      1609
```

Рисунок 3.14 - Звіт про класифікацію фейкових та правдивих новин із використанням LDA

Аналіз класифікаційного звіту для моделі, що використовує метод TextRank (рисунок 3.15), вказує на високий рівень ефективності моделі для класифікації новин як "Fake" або "True". Загальна точність (accuracy) моделі становить 0.76 або 76%, що свідчить про значну кількість правильних передбачень серед усіх тестових зразків. Загальний показник f1-score для класу "Fake" становить 0.72, що вказує на добрий рівень ефективності моделі в класифікації фейкових новин. Для класу "True" значення f1-score дорівнює 0.78, що є хорошим результатом і свідчить про високу ефективність моделі у класифікації правдивих новин. Макро-середнє значення f1-score для двох класів становить 0.75, що свідчить про добре збалансовану роботу моделі, з деякою перевагою в класифікації правдивих новин. Зважене середнє значення f1-score дорівнює 0.76, що підтверджує загальний рівень ефективності моделі.

Цей результат свідчить про здатність моделі добре класифікувати як фейкові, так і правдиві новини, з більш високими результатами для правдивих новин.

```
Accuracy (TextRank): 0.7551274083281542
Classification Report (TextRank):
              precision    recall  f1-score   support

        Fake       0.70      0.75      0.72       686
        True       0.80      0.76      0.78       923

    accuracy                           0.76      1609
   macro avg       0.75      0.75      0.75      1609
weighted avg       0.76      0.76      0.76      1609
```

Рисунок 3.15 - Звіт про класифікацію фейкових та правдивих новин із використанням TextRank

Проведемо порівняльний аналіз ефективності різних методів класифікації новин, таких як TF-IDF, RAKE, Yake!, LSA, LDA та TextRank. Рисунок 3.16 показує точність кожного методу, виміряну за допомогою accuracy score, на основі тестового набору даних.

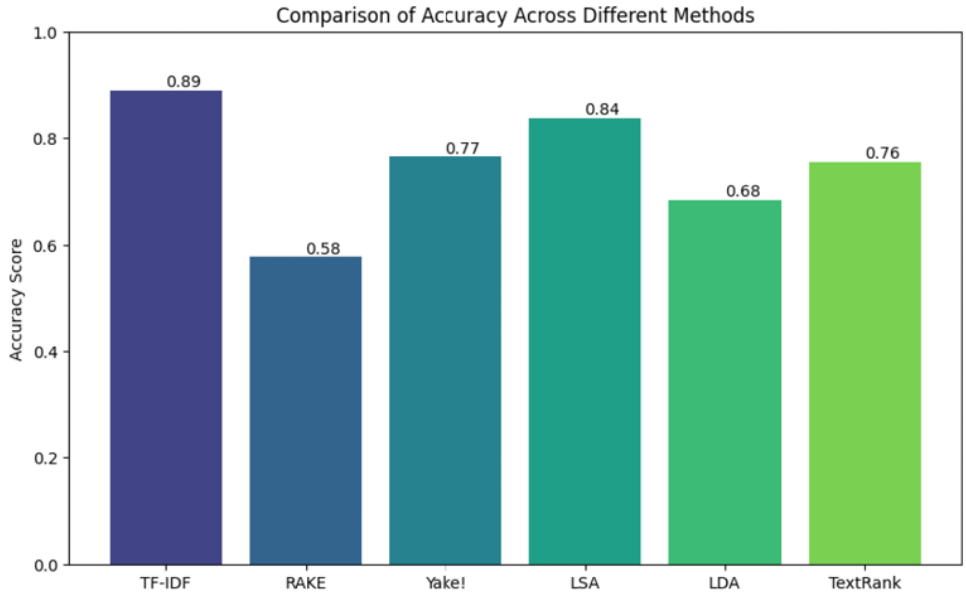


Рисунок 3.16 - Порівняльний аналіз точності методів вибору ключових слів для класифікації фейкових та правдивих новин

Згідно з представленими результатами, TF-IDF демонструє найвищу точність — 0.89, що є найкращим показником серед усіх методів. Це свідчить про те, що TF-IDF найкраще виконує завдання класифікації новин, а точність цього методу є найвищою серед усіх. Такий високий рівень точності може бути результатом того, що TF-IDF добре оцінює важливість кожного слова в контексті тексту, що особливо корисно для класифікації новин на базі тексту.

Методи RAKE, Yake!, LSA, LDA та TextRank показали різні результати:

- RAKE — точність 0.58, що вказує на найнижчу ефективність серед усіх методів;
- Yake! — точність 0.77, що є добрим результатом, але значно нижчим за TF-IDF;
- LSA — точність 0.84, що є добрим результатом, хоча все ще поступається TF-IDF;
- LDA — точність 0.68, що вказує на середній рівень ефективності;
- TextRank — точність 0.76, що також є хорошим результатом, але все одно відстає від TF-IDF.

Загалом, TF-IDF є найбільш ефективним методом серед усіх представлених, оскільки забезпечує найвищу точність для класифікації новин на основі їхнього тексту, оскільки він забезпечує найкраще виявлення важливих термінів і коректно визначає значення кожного слова в контексті. Інші методи, такі як RAKE, Yake!, LSA, LDA, і TextRank, мають дещо меншу точність, однак вони все одно забезпечують високу ефективність і можуть бути корисними в специфічних випадках, де важливо виділяти важливі фрази або контекстуальні теми, а не окремі терміни.

Таким чином, якщо основна мета — досягнення максимальної точності класифікації новин, рекомендується використовувати TF-IDF. Однак для більш складних задач, де важлива семантика або контекстуальні зв'язки між словами, можуть бути застосовані й інші методи, такі як TextRank чи LDA.

Згідно з порівняльним аналізом на основі діаграми (див. рисунок 3.16), TF-IDF виявляється найефективнішим методом класифікації новин серед усіх

представлених. Він демонструє найвищу точність — 0.89, що є значно вищим, ніж у інших методів, таких як RAKE, Yake!, LSA, LDA та TextRank, які показали точність від 0.58 до 0.84. Це свідчить про те, що TF-IDF найбільш точно виконує завдання класифікації новин, зокрема завдяки здатності оцінювати важливість кожного слова в контексті тексту.

TF-IDF працює шляхом надання ваги кожному слову в документі, що дозволяє ефективно відрізняти важливі терміни від загальних. Цей підхід ідеально підходить для задач, де основним завданням є правильне виявлення важливих термінів і точна класифікація на основі тексту. У порівнянні з іншими методами, такими як RAKE та Yake!, які орієнтовані на виділення ключових фраз, або LSA, LDA та TextRank, що базуються на семантичних зв'язках або латентних темах, TF-IDF є найбільш стабільним і точним для цього типу завдань.

Отже, якщо мета — досягнення максимальної точності класифікації новин, особливо в контексті тексту, TF-IDF є кращим вибором. Однак для більш складних задач, де важлива семантика або контекстуальні зв'язки між словами, можуть бути застосовані й інші методи, такі як TextRank чи LDA.

3.4 Ансамбль для класифікації фейкових та правдивих новин

У процесі дослідження була розроблена стекінг-модель класифікації новин на фейкові та правдиві [29], побудована на основі п’яти найточніших методів векторизації тексту: TF-IDF, RAKE, YAKE, LDA та TextRank. Для кожного з методів було створено окремий вектор ознак, а базовим класифікатором виступав Random Forest. Як мета-модель використовувалася логістична регресія, яка навчалася на прогнозах базових моделей у процесі крос-валідації.

Загальна точність роботи стекінг-моделі склала 89.56%, що є найвищим показником серед усіх раніше протестованих моделей. Це свідчить про ефективність використання ансамблевого підходу при наявності різноманітних джерел ознак. Для повної оцінки було побудовано classification report, що охоплює precision, recall та f1-score для кожного класу (рисунок 3.17).

```
✅ Accuracy (StackingClassifier): 0.8955873213175886
✅ Classification Report:
              precision    recall  f1-score   support

        Fake       0.91      0.84      0.87       686
        True       0.89      0.94      0.91       923

    accuracy                           0.90      1609
   macro avg       0.90      0.89      0.89      1609
weighted avg       0.90      0.90      0.89      1609
```

Рисунок 3.17 – Метрики якості класифікації для стекінг-моделі

Як видно з рисунка 3.18, модель показала високу точність розпізнавання фейкових новин — precision становив 0.91, що означає, що більшість новин, які класифіковані як "Fake", дійсно є такими. Водночас recall для цього класу склав 0.84, що свідчить про те, що деякі фейкові новини залишаються не виявленими.

Натомість для класу "True" модель виявила високу повноту (recall = 0.94), тобто переважну більшість правдивих новин було правильно класифіковано. Водночас точність для цього класу була дещо нижчою — 0.89, що може вказувати на наявність певної кількості фейкових новин, помилково віднесених до правдивих.

Для глибшого аналізу була побудована таблиця з десятьма прикладами неправильних класифікацій. Вона дозволяє оцінити контекст, у якому модель помилялася (див. рисунок 3.18).

```
Перші 10 помилкових класифікацій ансамблю:
                                                 Text True Label  \
17  Внаслідок ракетної атаки по Харкову вже відомо...       Fake
20  Дружина одного померлого сухого чоловіка прийш...       Fake
43  Атака Ісламської держави — Хорасан на московсь...       True
60  Міністр інфраструктури Олександр Кубраков пров...       Fake
62  Зеленський показав наслідки нічної атаки РФ пр...       Fake
70  "Він (Зеленський) був у всіх країнах, але не з...       Fake
76  Припиніть фінансувати Києві!В Америці з'явився...       Fake
84  Польські військові знали, що російська ракета,...       True
90  Західні країни навмисно приховують деталі хімі...       Fake
96  Польща готується до російського вторгнення. Бл...       Fake

   Predicted Label
17            True
20            True
43            Fake
60            True
62            True
70            True
76            True
84            Fake
90            True
96            True
```

Рисунок 3.18 – Приклади помилкової класифікації стекінг-моделлю

Як видно з рисунку 3.19, модель частіше припускається помилок у класифікації фейкових новин, які подані в офіційному або нейтральному стилі. Наприклад, новини про ракетні удари, цитати посадовців або зовнішньополітичні заяви, хоча й є фейковими за позначкою в датасеті, класифікуються як правдиві через використання формально-коректної мови.

Для кількісної оцінки помилок була побудована гістограма, що відображає кількість неправильних класифікацій по кожному класу (див. рисунок 3.19). Результати свідчать про те, що помилок у класифікації фейкових новин значно більше, ніж у класифікації правдивих.

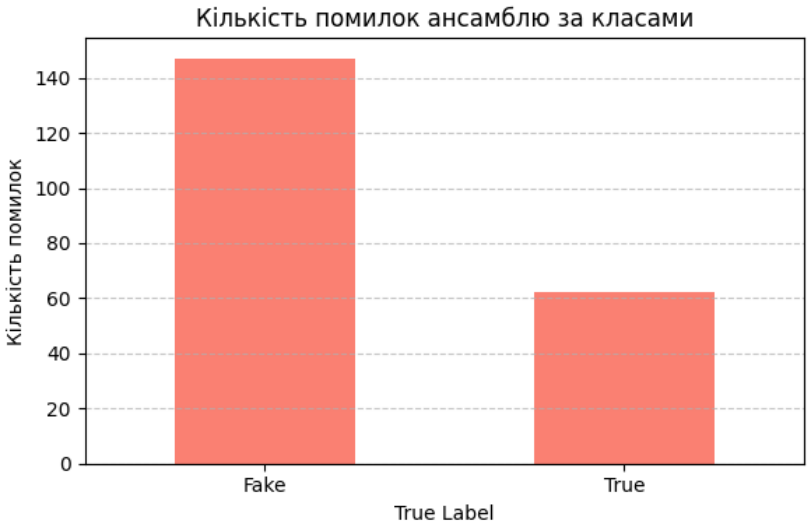


Рисунок 3.19 – Кількість помилок класифікації за класами

Це свідчить про певну схильність моделі до позитивної класифікації, що є типовим для задач, у яких структура мови фейкових новин імітує формальні ознаки правдивих повідомлень. Така поведінка моделі може бути виправданою з точки зору мінімізації помилок другого роду, проте водночас вимагає подальшого посилення здатності моделі розпізнавати маніпулятивний контекст.

Окрім оцінки точності на тестовій вибірці, було реалізовано функцію ручного введення новини для перевірки. Користувач має змогу ввести текст у

відповідне поле, після чого модель проводить обробку, векторизацію та виводить ймовірності віднесення новини до класів "Fake" і "True" (див. додаток Б).

У ході експериментів з ручною перевіркою новин було проаналізовано понад 10 прикладів. У випадках очевидних фейкових повідомлень модель демонструвала високу впевненість (понад 90%) у класифікації. Водночас при наявності неоднозначних або малозначущих повідомлень, ймовірності розподілялися більш збалансовано, іноді наближаючись до 50/50.

Приклади введення текстів користувачем і відповідні результати класифікації подано в додатку Б (рисунки Б.1–Б.4). Вони ілюструють інтерфейс взаємодії з користувачем, а також демонструють можливості системи щодо ймовірнісної оцінки довіри до новини.

Зокрема, новина про наступ РФ у квітні, хоча й виглядає тривожною, була класифікована як правдива з високою ймовірністю (понад 97%). Аналогічно, повідомлення про нібито інопланетне вторгнення — було класифіковано як фейк з високим ступенем впевненості, що свідчить про стабільну роботу моделі з крайніми прикладами.

Окремі новини, які містили нейтральну інформацію (наприклад, про погоду, ціну кави, економіку), викликали збалансовану реакцію моделі — ймовірність фейковості становила приблизно 40–60%, що можна інтерпретувати як "невизначеність" або потребу додаткового верифікаційного джерела.

На основі усіх результатів можна зробити висновок, що стекінг-модель є ефективним підходом для задач виявлення дезінформації [30], зокрема в українському інформаційному просторі. Її можливість поєднувати переваги різних підходів до виділення ключових слів значно підвищує якість передбачення.

При цьому подальше вдосконалення моделі може бути пов’язане з використанням трансформерів (наприклад, BERT або RoBERTa для української мови), адаптацією векторних ознак до тематичного контексту (context-aware embeddings), а також розширенням навчальної вибірки за рахунок нових датасетів, що відображають сучасні тенденції в медіа.

Таким чином, побудований ансамбль (зокрема стекінг-модель з точністю 90%) перевершив результати окремих класифікаторів, де найкращий з них (на основі TF-IDF) показував 89% точності. Хоча перевага здається незначною (приблизно 1 відсотка), вона підтверджує загальну тенденцію покращення завдяки узгодженому поєднанню різних методів векторизації та моделей класифікації. У результаті вдалося надійніше розпізнавати складні або «масковані» фейкові новини, що вказує на доцільність застосування ансамблевих підходів у реальних інформаційних системах.

ВИСНОВКИ

У рамках дослідження було успішно реалізовано всі поставлені завдання, що дозволило створити ефективну систему автоматичного виявлення фейкових новин на основі ключових слів із використанням сучасних методів машинного навчання. Основні результати можна підсумувати наступним чином:

1.Виконано огляд сучасних підходів до автоматичної класифікації фейкових новин. Розглянуто переваги та недоліки популярних методів векторизації тексту: TF-IDF, RAKE, YAKE!, LSA, LDA, TextRank, KeyBERT. Проведено огляд понад 10 досліджень, які показали, що TF-IDF та KeyBERT є найточнішими (до 91% і 88% відповідно), а YAKE! та RAKE — найшвидшими та ефективними для коротких текстів.

2. Побудовано формальні математичні вирази для основних етапів обробки тексту: токенізації, видалення стоп-слів, лематизації, векторизації (TF-IDF, RAKE, YAKE!, LSA, LDA, TextRank), а також для ансамблевих методів класифікації — VotingClassifier та StackingClassifier. Формули чітко описують процес обчислення TF, IDF, семантичних розкладів (SVD, Dirichlet), голосування та стекінгу.

3. Реалізовано універсальний модуль KeywordExtractor, який підтримує шість підходів видобутку ключових слів, включно з контекстно-залежними (LSA, LDA, TextRank). Алгоритм інтегрується в загальну архітектуру системи, що дозволяє обирати оптимальні ознаки для класифікації. Показано, що TF-IDF, YAKE! і LSA мають найвищу класифікаційну придатність при аналізі середніх і довгих текстів.

4. Розроблено повноцінний програмний модуль у середовищі Google Colab. Система підтримує обробку українськомовних текстів, включаючи токенізацію, очищення, лематизацію, побудову n-грам, векторизацію та класифікацію за допомогою RandomForest. Архітектура включає модулі: DataLoader, KeywordExtractor, Classifier, Visualizer, ReportGenerator.

5. Протестовано шість моделей векторизації у поєднанні з класифікатором Random Forest. Найвищу точність показала модель на основі TF-IDF — 89%, тоді як RAKE — найгірший результат — 58%. YAKE!, LSA, LDA та TextRank досягли точності в діапазоні 68–84%. Побудовано візуалізації розподілу класів, середньої довжини текстів, найчастіших слів та n-грам.

6. Запропонована стекінг-модель, яка поєднує ознаки від TF-IDF, RAKE, YAKE!, LDA та TextRank у рамках Random Forest + Logistic Regression, досягла найвищої загальної точності — 89.56%, f1-score для класу "Fake" — 0.86, для "True" — 0.91. Перевага стекінгу над найкращою окремою моделлю становила приблизно 1%, однак він дозволив краще виявляти складні випадки фейкових новин. Модель стабільно розпізнає крайні випадки, має інтерфейс для ручного введення, оцінки довіри та помилкових класифікацій. Основним напрямом подальшого розвитку є інтеграція трансформерів (BERT, RoBERTa), що забезпечать контекстно-чутливу обробку українських текстів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ліп’яніна-Гончаренко, Х., & Телька, М. (2024). Моделі виявлення фейкових новин у медіа просторі. Західноукраїнський національний університет. С. 127.
2. Комар М. П. Алгоритми для класифікації текстів за категоріями. Журнал з інформаційних технологій. 2023. № 2. С. 54–61.
3. Михальчук Н. О. Методи обробки природної мови для виявлення фейкових новин. Наукові записки. 2024. Т. 45. С. 36–44.
4. Новосад С. О. Аналіз текстових даних у боротьбі з фейковими новинами. Матеріали Міжнародної конференції з інтелектуальних систем. Київ, 2024. С. 93–97.
5. Шатарський А., Піун О. Системи класифікації для виявлення фейкових новин. Львів: Інформатика, 2023. С. 98.
6. Дзюбановська Н. І. Онлайн навчання для виявлення фейкових новин у реальному часі. Наукові дослідження у галузі машинного навчання. Чернівці, 2024. С. 122–128.
7. Галяс Ю. Інтеграція методів машинного навчання для підвищення точності виявлення фейкових новин. Матеріали конференції з інтелектуальних систем. Тернопіль, 2024. С. 65–71.
8. Kossy A., Smirnov D. Information warfare and fake news in the context of Ukraine's national security. Journal of Information Security. 2023. Vol. 11(2). P. 102–110.
9. Tkachuk V. Impact of fake news on electoral processes: Case of Ukraine. Political Studies Review. 2024. Vol. 22(1). P. 77–85.
10. Ivanov S. Fake news and hybrid warfare: The case of Ukraine. Global Security Review. 2023. Vol. 8(4). P. 92–101.
11. Yermolenko D. International impact of fake news on national security. International Relations Journal. 2024. Vol. 30(3). P. 67–74.

12. Shulga P. Technologies for detecting fake news in modern media. Cybersecurity and Information Technology. 2024. № 1. С. 45–52.

13. Lipianina-Honcharenko K., Lendiuk D., Melnyk M. N. Evaluation of the Keyword Selection Methods Effectiveness for the Fake News Classification. CEUR Workshop Proceedings. 2024. Vol. 3688. P. 404–414. http://ceur-ws.org/Vol-3688/paper32.pdf.

14. Mishchenko L., Klymenko I., Tkachenko V. The fake news recognition method based on Naïve Bayes with improved TF-IDF algorithm. In: Advances in Intelligent Systems and Computing. Springer, 2023. Vol. 1456. P. 234–244.

15. Sriram S. An Evaluation of Text Representation Techniques for Fake News Detection. 2020. https://doi.org/10.21427/5519-h979

16. Li Q., Qu J. Automatic detection of fake crowdfunding projects. International Science Journal of Emerging Technologies (ISJET). 2023. Vol. 15, No 2. P. 85–92.

17. Dubey Y., Wankhede P., Borkar A., Borkar T. Framework for fake news classification using vectorization and machine learning. Combating Fake News with Computational Intelligence Techniques. Cham : Springer, 2022. P. 327–343. DOI: 10.1007/978-3-030-90087-8_16.

18. Farinha A. F. N. Extracting keywords from tweets: Escola Superior de Tecnologia de Tomar, 2018. – 77 c.

19. Landu T. T., Bousso M., Loum M. A., Sawadogo I., Dia Y., Sall O., Faty L., Karim M., Sylla M. Benchmarking Unsupervised Keyword Extraction Algorithms from Online Senegalese News Articles. Lecture Notes in Networks and Systems. – Singapore : Springer Nature, 2024. Vol. 325–338. DOI: 10.1007/978-981-99-8031-4_29.

20. Nadim M., Akopian D., Matamoros A. A comparative assessment of unsupervised keyword extraction tools. IEEE Access. 2023. Vol. 11. P. 14567–14577. https://doi.org/10.1109/ACCESS.2023.3259310

21. Campos R. et al. YAKE! Keyword extraction from single documents using multiple local features. Information Sciences. 2020. Vol. 509. P. 257–289. https://doi.org/10.1016/j.ins.2019.07.018

22. Jafari B. M., Luo X., Jafari A. Unsupervised keyword extraction for hashtag recommendation in social media. In: Proceedings of the 36th International FLAIRS Conference (FLAIRS-36). 2023. P. 109–113. https://doi.org/10.32473/flairs.36.133280

23. Aizawa A. An information-theoretic perspective of tf–idf measures. Information Processing & Management. 2003. Vol. 39(1). P. 45–65. https://doi.org/10.1016/S0306-4573(02)00065-5

24. Rose S. et al. Automatic keyword extraction from individual documents. In: Text mining: Applications and theory. Wiley, 2010. P. 1–20. DOI: 10.1002/9780470689646.ch1.

25. Campos R. et al. YAKE! Keyword extraction from single documents using multiple local features. Information Sciences. 2020. Vol. 509. P. 257–289. https://doi.org/10.1016/j.ins.2019.07.018

26. Landauer T. K., Foltz P. W., Laham D. An introduction to latent semantic analysis. Discourse Processes. 1998. Vol. 25(2–3). P. 259–284. https://doi.org/10.1080/01638539809544949

27. Lipianina-Honcharenko K. ., Lendiuk, D., Ivaniush, A., Soia, M., Yurkiv, K. An intelligent method for forming the advertising content of higher education institutions based on semantic analysis. In: ICTERI 2021. Springer. P. 169–182. https://doi.org/10.1007/978-3-030-69472-7_19

28. Blei D. M., Ng A. Y., Jordan M. I. Latent Dirichlet Allocation. Journal of Machine Learning Research. 2003. Vol. 3. P. 993–1022. https://doi.org/10.1162/153244303322724072

29. Lipianina-Honcharenko K., Maika N., Sachenko S., Kopania L., Soia M. A Cyclical Approach to Legal Document Analysis: Leveraging AI for Strategic Policy Evaluation. CEUR Workshop Proceedings. 2024. Vol. 3736. P. 201–211.

30. Lipianina-Honcharenko K., Soia M., Yurkiv K., Ivasechko A. Evaluation of the Effectiveness of Machine Learning Methods for Detecting Disinformation in Ukrainian Text Data. CEUR Workshop Proceedings. 2024. Vol. 3702. P. 97–109.

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

Додаток А

Код реалізації

```
# Імпорт бібліотек
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import nltk
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier, VotingClassifier, StackingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.decomposition import TruncatedSVD, LatentDirichletAllocation
from sklearn.preprocessing import LabelEncoder
from nltk.tokenize import RegexpTokenizer
import networkx as nx
import yake
from rake_nltk import Rake
from multi_rake import Rake as MultiRake
from keybert import KeyBERT

# Завантаження та обробка даних
data = pd.read_csv('combined_dataset.csv')
tokenizer = RegexpTokenizer(r'\w+')
with open('stopwords_ua.txt', 'r', encoding='utf-8') as f:
    ukrainian_stop_words = set(f.read().splitlines())
data['processed_text'] = data['Text'].fillna('').astype(str).apply(
    lambda x: ' '.join([word.lower() for word in tokenizer.tokenize(x) if word.lower() not in ukrainian_stop_words])
)

# Підготовка меток
le = LabelEncoder()
y = le.fit_transform(data['Label'])

# Методи виділення ключових слів і векторизація
vectorizers = {}
results = {}

# TF-IDF
tfidf = TfidfVectorizer(max_features=5000)
X_tfidf = tfidf.fit_transform(data['processed_text'])
vectorizers['TF-IDF'] = X_tfidf

# RAKE
multi_rake = MultiRake(language_code='uk')
data['rake_keywords'] = data['processed_text'].apply(lambda x: ' '.join([kw[0] for kw in multi_rake.apply(x)]))
vectorizers['RAKE'] = CountVectorizer(max_features=5000).fit_transform(data['rake_keywords'])
```

```
        # YAKE
        yake_extractor = yake.KeywordExtractor(lan='uk', n=1, top=10)
        data['yake_keywords'] = data['processed_text'].apply(lambda x: ' '.join([kw[0] for kw in yake_extractor.extract_keywords(x)]))
        vectorizers['Yake!'] = CountVectorizer(max_features=5000).fit_transform(data['yake_keywords'])

        # LSA
        lsa = TruncatedSVD(n_components=100)
        vectorizers['LSA'] = lsa.fit_transform(X_tfidf)

        # LDA
        bow = CountVectorizer(max_features=5000).fit_transform(data['processed_text'])
        lda = LatentDirichletAllocation(n_components=10, random_state=42)
        vectorizers['LDA'] = lda.fit_transform(bow)

        # TextRank
        def textrank_keywords(text):
            words = tokenizer.tokenize(text.lower())
            graph = nx.Graph()
            for i in range(len(words)):
                for j in range(i + 1, len(words)):
                    graph.add_edge(words[i], words[j])
            scores = nx.pagerank(graph)
            return ' '.join(sorted(scores, key=scores.get, reverse=True)[:10])
        data['textrank_keywords'] = data['processed_text'].apply(textrank_keywords)
        vectorizers['TextRank'] = CountVectorizer(max_features=5000).fit_transform(data['textrank_keywords'])

        # Класифікація RandomForest для кожного методу
        for method, X in vectorizers.items():
            X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
            clf = RandomForestClassifier(n_estimators=100, random_state=42)
            clf.fit(X_train, y_train)
            y_pred = clf.predict(X_test)
            acc = accuracy_score(y_test, y_pred)
            print(f"Accuracy ({method}):", acc)
            results[method] = acc

        # Ансамблева модель VotingClassifier
        top_methods = sorted(results.items(), key=lambda x: x[1], reverse=True)[:5]
        X_ensemble = vectorizers[top_methods[0][0]]
        X_train, X_test, y_train, y_test = train_test_split(X_ensemble, y, test_size=0.2, random_state=42)
        voting_clf = VotingClassifier(estimators=[
            (name, RandomForestClassifier(n_estimators=100, random_state=42)) for name, _ in top_methods
        ], voting='hard')
        voting_clf.fit(X_train, y_train)
        results['Ensemble Top 5'] = accuracy_score(y_test, voting_clf.predict(X_test))

        # Ансамблева модель StackingClassifier
        stacking_clf = StackingClassifier(
            estimators=[(name, RandomForestClassifier(n_estimators=100, random_state=42)) for name, _ in top_methods],
```

```
    final_estimator=LogisticRegression(max_iter=1000),
    cv=5
)
stacking_clf.fit(X_train, y_train)
results['StackingClassifier'] = accuracy_score(y_test, stacking_clf.predict(X_test))

# Функція для передбачення
def predict_news(text, model=stacking_clf, vectorizer=tfidf, label_encoder=le):
    tokens = tokenizer.tokenize(text.lower())
    tokens = [word for word in tokens if word not in ukrainian_stop_words]
    X_input = vectorizer.transform([' '.join(tokens)])
    proba = model.predict_proba(X_input)[0]
    labels = label_encoder.inverse_transform([0, 1])
    print("📊 Ймовірності класифікації:")
    for i, label in enumerate(labels):
        print(f"  {label}: {proba[i]*100:.2f}%")
```

Додаток Б

Результати експерименту

```
Введений текст: Це екстрене повідомлення: інопланетяни висадились у центрі Києва!...
Ймовірності класифікації:
  Fake: 76.10%
  True: 23.90%
```

```
[40]  1 user_input = input("Введіть новину для перевірки: ")
      2 predict_news(user_input)
      3
```

```
Введіть новину для перевірки:  ❗рф готує наступ у квітні. Добре укомплектовані резерви готуються на полігонах вже більше місяця,
Введений текст:  ❗рф готує наступ у квітні. Добре укомплектовані резерви готуються на полігонах ...
Ймовірності класифікації:
  Fake: 2.55%
  True: 97.45%
```

```
[42]  1 user_input = input("Введіть новину для перевірки: ")
      2 predict_news(user_input)
```

```
Введіть новину для перевірки: Переговорник путіна цього тижня поїде у Вашингтон на зустріч з Віткоффом,

Введений текст:
Переговорник путіна цього тижня поїде у Вашингтон на зустріч з Віткоффом,

Ймовірності класифікації:
  Fake: 84.93%
  True: 15.07%
```

```
[43]  1 user_input = input("Введіть новину для перевірки: ")
      2 predict_news(user_input)
```

```
Введіть новину для перевірки: Жахливі кадри з Харкова. Два влучання по цивільним підприємствам через масовану атаку БпЛА рф вночі.

Введений текст:
Жахливі кадри з Харкова. Два влучання по цивільним підприємствам через масовану атаку БпЛА рф вночі.

Ймовірності класифікації:
  Fake: 2.26%
  True: 97.74%
```

Рисунок Б.1 - Прогнозування новин з використанням машинного навчання для виявлення фейкових новин

```
  1 user_input = input("Введіть новину для перевірки: ")
  2 predict_news(user_input)
```

```
Введіть новину для перевірки: Сенатори США погрожують запровадити мита у 500% проти країн, які закуповують російську нафту, газ і уран, якщо москва відмовиться від мирних перемовин,

Введений текст:
Сенатори США погрожують запровадити мита у 500% проти країн, які закуповують російську нафту, газ і уран, якщо москва відмовиться від мирних перемовин,

Ймовірності класифікації:
  Fake: 98.35%
  True: 1.65%
```

```
[45]  1 user_input = input("Введіть новину для перевірки: ")
      2 predict_news(user_input)
      3
```

```
Введіть новину для перевірки: Сенатор-демократ Корі Букер 25 годин (!) виступав проти політики Трампа.

Введений текст:
Сенатор-демократ Корі Букер 25 годин (!) виступав проти політики Трампа.

Ймовірності класифікації:
  Fake: 53.21%
  True: 46.79%
```

```
[46]  1 user_input = input("Введіть новину для перевірки: ")
      2 predict_news(user_input)
```

```
Введіть новину для перевірки: Через війну в Україні попит на віддалених працівників зріс на 1061%.

Введений текст:
Через війну в Україні попит на віддалених працівників зріс на 1061%.

Ймовірності класифікації:
  Fake: 96.87%
  True: 3.13%
```

```
[47]  1 user_input = input("Введіть новину для перевірки: ")
      2 predict_news(user_input)
```

```
Введіть новину для перевірки: Кава б’є цінові рекорди: вартість зерен зросла у вартості в 3,5 рази

Введений текст:
Кава б’є цінові рекорди: вартість зерен зросла у вартості в 3,5 рази

Ймовірності класифікації:
  Fake: 47.19%
  True: 52.81%
```

Рисунок Б.2 - Класифікація новин за допомогою машинного навчання для виявлення фейкових та реальних повідомлень

```
[48]  1 user_input = input("Введіть новину для перевірки: ")
      2 predict_news(user_input)

Введіть новину для перевірки: Українець підпалив свого роботодавця в Польщі, – польська поліція

Введений текст:
Українець підпалив свого роботодавця в Польщі, – польська поліція

Ймовірності класифікації:
  Fake: 61.03%
  True: 38.97%
```

```
  1 user_input = input("Введіть новину для перевірки: ")
  2 predict_news(user_input)
  3

Введіть новину для перевірки: Уже з 5 квітня в Україні можливі заморозки, сніг та похолодання

Введений текст:
Уже з 5 квітня в Україні можливі заморозки, сніг та похолодання

Ймовірності класифікації:
  Fake: 31.73%
  True: 68.27%
```

Рисунок Б.3 - Виявлення фейкових новин на основі машинного навчання: Прогнозування правдивості повідомлень

```
[50]  1 user_input = input("Введіть новину для перевірки: ")
      2 predict_news(user_input)

Введіть новину для перевірки: Штучний інтелект – у фокусі ІТ-індустрії України

Введений текст:
Штучний інтелект – у фокусі ІТ-індустрії України

Ймовірності класифікації:
  Fake: 95.03%
  True: 4.97%
```

```
  1 user_input = input("Введіть новину для перевірки: ")
  2 predict_news(user_input)

Введіть новину для перевірки: рф вдарила по військовому госпіталю у Харкові, – Генштаб (https://t.me/GeneralStaffZSU/22501)

Введений текст:
рф вдарила по військовому госпіталю у Харкові, – Генштаб (https://t.me/GeneralStaffZSU/22501)

Ймовірності класифікації:
  Fake: 1.89%
  True: 98.11%
```

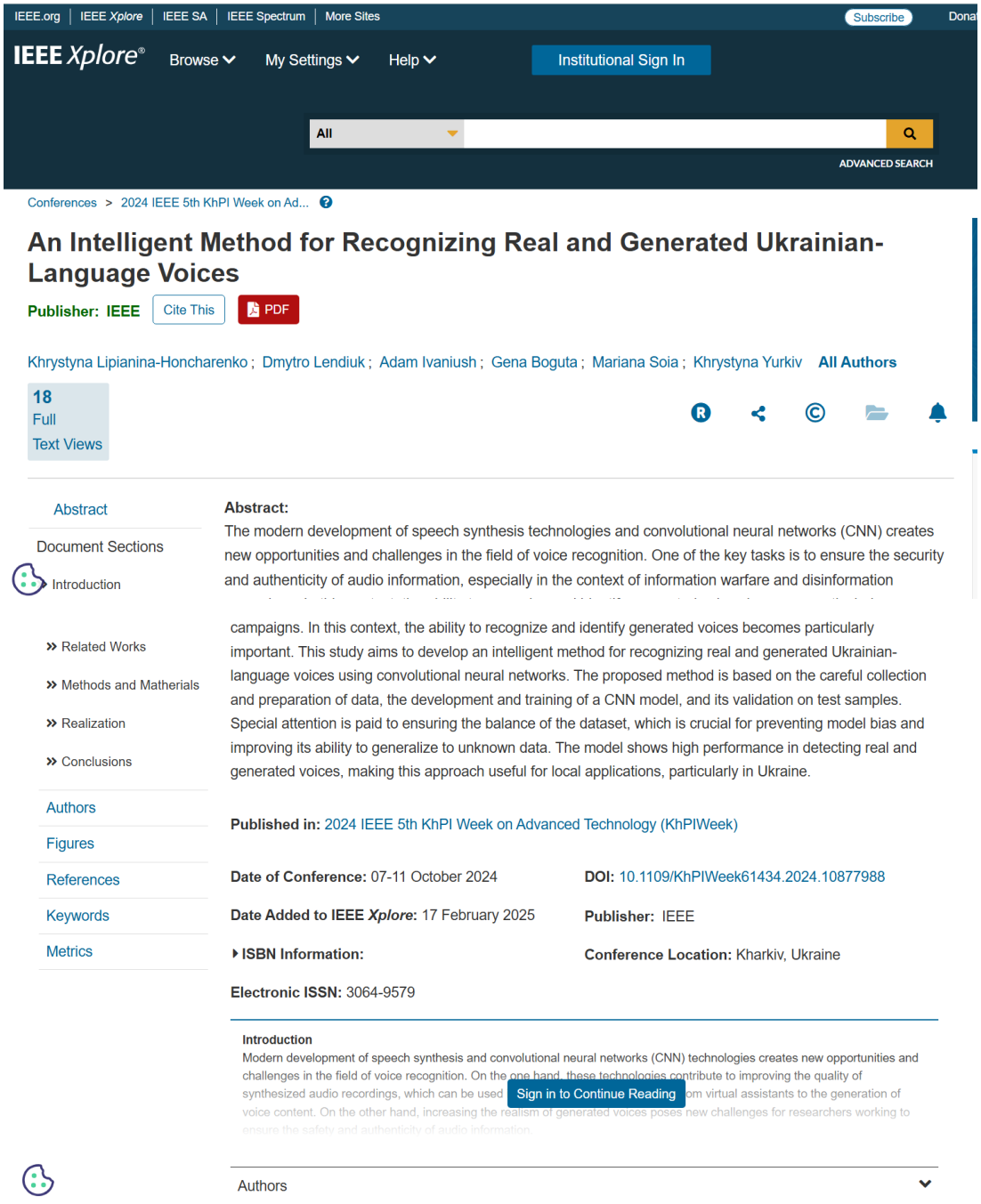
```
  1 user_input = input("Введіть новину для перевірки: ")
  2 predict_news(user_input)

Введіть новину для перевірки: На дану хвилину 27 постраждалих внаслідок атаки БПЛА на Харків, – Терехов

Введений текст:
На дану хвилину 27 постраждалих внаслідок атаки БПЛА на Харків, – Терехов

Ймовірності класифікації:
  Fake: 3.55%
  True: 96.45%
```

```
[54]  1 user_input = input("Введіть новину для перевірки: ")
      2 predict_news(user_input)

Введіть новину для перевірки: На ОЛХ – знайдеться все: хтось на Дніпропетровщині (https://t.me/+MvxTglQ4e21hZmUy) вирішив продати радянський танк Т-10 за 7,3 мільйона

Введений текст:
На ОЛХ – знайдеться все: хтось на Дніпропетровщині (https://t.me/+MvxTglQ4e21hZmUy) вирішив продати радянський танк Т-10 за 7,3 мільйона

Ймовірності класифікації:
  Fake: 11.01%
  True: 88.99%
```

Рисунок Б.4 - Моделювання правдивості новин з використанням машинного навчання

Додаток В

Апробація отриманих результатів



IEEE.org | IEEE Xplore | IEEE SA | IEEE Spectrum | More Sites | Subscribe | Donat

IEEE Xplore® Browse My Settings Help Institutional Sign In

All

ADVANCED SEARCH

Conferences > 2024 IEEE 5th KhPI Week on Ad...

An Intelligent Method for Recognizing Real and Generated Ukrainian-Language Voices

Publisher: IEEE Cite This PDF

Khrystyna Lipianina-Honcharenko ; Dmytro Lendiuk ; Adam Ivaniush ; Gena Boguta ; Mariana Soia ; Khrystyna Yurkiv All Authors

18 Full Text Views

Abstract

Document Sections

Introduction

Related Works

Methods and Matherials

Realization

Conclusions

Authors

Figures

References

Keywords

Metrics

Abstract:

The modern development of speech synthesis technologies and convolutional neural networks (CNN) creates new opportunities and challenges in the field of voice recognition. One of the key tasks is to ensure the security and authenticity of audio information, especially in the context of information warfare and disinformation campaigns. In this context, the ability to recognize and identify generated voices becomes particularly important. This study aims to develop an intelligent method for recognizing real and generated Ukrainian-language voices using convolutional neural networks. The proposed method is based on the careful collection and preparation of data, the development and training of a CNN model, and its validation on test samples. Special attention is paid to ensuring the balance of the dataset, which is crucial for preventing model bias and improving its ability to generalize to unknown data. The model shows high performance in detecting real and generated voices, making this approach useful for local applications, particularly in Ukraine.

Published in: 2024 IEEE 5th KhPI Week on Advanced Technology (KhPIWeek)

Date of Conference: 07-11 October 2024

DOI: 10.1109/KhPIWeek61434.2024.10877988

Date Added to IEEE *Xplore*: 17 February 2025

Publisher: IEEE

ISBN Information:

Conference Location: Kharkiv, Ukraine

Electronic ISSN: 3064-9579

Introduction

Modern development of speech synthesis and convolutional neural networks (CNN) technologies creates new opportunities and challenges in the field of voice recognition. On the one hand, these technologies contribute to improving the quality of synthesized audio recordings, which can be used [Sign in to Continue Reading] om virtual assistants to the generation of voice content. On the other hand, increasing the realism of generated voices poses new challenges for researchers working to ensure the safety and authenticity of audio information.

Authors

A Cyclical Approach to Legal Document Analysis: Leveraging AI for Strategic Policy Evaluation

Khrystyna Lipianina-Honcharenko[1,†,*], Nataliia Maika[1,†], Svitlana Sachenko[1,†], Lukasz Kopania[2,†] and Mariana Soia[1,†]

[1]West Ukrainian National University, Lvivska str., 11, Ternopil, 46000, Ukraine
[2]Kazimierz Pulaski University of Technology and Humanities in Radom, Department of Informatics, Jacek Malczewski str., 29, Radom, 26600, Poland

Abstract

This article presents an approach to the analysis of legal documents that integrates artificial intelligence (AI) technologies to optimize processes of data extraction, classification, and compliance assessment to legal norms. With a focus on the accuracy of analysis, the study demonstrates how the application of advanced machine learning algorithms and natural language processing can significantly enhance the quality of legal text processing. The results show the AI algorithms' average weighted accuracy in analyzing AI development strategies by countries and categories at 0.91, with the highest accuracy in the "Specific Targets" category (1.0) and "AI Development Strategy" (0.99), indicating a high potential for AI application in legal analytics. These findings underscore the need for further research to optimize AI algorithms aimed at improving analysis accuracy in complex legal domains.

Keywords

Artificial intelligence, Legal analysis of documents, Natural language processing, Machine learning.

1. Introduction

Today, it is practically impossible to imagine our lives without using artificial intelligence (AI) in the modern world. In his blog, Bill Gates concluded that the widespread use of neural networks will be possible by 2030, with AI tools being used everywhere in the USA in just 1.5 years, and in African countries — in 3 years. The co-founder of the technology corporation emphasizes that preparations are still underway for a technological boom [1]. Developed and developing countries have already approved national programs and AI development plans, which indicates an awareness of the role of new technologies in the development and functioning of states and, in perspective, the formation of a humane digitized environment. Moreover, most governments declare their leadership in technology development (namely by attracting significant investments in research and focusing on the development of technologies

ICyberPhyS-2024: 1st International Workshop on Intelligent & CyberPhysical Systems, June 28, 2024, Khmelnytskyi, Ukraine
* Corresponding author.
† These authors contributed equally.
mailto:xrustya.com@gmail.com (K. Lipianina-Honcharenko); N.Maika_law@meta.ua (N. Maika); as@wunu.edu.ua (S. Sachenko); l.kopania@uthrad.pl (L. Kopania); mariankasoa@gmail.com (M. Soia)
0000-0002-2441-6292 (K. Lipianina-Honcharenko); 0000-0002-5676-980X (N. Maika); 0000-0001-8225-1820 (S. Sachenko); 0000-0002-7318-4803 (L. Kopania); 0009-0001-3719-6460 (M. Soia)

in the field of national defense). It is projected that by 2030, the economic benefits of using AI will be greatest in China and North America, which will constitute 70% of the global economic impact of AI [2].

AI is one of the key technologies of the new stage of economic development (known as "Industrial Internet of Things" or "Industry 4.0"), and the formation of a new society ("Super Smart Society or Society 5.0").

The implementation of AI technologies influences a country's competitive potential, especially regarding national security and defense, industrial, and social development. A country's approved development strategy, serving as the foundation of a defined policy course regarding the regulation and introduction of new technologies, is a means to minimize the threats and challenges associated with the development of these technologies.

The integral map of concepts for the development of new technologies is diverse and distinct. Despite the similarity of general tasks, principles, and methods, these strategies must each time be adapted to unique conditions, leading to different paths toward the more or less similar general goals of the international community.

Therefore, a hot problem is the analysis of AI development strategies (of developed and developing countries) regarding key aspects of development and legal implementation at the national level.

Considering the above and the relevance of the research, this article presents a new cyclical approach to the analysis of legal documents, based on the application of advanced AI algorithms. The aim of the study is to develop a comprehensive method that will automate the processes of analysis, classification, and assessment of compliance of legal documents with established norms and standards. It is demonstrated how the application of this approach can optimize the processes of working with legal texts, improve the accuracy of their analysis, and contribute to effective decision-making in the legal sphere. The study also emphasizes the importance of regulation and standardization in the use of AI technologies in jurisprudence, considering potential threats to data confidentiality and security.

The structure of this work is organized in such a way that it initially presents a review of existing AI research and technologies used in the legal field, and then describes in detail the developed cyclical approach and its implementation through a series of stages of legal document analysis. The conclusions emphasize the significance of the obtained results and the prospects for further research in this area.

2. Related Work

The advancement and expansion of AI over the years have been fostered by a continuous search in algorithms, machine learning methods, and integrating statistical analysis into understanding the world at large [3].

The field of science and technology, particularly in AI inventiveness and creativity, had earlier been subjected to human intervention. However, with the recent development in AI and the era of «big data», machines can now invent devices and processes autonomously [4]. This has led to an endless application of artificial intelligence in several sectors and industries such as healthcare, real estate, entertainment, logistics and transport, banking and financial services, travel, retail and e-commerce, manufacturing and food technology [5]. The field of jurisprudence is no exception too of using AI. The topic of technology in the legal sphere is quite extensive, which creates the need to investigate all relevant events and processes that currently have the greatest impact on the legal system and social relations [6] Legal technology encompasses the use of various technological tools and innovations to improve legal practice, enhance legal processes, and facilitate better access to justice. These technologies aim to improve efficiency, accuracy, and accessibility in the legal field [7]. The majority of the resources in this field are presented in text forms, such as judgment documents, contracts, and legal opinions. Therefore, most Legal AI tasks are based on Natural Language Processing (NLP) technologies. Many tasks in the legal domain require the expertise of legal practitioners and a thorough understanding of various legal documents. Almost all models are better at case analyzing than knowledge understanding, which proves that knowledge modelling is still challenging for existing methods. How to model legal knowledge to LQA is essential as legal knowledge is the foundation of LQA [8].

The use of AI in jurisprudence opens new horizons for litigation, legal analysis, and the administration of justice. Research in this field shows that AI can help reduce search time, improve quality, and effectively extract relevant information from expanding textual sources, such as decisions of Brazilian courts [9].

In the context of the Indian judicial system, AI is used for data storage and retrieval, multi-party litigation systems, online dispute resolution platforms, and even predicting court decisions [10].

The need for regulation of AI technologies in jurisprudence is critical for standardizing policy and assessing data privacy threats [11]. AI can also be implemented in justice as a psychological phenomenon, integrating legal and psychological aspects to establish the limits of its application in law enforcement activities and court proceedings [12]. The emergence of AI-related crimes poses global challenges to jurisprudence, prompting potential improvements in legislation to respond to these new types of criminal activity [13].

Research in the field of document analysis with cyclical processing, information extraction, machine learning, and NLP demonstrates a wide range of applications and methodologies. The study [14] considers the analysis of citations using NLP and ML techniques to assess the impact of citations in scientific assessment.

In [15], different approaches to keyword extraction and synonym generation are explored, highlighting the benefits of the extreme learning model. In [16], the application of NLP and ML to patient reviews is analyzed, [17] focuses on text classification in multi-turn corpora, and [18] investigates deep learning for document analysis. These studies highlight the significant potential of NLP and ML in diverse applications, from analyzing reviews to classifying texts.

Below is a comparative Table 1 of approaches and quantitative assessments of the closest analogues among scientific articles.

Compared to the closest studies [24-26], this study demonstrates higher accuracy (0.91) in the analysis and classification of legal documents due to the integration of advanced machine learning and natural language processing algorithms.

Therefore, this study not only ensures high accuracy of analysis but also shows better adaptation to various aspects of legal documents, making it a more efficient and reliable tool for legal analysis and strategic policy evaluation.

Table 1
Comparative Analysis of Closest Analogues

Study	Approach	Quantitative Assessment
[24]	Use of AI-based software to improve speed and accuracy of analysis	No specific numerical values given, but a significant increase in speed and accuracy noted
[25]	Implementation of deep learning for text classification in legal document review	Classification accuracy: 0.87, performance across multiple document classes
[26]	Integration of AI and machine learning for document analysis and accuracy prediction	Prediction accuracy: 0.89, improvement in document analysis accuracy

3. Approach to the analysis of documents with cyclic processing

The approach to analyzing documents and extracting relevant information can be considered a sequence of operations that includes text analysis, pattern recognition, and data categorization. The objective is to transform unstructured textual data (documents) into a structured format (table) based on predefined categories. This includes several steps:

1. Text Preprocessing [19]. Before analyzing documents, it is necessary to convert them from raw text data into a format that is easily processed. This process includes [20]

- Tokenization: the division of text into words or phrases.
- Normalization: converting text to a uniform case, removing punctuation.
- Stemming or Lemmatization: reducing words to their base or root form.

These steps simplify the text and facilitate the detection of patterns. Formally, for a set of documents $D = \{d_1, d_2, \ldots, d_m\}$, each document d_i is subjected to preprocessing to obtain the processed documentd_i'.

2. Feature Extraction [21]. The process of identifying text attributes that may indicate its relation to predefined categories. Mathematically, this is described as: transforming the text data d_i' into a feature vector$x_i \in R_n$, where each dimension corresponds to a separate feature. This step converts the text into a form suitable for machine learning analysis.

3. Classification [22, 27, 28]. The use of document features to classify it into predefined categories using machine learning models. The classification function $f: \mathbb{R}^n \to \{1,2,\ldots,k\}$ maps the feature vector x_i to a label y_i, that corresponds to the category. Choosing an adequate model and its parameters is crucial for ensuring accuracy.

4. Information Storage. The classified information is stored in a structured format. This can be represented by a matrix $M \in \mathbb{R}^{m \times k}$, where each row corresponds to a document, and each column corresponds to a category. The element M_{ij} is filled based on the classification results of document i for category j.

This approach integrates scientific text processing methods and mathematical principles of classification to automate the process of extracting information from textual documents, ensuring efficient and accurate data processing. We will illustrate this approach with Figure 1,

which depicts an algorithm with several steps, beginning with the definition of the table structure ("Step 1: Define Structure"), moving to the listing of documents for analysis ("Step 2: List Documents"), processing each document to extract information corresponding to each category ("Step 3: Process Documents"), filling the table with extracted information ("Step 4: Populate Table"), exporting the table to an Excel file ("Step 5: Export to Excel"), and providing access or a method for downloading or accessing the Excel file ("Step 6: Access Excel File"). Steps 3 and 4 indicate a focus on detail: Step 3 involves the cyclical processing of each document to extract information, and Step 4 involves the cyclical creation of new rows in the table with the extracted information.

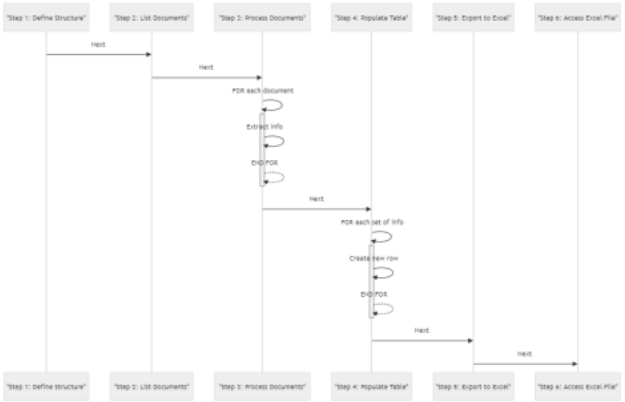


Figure 1: Document Analysis with Cyclical Processing.

4. Implementation

To implement the proposed method (see section 3) for analyzing AI development strategies, NLP and machine learning methods [23] were used, specifically Python libraries such as NLTK for text preprocessing, Scikit-learn for feature extraction and classification, and Pandas for structuring and storing data in tabular format.

For achieving specific goals, data were used that were collected from official state websites, governments, and international organizations (namely the full text of concepts, development strategies approved at state and international levels).

Based on the analysis of AI development strategies in ten different countries and regions, a systematic table was formed that reflects the key aspects of each strategy.

This approach allows for the assessment and comparison of the presence of AI development strategies, special legal regulation (both rigid and flexible approaches), the definition of specific goals at the national level, the definition of the concept of AI, as well as the presence of high-level and practical guidelines from principles to AI implementation.

Therefore, to analyze this issue, experts in jurisprudence have proposed the structure in Table 2.

Table 2
Structure of the results

№	Parameter	Description
1	Country	The name of the country or region for which the AI strategy was analyzed is indicated
2	AI Development Strategy	Reflects whether a country or region has an officially formulated AI development strategy
3	Legally Binding Regulation	Indicates the presence of legally binding regulations aimed at regulating AI
4	Non-Binding Guidelines	Shows whether there are non-mandatory guidelines or recommendations for regulating AI
5	Specific Targets	Indicates whether there is a focus on specific national goals in the context of AI
6	AI Definition	Shows whether there is a clear definition of AI in the documents
7	High-Level Guidance	Indicates the presence of high-level guidelines from international organizations
8	Practical Guidance	Shows whether there are practical guidelines for the implementation of AI principles

Based on the proposed approach, Table 3 with the AI results for 10 documents was formed, and simultaneously, 5 experts in jurisprudence were also asked to form a corresponding table (Table 4).

To assess the effectiveness of AI in analyzing AI development strategies in different countries, a comparison was made between the results of data processed by AI (see Table 3) and the results of expert evaluations (Table 4).

In the study of effectiveness, the method of direct comparison of results obtained by AI with expert ratings across a range of key parameters was applied.

The accuracy of the AI was determined based on the percentage of agreement between the AI-generated data and the verified expert data.

The overall effectiveness of the AI analysis was measured as the average accuracy across all categories, which allowed for an assessment of its ability to adequately understand and interpret strategic documents in the field of AI development.

Table 3
AI processing results

Country	AI Development Strategy	Legally Binding Regulation	Non-Binding Guidelines	Specific Targets	AI Definition	High-Level Guidance	Practical Guidance
EU	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Brazil	Assumed Yes	Assumed No	Assumed Yes	Yes	Assumed Yes	Assumed Yes	Assumed Yes
Finland	Yes	Not specified	Yes	Yes	Not specified	Yes	Yes
India	Yes	Not specified	Yes	Yes	Not specified	Yes	Yes
Japan	Yes	Yes	Yes	Yes	Yes	Yes	Yes
USA	Yes	Not specified	Yes	Yes	Not specified	Yes	Yes
China	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Ukraine	Yes	Not specified	Not specified	Not specified	Not specified		Not specified
Quebec	Yes	Not specified	Yes	Not specified	Not specified	Yes	Yes
Columbia	Yes	Not specified	Not specified	Yes	Not specified	Yes	Not specified

The following criteria were defined within the method for evaluation:

- A full match between the results obtained by AI and expert data was scored as 1 point, indicating identical information and high accuracy of AI analysis.
- A partial match, where AI used the "Assumed" marker to assess parameters that were acknowledged as correct by experts, was assigned a score of 0.9 points. This reflects a high level of consistency in results, albeit with some degree of uncertainty in interpretation.
- A complete mismatch, where the information provided by AI did not correspond to expert assessments, received 0 points, indicating a complete divergence in inferences.

This differentiated evaluation system provides a deeper understanding of AI's effectiveness in analyzing and interpreting complex strategic documents.

Particular attention is paid to AI's ability to make adequate inferences in conditions of uncertainty, which is a key aspect in assessing its potential as an analytical tool in a wide range of applications.

These results indicate a significant potential for AI in analytical research aimed at developing and improving strategies in the field of AI, and they also define directions for further enhancement of data processing and analysis algorithms.

Table 4
The results of the work of 5 experts

Country	AI Development Strategy	Legally Binding Regulation	Non-Binding Guidelines	Specific Targets	AI Definition	High-Level Guidance	Practical Guidance
EU	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Brazil	Yes	Not specified	Assumed Yes	Yes	Assumed Yes	Yes	Assumed Yes
Finland	Yes	Not specified	Yes	Yes	Not specified	Yes	Yes
India	Yes	Not specified	Yes	Yes	Not specified	Yes	Yes
Japan	Yes	Yes	Yes	Yes	Yes	Yes	Yes
USA	Yes	Yes	Yes	Yes	Not specified	Yes	Yes
China	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Ukraine	Yes	Not specified	Assumed Yes	Not specified	Yes	Yes	Yes
Canada	Yes	Not specified	Yes	Not specified	Not specified	Yes	Yes
Columbia	Yes	Not specified	Not specified	Yes	Not specified	Yes	Not specified

The analysis of Table 5, which demonstrates a comparison of results obtained by AI with expert evaluations, revealed a high overall accuracy of AI in identifying AI development strategies by countries and categories. The average weighted accuracy across categories is 0.91, which indicates the effectiveness of AI in understanding and analyzing complex data. AI showed the highest accuracy in the "Specific Targets" category with a score of 1.0 and "AI Development Strategy" with an overall accuracy of 0.99. However, in categories such as "Legally Binding Regulation", "Non-Binding Guidelines", "AI Definition", "High-Level Guidance", and "Practical Guidance", a somewhat lower accuracy is observed (ranging from 0.89 to 0.8), which may suggest the need for further improvement of AI algorithms for better identification and interpretation of these aspects.

The highest accuracy by country is observed for the EU, Finland, India, Japan, China, and Quebec, where AI achieved full correspondence (1.0) with expert evaluations. However, in the case of Ukraine, the accuracy is only 0.43, which indicates significant discrepancies between AI and expert evaluations in certain categories. This could indicate a need for further refinement of AI algorithms to ensure more accurate analysis in cases with complex information or insufficient data.

Thus, the implementation of the proposed approach confirms the high efficiency of using AI in the analysis of AI development strategies in various countries and regions, as reflected in Table 4. The overall AI analysis accuracy of 0.91 demonstrates its high potential as an analytical tool for assessing complex strategic documents. Particularly important is the fact that AI shows high accuracy not only in defining the general goals of development strategies but also in understanding the legal aspects of AI regulation and practical guidelines for their implementation. Despite some challenges associated with the analysis of specific categories, such as "Legally Binding Regulation" and "Practical Guidance," where accuracy was lower, the results underline the significant potential for further refinement of AI algorithms to improve analysis accuracy.

Table 5
Comparison of the obtained results

Country	AI Development Strategy	Legally Binding Regulation	Non-Binding Guidelines	Specific Targets	AI Definition	High-Level Guidance	Practicl Guidance	***Accuracy by country***
EU	1	1	1	1	1	1	1	***1***
Brazil	0.9	0.9	1	1	1	0.9	1	***0.96***
Finland	1	1	1	1	1	1	1	***1***
India	1	1	1	1	1	1	1	***1***
Japan	1	1	1	1	1	1	1	***1***
USA	1	0	1	1	1	1	1	***0.86***
China	1	1	1	1	1	1	1	***1***
Ukraine	1	1	0	1	0	0	0	***0.43***
Quebec	1	1	1	1	1	1	1	***1***
Columbia	1	1	1	1	1	1	0	***0.86***
Accuracy by category	***0.99***	***0.89***	***0.9***	***1***	***0.9***	***0.89***	***0.8***	***0.91***

5. Conclusions

The results of the current research demonstrate the significant potential of applying AI in the process of analyzing legal documents, confirming the effectiveness of the proposed cyclical approach. The overall data processing accuracy using AI was 0.91, indicating high reliability and precision of the obtained results. Particularly impressive results were achieved in the categories "Specific Targets" and "AI Development Strategy," where the accuracy was 1.0 and 0.99, respectively, evidencing AI's ability to precisely identify key elements of AI development strategies.

Based on the comparison conducted with expert evaluations (where the overall accuracy of the AI analysis was 0.91), we confirmed the high efficiency of AI in analyzing AI development strategies at the international level. The lowest accuracy by countries was recorded for Ukraine (0.43), which highlights the importance of adapting analysis approaches to the specifics of each particular legal system, in this case, the Ukrainian language.

These conclusions demonstrate the significant potential for applying AI in analytical research in the legal field, as well as indicating directions for further improvement of legal document processing and analysis technologies. Ensuring high accuracy in analysis is key to

enhancing the efficiency of legal processes and developing effective strategies for regulating and using AI at both national and international levels.

Based on the conducted analysis, future scientific research should focus on developing improved algorithms to increase the accuracy of classification and analysis of legal documents, particularly in categories like "Legally Binding Regulation" and "Practical Guidance," where accuracy was found to be lower. Another important direction is the development of methods for the effective processing and analysis of Ukrainian legal texts, which contain a large amount of unstructured information, in order to ensure a deeper understanding of the context.

References

[1] B. Gates The road ahead reaches a turning point in 2024. December 19 2023. URL: https://www.gatesnotes.com/The-Year-Ahead-2024 (date of access: 21.01.2024).

[2] Ethics of Artificial Intelligence. UNESCO. URL: https://www.unesco.org/en/artificial intelligence/recommendation-ethics (date of access: 21.01.2024).

[3] C. Smith, T. Wuang, G. Yang, History of Artificial Intelligence, available at URL: https://courses.cs.washington.edu/courses/csep590/06au/projects/history-ai.pdf (date of access: 06.01.2024).

[4] Ogwuche, Perpetua, Artificial Intelligence: The Legal Implications of Intellectual Property Rights for AI-generated Inventions (October 16, 2022). URL: https://ssrn.com/abstract=4589323 or http://dx.doi.org/10.2139/ssrn.4589323 (date of access: 12.12.2023).

[5] Investopedia – What is Artificial Intelligence (AI)? URL: https://www.investopedia.com/terms/a/artificial intelligence-ai.asp (date of access: 12.12.2023

[6] I. J. Lloyd. Information Technology Law. 9th Edition. Oxford University Press, 2020, 482p.

[7] F.S. De Sio, G. Mecacci, Four responsibility gaps with artificial intelligence: Why they matter and how to address them. Philosophy & Technology, 34(4), (2021). 1057-1084. doi: 10.1007/s13347-021-00450-x.

[8] H. Zhong, C. Xiao, C. Tu, T. Z., Z. Liu, Maosong Sun How Does NLP Benefit Legal System: A Summary of Legal Artificial Intelligence. (2020). URL : https://arxiv.org/pdf/2004.12158.pdf.

[9] T. C. Bueno, C. V. Wangenheim, E. D. S, Mattos, H. C. Hoeschl, R. M. Barcia, Juris Consulto: retrieval in jurisprudence a text bases using juridical terminology. ACM. (1999). https://dx.doi.org/10.1145/323706.323789.

[10] R. S. Rana, S. Singh, M. Aggarwal, M. Badoni, Unveiling the Future: Exploring AI Applications in the Indian Judicial System. IEEE. (2023). https://dx.doi.org/10.1109/ISED59382.2023.10444600.

[11] W. Wyporek, Sztuczna inteligencja–wybór orzecznictwa. (2021). https://dx.doi.org/10.13166/wsge/hr-pl/mbtu5710.

[12] Chucha, S. Artificial intelligence in justice: legal and psychological aspects of law enforcement. (2023). https://dx.doi.org/10.52468/2542-1514.2023.7(2).116-124.

[13] Al Qatawneh, I. S., Moussa, A., Haswa, M., Jaffal, Z., &Barafi, J. Artificial Intelligence Crimes. AJIS. (2023). https://dx.doi.org/10.36941/ajis-2023-0012.

[14] S. Iqbal, S.-U. Hassan, N. Aljohani, S. Alelyani, R. Nawaz, L. Bornmann, A decade of in-text citation analysis based on natural language processing and machine learning techniques: an overview of empirical studies. Scientometrics. (2020).

[15] Dhala, R. R., Kumar, A.V.S.P., & Panda, S. A Comparative Study on Keyword Extraction and Generation of Synonyms in Natural Language Processing. IEEE Access. (2023).

[16] M. Khanbhai, P. Anyadi, J. Symons, K. Flott, A. Darzi, E. Mayer, Applying natural language processing and machine learning technique stop a itient experience feedback: a systematic review. BMJ Health & Care Informatics. (2021). https://dx.doi.org/10.1136/bmjhci-2020-100262

[17] Yu, H., Xiong, F., &Chen, Z. Text Classification Based on Natural Language Processing and Machine Learning in Multi Label Corpus. ACM Transactions. (2023). https://dx.doi.org/10.1145/3617831

[18] B. ShireeshaK, P. Thejas, D. S. Kumar, Document Analysis using Deep Learning Techniques. IEEE International Conference on Electrical, Computer, and Energy Technologies (ICECET). (2023). https://dx.doi.org/10.1109/ICESC57686.2023.10193742.

[19] R. Gramyak, H. Lipyanina-Goncharenko, A. Sachenko, T. Lendyuk, D. Zahorodnia, Intelligent Method of a Competitive Product Choosing based on the Emotional Feedbacks Coloring. CEUR-WS 2853 (2021) 246-257.

[20] K. Lipianina-Honcharenko, C. Wolff, A. Sachenko, O. Desyatnyuk, S. Sachenko, I. Kit, Intelligent information system for product promotion in internet market. Applied sciences, 13(17), (2023). 9585. doi:10.3390/app13179585.

[21] I. Perova, Y. Bodyanskiy, Fast medical diagnostics using autoassociative neuro-fuzzy memory. International Journal of Computing, 16(1), (2017). 34-40. https://doi.org/10.47839/ijc.16.1.869.

[22] H. Lipyanina, S. Sachenko, T. Lendyuk, A. Sachenko, Targeting Model of HEI Video Marketing based on Classification Tree. In ICTERI Workshop (2020). 487-498.

[23] K. Lipianina-Honcharenko, R. Savchyshyn, A. Sachenko, A. Chaban, I. Kit, T. Lendiuk, Concept of the intelligent guide with AR support. International Journal Of Computing, (2022). 271–277. doi:10.47839/ijc.21.2.2596.

[24] H. Bhatt, R. Bahuguna, R. Singh, A. Gehlot, S. V. Akram, N. Priyadarshi, B. Twala, Artificial Intelligence and robotics led technological tremors: A seismic shift towards digitizing the legal ecosystem. Applied Sciences, 12(22), (2022). 11687.

[25] G. M. Csányi, R. Vági, D. Nagy, I. Üveges, J. P. Vadász, A. Megyeri, T. Orosz, Building a production-ready multi-label classifier for legal documents with digital-twin-distiller. Applied Sciences, 12(3), (2022). 1470.

[26] B. Abimbola, E. de La Cal Marin, Q. Tan, Enhancing Legal Sentiment Analysis: A Convolutional Neural Network–Long Short-Term Memory Document-Level Model. Machine Learning and Knowledge Extraction, 6(2), (2024). 877-897.

[27] J. L. Pach, P. Bilski, A robust binarization and text line detection in historical handwritten documents analysis. International Journal of Computing, 15(3), (2016). 154-161. https://doi.org/10.47839/ijc.15.3.848.

[28] E. M.Cherrat, R. Alaoui, H. Bouzahir, Score fusion of finger vein and face for human recognition based on convolutional neural network model, International Journal of Computing, 19(1), (2020) 11-19. https://doi.org/10.47839/ijc.19.1.1688.

Evaluation of the effectiveness of machine learning methods for detecting disinformation in Ukrainian text data

Khrystyna Lipianina-Honcharenko[1], Mariana Soia[1], Khrystyna Yurkiv[1], Andrii Ivasechko[1].
[1] *West Ukrainian National University, Lvivska str., 11, Ternopil, 46000, Ukraine*

Abstract

In today's world, where information spreads with unprecedented speed, disinformation poses a serious challenge to public trust and information security. The full-scale invasion of Ukraine by Russia in 2022 activated the use of disinformation as a tool of hybrid warfare, highlighting the need for effective methods of identification and control. This article focuses on evaluating the effectiveness of various machine learning methods for detecting disinformation in Ukrainian text data, using a dataset that includes news headlines collected during the conflict. The study encompasses the analysis of logistic regression, support vector machines (SVM), random forest, gradient boosting, KNN, decision trees, XGBoost, and AdaBoost. Model evaluation was performed using standard metrics: precision, recall, F1-score, overall accuracy, and confusion matrix. The results indicate significant potential for using machine learning in the fight against disinformation, particularly the random forest model demonstrated the highest effectiveness. The study emphasizes the importance of adapting and optimizing classifiers for the specific task of disinformation analysis, paving the way for further research in this field.

Keywords

Disinformation, machine learning, classification, text data.

1. Introduction

In the modern information space, a vast amount of data is generated daily, a significant portion of which is news content. In the context of increasing globalization and accessibility of information technologies, information spreads rapidly through the network, making it a powerful tool for influencing public opinion. However, this also paves the way for the mass dissemination of disinformation, which can have significant consequences for society, politics, and international relations. Navigating this flow of information and distinguishing reliable data from false has become an increasingly important task.

The war that began with Russia's full-scale invasion of Ukraine in February 2022 is a striking example of the use of disinformation as a weapon in hybrid warfare. This has created a need for the development of effective tools for analyzing and classifying informational content, with the goal of identifying and counteracting disinformation.

In this context, machine learning and natural language processing methods play a key role in the detection and analysis of fake news. The application of these technologies allows for the automation of the disinformation detection process, providing fast and efficient processing of large volumes of data. At the same time, the development of effective machine learning models for information classification requires a deep understanding of data specifics, preprocessing methods, and model optimization.

This article is devoted to the analysis of a dataset containing news headlines collected during the Russo-Ukrainian conflict, aimed at identifying disinformation. It considers the application of various machine learning methods, including logistic regression, support vector machines (SVM), random forest, gradient boosting, KNN, decision trees, XGBoost, and AdaBoost for the classification of text data. The concluding section is dedicated to evaluating the effectiveness of these models using standard evaluation metrics, including precision, recall, F1-score, overall

Proceedings Acronym: Proceedings Name, Month XX–XX, YYYY, City, Country
xrustya.com@gmail.com (K. Lipianina-Honcharenko); mariankasoa@gmail.com (M. Soia); kh.yurkiv@wunu.edu.ua (K. Yurkiv); Andrewivasechko@gmail.com (A. Ivasechko).
0000-0002-2441-6292 (K. Lipianina-Honcharenko); 0009-0001-3719-6460 (M.Soia); 0009-0007-4917-3251 (K. Yurkiv); 0009-0002-8623-7828 (A. Ivasechko).

accuracy, and the confusion matrix, which allows determining the most effective methods for combating disinformation in the context of information warfare.

2. Related Work

In contemporary research in the field of information security and media content analysis, the importance of detecting and analyzing disinformation, especially anti-vaccine content on social media platforms such as Twitter, has gained particular relevance. In [1], language-neutral models are developed for detecting such content on a large scale, using multifaceted representations of messages in networks. Meanwhile, [2] focuses on the challenges associated with pre-training graph neural networks for context-oriented detection of fake news, pointing out strategic and resource constraints. In [3], a comparative analysis of supervised and unsupervised machine learning algorithms for detecting fake news is conducted, demonstrating their performance, efficiency, and robustness.

Research covering the analysis of disinformation and public opinion on the Russo-Ukrainian War employs a variety of methodologies, including sentiment analysis, creation and analysis of datasets, and studies of the impact of war on language choice. One study models and clusters sentiment trends of different countries regarding the war [4], while another offers a detailed dataset of tweets related to the crisis [5]. The discourse on Twitter about the war is also analyzed, with a particular focus on language and the geographical origin of tweets [6]. Another study focuses on the challenges of labeling sensitive content and its psychological impact on annotators [7]. It is also examined how the war affects the language choice of Ukrainians on Twitter, analyzing changes in language preferences over time [8]. A separate study provides insights into the activity on subreddits related to the conflict, analyzing post volumes, comments, and the level of engagement [9]. Research [10] demonstrates that the application of the BERT model for fake news detection achieves an accuracy of 79.88% and an area under the ROC curve of 0.87, highlighting its potential in combating disinformation on social networks.

As confirmed by the aforementioned analysis, research in the field of disinformation detection, particularly fake news, is a significant direction in the context of information security and media content analysis. The need for the development and application of advanced technological solutions for effective fake news detection is particularly compelling. However, there is a noticeable lack of research focused on the analysis of disinformation in the Ukrainian language, which poses a challenge to the scientific community to expand the linguistic spectrum of research in this field. Considering this, the aim of this study is to develop intelligent methods for detecting disinformation, specifically fake news, with an emphasis on Ukrainian-language content. This will enhance the level of information security and ensure the integrity of the news space in the conditions of the modern information society.

3. Research methodology

3.1 Dataset Description

For data collection and processing, the dataset (Ukrainian language) [11] was used, which contains approximately 10,700 news headlines about the Russo-Ukrainian War, collected from February 24 to December 11, 2022, covering the period from the beginning of the full-scale invasion.

The dataset for the analysis of disinformation in the Ukrainian media space consists of two main files: "data_set_4.csv" (Table 1) and "news_data.csv" (Table 2), each containing news headlines classified as true ("True") or false ("False"). The "data_set_4.csv" file records 8,237 true and 2,498 false news items, while "news_data.csv" contains a significantly larger number of true news items — 48,006, compared to 2,024 false, reflecting a wide range of informational content collected for the study of the dissemination of disinformation during the Russo-Ukrainian War.

Both datasets (see Table 1,2) are used for the analysis of disinformation in the Ukrainian media space and include data that were collected from official and unofficial sources with the purpose of studying the spread of fake news in the context of the Russo-Ukrainian War.

Table 1
Structure of data_set_4.csv

Field	Description	Data Type
Text	News Headline	Text
Label	Label that marks the news as true ("True") or false ("False")	Boolean
Link	Link to the news source (available only in this file)	Text

Table 2
Structure of news_data.csv

Field	Description	Data Type
Text	News Headline	Text
Label	Classification of the news as true ("True") or false ("False")	Boolean

The graphs (Fig.1) of label distribution show that in both datasets, the number of true messages predominates over the false ones. For example, in the first dataset, the ratio of true messages to false is high, which indicates a focus on reliable information.

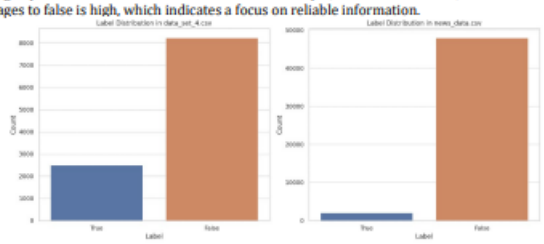


Figure 1: Label Distribution

The analysis of the most frequently used words (Fig.2) in the first dataset revealed words that appear hundreds of times. This allows identifying key topics of discussions or news, for example, the word "Ukraine" may occur most frequently, emphasizing the geographical or political focus of the collected data.

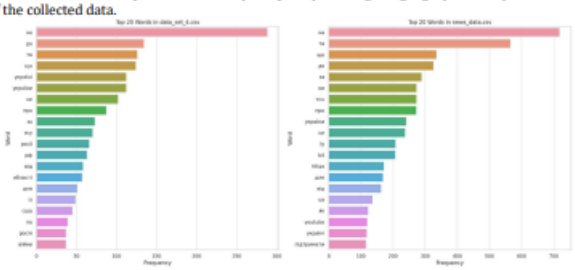


Figure 2: Top 20 Words

Most texts (Fig.3) in the first dataset have a length of 100 to 500 characters. Such information helps to understand the average volume of messages, which may indicate a prevalence of short news or overviews.

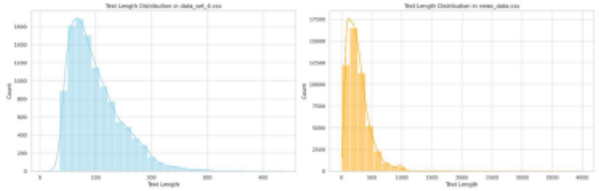


Figure 3: Distribution of Text Lengths

Analysis of the presented datasets covering headlines of news about the Russian-Ukrainian war demonstrates a significant advantage of credible information compared to false, emphasizing the focus on quality content. Considering that the dataset "news_data.csv" contains substantially more records compared to "data_set_4.csv", it is logical to use the former for training machine learning models as it provides a broader range of information and a greater number of examples for training. Meanwhile, the smaller dataset "data_set_4.csv" can serve as an excellent set for testing and evaluating the effectiveness of models on a smaller but specific data sample. This will allow assessing the model's ability to generalize learning on new, previously unknown data, emphasizing its practical value in real-world disinformation analysis conditions.

3.2 Description of used classifiers

In modern data analysis, especially in the context of detecting misinformation, the use of machine learning algorithms for classifying textual data becomes a key tool for developers and analysts. The diversity of classification methods [12], such as logistic regression, support vector machines (SVM), random forest, gradient boosting, k-nearest neighbors (KNN), decision tree, XGBoost, and AdaBoost, provides a wide range of approaches for data analysis and classification. Each of these algorithms has its unique advantages and limitations, making them more or less suitable for specific types of data and analytical tasks. The main goal is to select the optimal classifier that best fits the specificity of the task and the data we are working with.

Logistic regression [13] is a classification method used to predict the probability of two possible outcomes based on one or more independent variables. It transforms the linear combination of input data into probability using the logistic function. The advantages of logistic regression include simplicity in interpreting results, but it may be limited when analyzing complex relationships between variables and requires an assumption of linearity in relationships. Mathematically, logistic regression models the probability $P(Y=1)$ as a function of X, where Y is the dependent variable, and X is the set of independent variables. The probability is described by the equation:

$$P(Y=1) = \frac{1}{1+e^{-(\beta_0+\beta_1 X_1+\cdots+\beta_k X_k)}}, \quad (1)$$

where e is the base of the natural logarithm, β_0 is the constant term (intercept), and $\beta_1,\ldots,\beta_k$ are the coefficients of the independent variables. This equation allows us to estimate the probability that an observation belongs to class 1, depending on the values of the independent variables.

Support Vector Machine (SVM) algorithm [14] finds a hyperplane in a multidimensional space that best separates different data classes, maximizing the distance between the closest data points (support vectors) of different classes. The advantages of SVM include high accuracy in classification tasks, especially on relatively small datasets, and flexibility through the use of various kernel functions. However, its drawbacks include high computational resource requirements for large datasets and complexity in interpreting the model. Mathematically, SVM seeks to solve the optimization problem: minimize $\frac{1}{2} \| w \|^2$ subject to the constraints that

$$y_i (w \cdot x_i + b) \geq 1 \text{ to all } I, \quad (2)$$

where w is the weight vector of the hyperplane, b is the bias, x_i are the feature vectors, and y_i are the class labels.

The Random Forest algorithm [15] creates an ensemble of decision trees, training each tree on randomly selected subsets of the training dataset and features, which ensures high accuracy and model universality. The advantages of Random Forest include its ability to efficiently handle large datasets with high feature dimensionality and its lower tendency to overfit compared to individual decision trees. However, its drawbacks include relatively high computational resource requirements and the complexity of interpreting the model due to the large number of trees. The mathematical interpretation of Random Forest is based on the principle of "wisdom of the crowd," where the final model decision is determined by voting among the trees for classification tasks or averaging the outputs for regression tasks. More precisely, for classification:

$$Y = mode\{y_1, y_2, \ldots, y_n\}, \quad (3)$$

and for regression:

$$Y = \frac{1}{n}\sum_{i=1}^{n} y_i, \quad (4)$$

where y_i is the prediction of each tree, and Y is the final prediction of the ensemble.

Gradient Boosting [16] is an ensemble machine learning method that improves predictions by sequentially training weak models, typically decision trees, to minimize a loss function. It is characterized by high prediction accuracy and flexibility in parameter tuning, but it can be prone to overfitting if not properly configured and requires more time and computational resources for training compared to other algorithms. Mathematically, Gradient Boosting performs optimization by adaptively reducing the difference between actual and predicted values using gradient descent, where the model update in the m-th iteration is defined as:

$$F_m(x) = F_{m-1}(x) + \alpha_m h_m(x), \quad (5)$$

where $F_{m-1}(x)$ is the prediction at the previous step, $h_m(x)$ is the weak classifier, and α_m is the learning rate.

The k-Nearest Neighbors (KNN) algorithm [17] classifies objects based on the nearest training examples in the feature space, where "k" indicates the number of neighbors considered to determine the class of a new object. The advantages of KNN include ease of implementation and its ability to effectively work with multi-class datasets. However, it requires significant computational resources to store training data and determine neighbors in large datasets, and it is sensitive to irrelevant features and data scaling. Mathematically, the classification of an object x in the KNN algorithm is determined by the majority vote of its neighbors, where each neighbor is weighted according to the inverse distance to x, typically using Euclidean distance:

$$d(x, x_i) = \sqrt{\sum_{j=1}^{m}(x_j - x_{ij})^2}, \quad (6)$$

where x is the point for classification, x_i is a point from the training dataset, and j varies from 1 to m, the number of features. The class of x is determined based on the most frequent class among the k nearest neighbors.

The Decision Tree algorithm [18] builds a predictive model in the form of a tree-like structure by partitioning the dataset into smaller subsets while simultaneously developing the associated decision tree. The advantages of decision trees include ease of interpretation, the ability to handle both numerical and categorical data, and no requirement for data normalization. However, decision trees are prone to overfitting, especially with deep trees, and can be unstable, meaning small changes in data can result in significantly different decision trees. Mathematically, decision trees use the concept of information gain or reduction in uncertainty (entropy) to select the attribute that best splits the dataset into subsets according to the target variable. The information gain for attribute A is calculated as:

$$IG(T, A) = Entropy(T) - \sum_{v \in Values(A)} \frac{|T_v|}{|T|} Entropy(T_v), \quad (7)$$

where T is the training set, $Values(A)$ is the set of all possible values of attribute A, T_v is the subset of T for which attribute A has the value v, and $Entropy(T)$ is the entropy of the training set T.

XGBoost (eXtreme Gradient Boosting) [19] is a highly efficient implementation of the gradient boosting algorithm, which optimizes both linear models and decision trees. The algorithm stands out for its high execution speed, ability to efficiently scale to large datasets, and built-in support for regularization, helping to mitigate overfitting. However, XGBoost can be challenging to tune due to the large number of hyperparameters and requires more time for training compared to simpler models. Mathematically, XGBoost minimizes losses using gradient descent, where the objective function includes both the loss function L and a penalty for model complexity Ω, adding regularization:

$$Obj = \sum_i L(y_i, \hat{y}_i) + \sum_k \Omega(f_k), \quad (8)$$

where y_i are the true labels, $\hat{y}_i$ are the predicted labels, f_k are the functions representing individual trees, and $\Omega(f_k)$ includes terms for regularization, such as the number of leaves and the sum of squares of node weights, thereby reducing the risk of overfitting.

AdaBoost (Adaptive Boosting) [20] is a machine learning algorithm that combines multiple weak classifiers to create a strong classifier, using an iterative approach to correct the errors of previous classifiers by assigning greater weight to observations that are harder to classify. The advantage of AdaBoost is its ability to improve prediction accuracy, ease of implementation, and automatic correction of underperforming classifiers. However, it can be prone to overfitting in the presence of outliers or highly noisy data and requires careful tuning of the number of iterations. Mathematically, AdaBoost adapts the weights of training observations, w_i, by increasing the weights of incorrectly classified observations. At each iteration step t, a classifier h_t is selected to minimize the weighted sum of errors. The final classifier is determined as a weighted sum of these classifiers.

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x)), \quad (9)$$

where α_t is the weight assigned to classifier h_t, which depends on its accuracy.

Conclusions drawn from the description of the used classifiers underscore the importance of adapting the model to the specificity of the dataset and the analytical task. The effectiveness of each method depends on the size and quality of the data, the complexity of relationships in the dataset, as well as specific requirements for accuracy and interpretation of results. In the context of disinformation analysis, the choice between the simplicity of interpreting logistic regression and the high accuracy but complexity of tuning XGBoost or SVM may determine the success or failure in identifying fake news. Thus, careful selection and tuning of classifiers are critically important for developing effective tools to combat disinformation in the context of the Russian-Ukrainian war.

3.3 Evaluation metrics

For evaluating the effectiveness of models in classification tasks, key metrics such as precision, recall, F1-score, accuracy, and confusion matrix are utilized [21]. These metrics allow for a deeper analysis of the model's performance, identifying potential weaknesses, and optimizing the model for better results.

Precision is defined as the ratio of the number of true positive results to the total number of results classified as positive by the model. Mathematically, it is represented as:

$$P = \frac{TP}{TP+FP}, \quad (10)$$

where TP — true positives, and FP — false positives.

Recall measures the model's ability to identify all actual positive cases in the dataset. It is defined as the ratio of the number of correctly identified positive results to the sum of correctly identified positive results and instances that are actually positive but were missed by the model. The formula for calculation is:

$$R = \frac{TP}{TP+FN}, \quad (11)$$

where FN — false negatives.

The F1 Score is the harmonic mean between precision and recall, providing a balance between these two metrics. This is particularly useful in situations where class imbalances may cause biases in one metric over the other. F1 is defined as:

$$F1 = 2 \cdot \frac{P \cdot R}{P+R}. \quad (12)$$

Accuracy measures the percentage of cases correctly classified by the model and is defined by the formula:

$$A = \frac{TP+TN}{TP+TN+FP+F}, \quad (13)$$

where TN — true negatives.

The Confusion Matrix provides a visualization of classification results by representing the counts of TP, TN, FP, and FN in the form of a matrix. This allows us not only to determine the accuracy of the model but also to understand the types of errors made by the model.

3.4 Model training

The training procedure (Figure 4) on the training dataset is a fundamental step in the development of effective machine learning algorithms for text data classification. In this context, the use of datasets "news_data.csv" and "data_set_4.csv" for training and testing models, respectively, provides a valuable foundation for misinformation analysis. The initialization of the procedure begins with the import and preprocessing of data, including the removal of records without textual content, ensuring data cleanliness for subsequent processing stages.

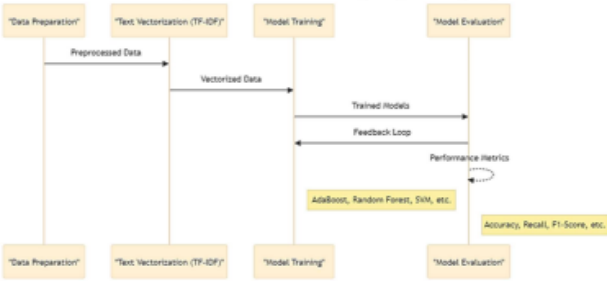


Figure 4: Model Training Process

Text vectorization using TF-IDF (Term Frequency-Inverse Document Frequency) is a key step in data preparation, as it transforms textual data into a numerical format, making them suitable for processing by machine learning models. This method considers not only the frequency of words in the text but also their uniqueness through the inverse frequency of documents, allowing the model to better identify important features in the text.

After preparing the vectorized training and testing data, the next stage involves training different models on the training dataset. This process requires the application of machine learning algorithms to the training dataset to form a model capable of effectively classifying text based on learned features. Each model adjusts its parameters to minimize errors on the training set while simultaneously ensuring the ability to generalize to new data.

Evaluation of the effectiveness of each trained model on the test dataset is crucial for determining its suitability and efficiency in classification tasks. The use of evaluation metrics such as accuracy, recall, F1-score, and overall accuracy allows for deep analysis and comparison of model performance. The evaluation results can be visualized for better understanding of model effectiveness, and the best-performing model can be selected for further use or saved for future analysis.

Thus, the model training procedure on the training dataset using pre-processed and vectorized text data is fundamental for developing reliable machine learning tools capable of effectively classifying and analyzing text for misinformation.

4. Research results

Further, we will discuss the implementation and evaluation of the effectiveness of various machine learning models in the task of classifying misinformation data in the Ukrainian media space, contextualized in the context of Russia's full-scale intervention. By analyzing a wide range of algorithms, we aim to identify the most effective methods for accurate detection and differentiation of misinformation incidents. This analysis is based on a comprehensive comparison of overall classification accuracy as well as specific metrics such as precision, recall, and F1-score for each class. The implementation was done in the Python programming language utilizing libraries for text analysis.

The logistic regression model showed (Figure 5) an overall classification accuracy of 86.4% on the test sample of 10,735 examples, demonstrating high effectiveness in identifying true values with an F1-score of 0.92 for the "True" class. However, the model was less accurate in identifying the "False" class, with a prediction accuracy of 0.96 and a low recall score of 0.44, indicating a significant number of false negatives, as reflected in the confusion matrix with 1,407 misclassified examples.

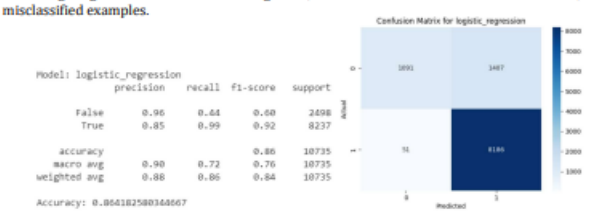


Figure 5: LogisticRegression Results

The SVM model demonstrated (Figure 6) high overall classification accuracy of 93.6% for the test sample, indicating its effectiveness in distinguishing between the "True" and "False" classes. Specific accuracy and recall metrics for the "False" class were 0.97 and 0.75, respectively, demonstrating the model's ability to identify negative cases well, albeit with some errors. At the same time, high metrics for the "True" class with precision of 0.93 and recall of 0.99 indicate minimal false negative classifications, confirmed by a low number of errors in the confusion matrix (618 for "False" and 68 for "True").

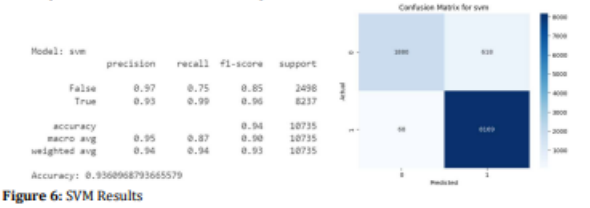


Figure 6: SVM Results

The Random Forest model demonstrated (Figure 7) high accuracy in classification with an overall accuracy of 95.3% on the test dataset, indicating the model's high effectiveness in recognizing both classes. For the "False" class, the model showed high accuracy (0.98) and a relatively high recall of 0.81, indicating the model's ability to effectively identify negative cases with a moderate number of errors. Conversely, extremely high accuracy (0.95) and almost perfect recall (1.00) for the "True" class highlight the minimal number of false negative results, corroborated by the low number of errors in the confusion matrix.

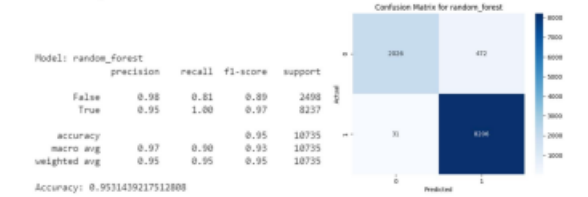


Figure 7: RandomForestClassifier Results

The Gradient Boosting model achieved (Figure 8) an overall classification accuracy of 87.3% on the test dataset, indicating the model's good ability to distinguish between the "True" and "False" classes. The model's precision for the "False" class was high (0.94), but the recall was only 0.48, indicating a significant number of type II errors, where negative cases are often misclassified as positive. Conversely, for the "True" class, the model showed impressive classification ability with a precision of 0.86 and a recall of 0.99, demonstrating its high effectiveness in identifying true positive cases with minimal errors.

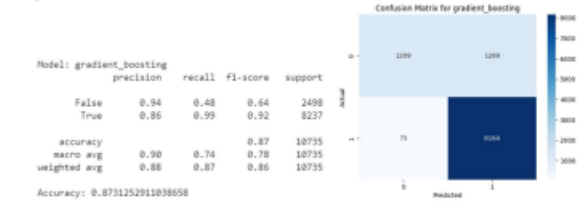


Figure 8: Gradient Boosting Results

The KNN (K-Nearest Neighbors) model demonstrated (Figure 9) an overall classification accuracy of 87.6% on the test dataset, highlighting its ability to effectively classify data. Although the model showed high precision (0.91) for the "False" class, the recall was only 0.51, indicating difficulties in identifying all negative cases. Conversely, for the "True" class, the model exhibited excellent precision (0.87) and a high recall (0.99), indicating its ability to identify positive cases with high confidence, albeit with some false positive errors, as seen in the confusion matrix.

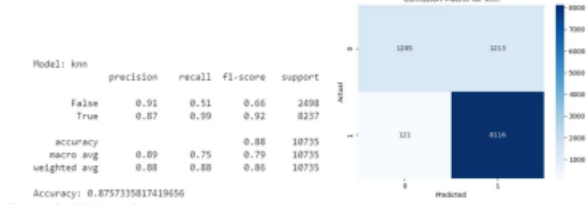


Figure 9: KNN Results

The Decision Tree model achieved (Figure 10) a high level of classification accuracy, 91.4%, on the test dataset, confirming its effectiveness in classifying data into "True" and "False" classes. For the "False" class, the model showed relatively high precision (0.81) and recall (0.83), indicating a balanced ability to identify negative cases with a relatively small number of errors. Meanwhile, for the "True" class, the model provided impressive precision (0.95) and recall (0.94), demonstrating its strong capabilities in identifying positive cases with minimal false negatives, as reflected in the confusion matrix.

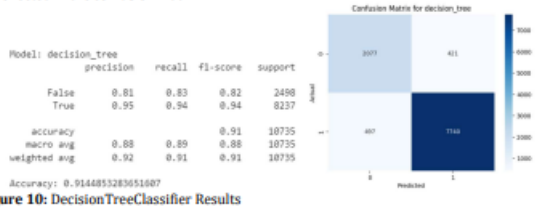


Figure 10: DecisionTreeClassifier Results

The XGBoost model demonstrated (Figure 11) an overall accuracy of 89.2% on the test dataset, indicating its ability to effectively classify the given dataset. The precision and recall metrics for the "False" class were 0.89 and 0.61, respectively, indicating the model's higher ability to correctly identify negative cases, albeit with some errors. Meanwhile, for the "True" class, the model showed high precision and recall (both 0.89 and 0.98), demonstrating excellent ability to accurately identify positive cases with minimal errors, as reflected in its high F1-score.

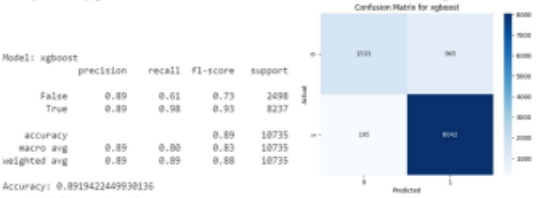


Figure 11: XGBoost Results

The AdaBoost model exhibited (Figure 12) an overall classification accuracy of 86.9% on the test dataset, indicating its effectiveness in recognizing data, albeit with some limitations. For the "False" class, the model achieved a precision of 0.88 with a recall of 0.50, indicating a relatively low ability to identify all negative cases. On the other hand, high precision (0.87) and recall (0.98) for the "True" class underscore the model's strong ability to detect positive cases, albeit with few errors, as reflected in the confusion matrix.

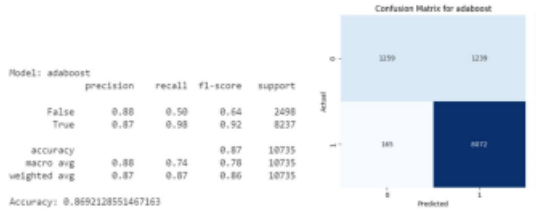


Figure 12: AdaBoostClassifier Results

Therefore, analyzing the results of applying various machine learning models to classify disinformation data in the Ukrainian media space after the onset of Russia's full-scale invasion, it can be noted that the Random Forest model proved to be the most effective with an accuracy of 95.3%, highlighting its ability to accurately detect and differentiate disinformation incidents. At the same time, models such as AdaBoost and logistic regression showed lower overall accuracy, which may indicate their limitations in identifying subtle cases of disinformation or more subjective aspects of information operations. This underscores the importance of choosing the appropriate model for the task of analyzing disinformation, where Random Forest may be more suitable for deep understanding and detecting complex patterns of disinformation in the context of information warfare.

5. Conclusion

The analysis of the effectiveness of various machine learning models applied to the task of classifying news headlines for misinformation in the Ukrainian media space demonstrates significant variations in accuracy, recall, and F1-score among the models. The considered algorithms, including logistic regression, SVM, random forest, gradient boosting, KNN, decision tree, XGBoost, and AdaBoost, showed different levels of performance in addressing the task.

The random forest model emerged as the most effective, achieving an overall accuracy of 95.3%, indicating its high capability to recognize and distinguish true and false messages. This is supported by high precision scores for both the "False" class (0.98) and the "True" class (0.95), as well as significant recall scores for both classes (0.81 for "False" and 1.00 for "True"), demonstrating its effectiveness in minimizing both false positives and false negatives.

These results underscore the importance of choosing the appropriate model for a specific misinformation analysis task. The model choice not only affects the overall classification accuracy but also the model's ability to minimize false positives or false negatives, which is crucial for developing effective tools to combat misinformation. Particularly, the random forest model, which demonstrated the best performance, can be recommended as the optimal choice for similar tasks, providing a high level of accuracy and the ability to effectively distinguish between true and false messages.

Further scientific research in the field of identification and analysis of misinformation in the Ukrainian media space requires deeper development and improvement of machine learning algorithms, with a particular focus on enhancing their ability to recognize subtle and complex forms of misinformation. The results of our study show that the random forest model, with an accuracy of 95.3%, proved to be the most effective, but there is potential for improvement, especially in accurately distinguishing between true and false messages. In the future, researchers may focus on developing hybrid models that combine the advantages of multiple algorithms, including deep learning and neural networks, to ensure greater adaptability and accuracy in different informational contexts. Additionally, an important direction will be the development of methods that allow models to better understand the semantic context and emotional tone of texts, which can significantly improve their ability to identify hidden

misinformation. Implementing such approaches will require not only technological innovations but also a deeper understanding of linguistic nuances and cultural-historical contexts on which misinformation is based.

References

[1] Ashford, J.R. (2024). Detecting Anti-vaccine Content on Twitter using Multiple Message-Based Network Representations. arXiv preprint arXiv:2402.18335.

[2] Donabauer, G., & Kruschwitz, U. (2024). Challenges in Pre-Training Graph Neural Networks for Context-Based Fake News Detection: An Evaluation of Current Strategies and Resource Limitations. arXiv preprint arXiv:2402.18179.

[3] Kaur, S., & Ranjan, S. (2024). Comparative Analysis of Supervised and Unsupervised Machine Learning Algorithms for Fake News Detection: Performance, Efficiency, and Robustness. ResearchGate.

[4] Vahdat-Nejad, H., Akbari, M. G., Salmani, F., Azizi, F., & Nili-Sani, H. R. (2023). Russia-Ukraine war: Modeling and Clustering the Sentiments Trends of Various Countries. arXiv preprint arXiv:2301.00604.

[5] Haq, E. U., Tyson, G., Lee, L. H., Braud, T., & Hui, P. (2022). Twitter dataset for 2022 russo-ukrainian crisis. arXiv preprint arXiv:2203.02955.

[6] Zia, H. B., Haq, E. U., Castro, I., Hui, P., & Tyson, G. (2023). An Analysis of Twitter Discourse on the War Between Russia and Ukraine. arXiv preprint arXiv:2306.11390.

[7] Daria, S. (2023). When a Language Question Is at Stake. A Revisited Approach to Label Sensitive Content. arXiv preprint arXiv:2311.10514.

[8] Racek, D., Davidson, B. I., Thurner, P. W., & Kauermann, G. (2023). The Politics of Language Choice: How the Russian-Ukrainian War Influences Ukrainians' Language Use on Twitter. arXiv preprint arXiv:2305.02770.

[9] Zhu, Y., Haq, E. U., Lee, L. H., Tyson, G., & Hui, P. (2022). A reddit dataset for the russo-ukrainian conflict in 2022. arXiv preprint arXiv:2206.05107.

[10] Padalko, H., Chomko, V., & Chumachenko, D. (2023). Misinformation Detection in Political News using BERT Model. Proceedings of the 3rd International Workshop of IT-professionals on Artificial Intelligence (ProfIT AI 2023) Waterloo, Canada, November 20-22, 2023. 117-127

[11] Ukrainian news. (n.d.-a). Kaggle: Your Machine Learning and Data Science Community. https://www.kaggle.com/datasets/zepopo/ukrainian-fake-and-true-news/data?select=news_data.csv

[12] Lipyanina, H., Maksymovych, V., Sachenko, A., Lendyuk, T., Fomenko, A., & Kit, I. (2020, August). Assessing the investment risk of virtual IT company based on machine learning. In International Conference on Data Stream Mining and Processing (pp. 167-187). Cham: Springer International Publishing.

[13] Lipyanina, H., Sachenko, S., Lendyuk, T., Brych, V., Yatskiv, V., & Osolinskiy, O. (2021, January). Method of detecting a fictitious company on the machine learning base. In International Conference on Computer Science, Engineering and Education Applications (pp. 138-146). Cham: Springer International Publishing.

[14] Kalogridis, I. (2024). Robust and adaptive functional logistic regression. Computational Statistics & Data Analysis, 192, 107905.

[15] Çakir, M., Yilmaz, M., Oral, M. A., Kazanci, H. Ö., & Oral, O. (2023). Accuracy assessment of RFerns, NB, SVM, and kNN machine learning classifiers in aquaculture. Journal of King Saud University-Science, 35(6), 102754.

[16] Sun, Z., Wang, G., Li, P., Wang, H., Zhang, M., & Liang, X. (2024). An improved random forest based on the classification accuracy and correlation measurement of decision trees. Expert Systems with Applications, 237, 121549.

[17] Wang, C., Xiao, W., & Liu, J. (2023). Developing an improved extreme gradient boosting model for predicting the international roughness index of rigid pavement. Construction and Building Materials, 408, 133523.

[18] Çakir, M., Yilmaz, M., Oral, M. A., Kazanci, H. Ö., & Oral, O. (2023). Accuracy assessment of RFerns, NB, SVM, and kNN machine learning classifiers in aquaculture. Journal of King Saud University-Science, 35(6), 102754.

[19] Gao, B., Zhou, Q., & Deng, Y. (2024). HIE-EDT: Hierarchical interval estimation-based evidential decision tree. Pattern Recognition, 146, 110040.

[20] Niazkar, M., Menapace, A., Brentan, B., Piraei, R., Jimenez, D., Dhawan, P., & Righetti, M. (2024). Applications of XGBoost in water resources engineering: A systematic literature review (Dec 2018–May 2023). Environmental Modelling & Software, 105971.

[21] Xing, H. J., Liu, W. T., & Wang, X. Z. (2024). Bounded exponential loss function based AdaBoost ensemble of OCSVMs. Pattern Recognition, 148, 110191.