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

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

«На правах рукопису»
УДК 004.89, 004.912

«До захисту допущено»
Завідувач кафедри
_____ Едуард ЖАРІКОВ
«___»_____2022 р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою «Інженерія програмного забезпечення інформаційних систем»

зі спеціальності 121 «Інженерія програмного забезпечення»

на тему: «Інтелектуальна система виявлення пропаганди в текстах»

Виконав:
Студент ІІ курсу, групи ІТ-01мн
Минзар Богдан Миколайович _____

Керівник:
Професор кафедри ІПІ, д.т.н, проф.,
Стеценко Інна Вячеславівна _____

Рецензент:
Професор кафедри ОТ, д.т.н., доц.,
Клименко Ірина Анатоліївна _____

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

Київ – 2022 року

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

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Рівень вищої освіти – другий (магістерський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-наукова програма «Інженерія програмного забезпечення комп'ютерних систем»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Едуард ЖАРІКОВ

«___»_____2022р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Минзару Богдану Миколайовичу

1. Тема дисертації «Інтелектуальна система виявлення пропаганди в текстах», науковий керівник дисертації Стеценко Інна Вячеславівна, д.т.н, професор, затверджені наказом по університету від «24» квітня 2022 р. № 88

2. Термін подання студентом дисертації «6» червня 2022 р.

3. Об'єкт дослідження – обробка природної мови у контексті методів розпізнавання пропаганди в текстах.

4. Предмет дослідження – методи та засоби розпізнавання пропаганди в текстах.

5. Перелік завдань, які потрібно розробити – аналіз існуючих досліджень; аналіз існуючих засобів; розробка моделі класифікації; адаптація моделі класифікації для роботи з даними українською мовою; дослідження ефективності запропонованої моделі; проєктування архітектури та побудова прототипу системи;

6. Орієнтовний перелік графічного (ілюстративного) матеріалу – презентація з матеріалами за результатами дослідження

7. Орієнтовний перелік публікацій – одна публікація

8. Дата видачі завдання «30» вересня 2021 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання	Примітка
1	Аналіз стану досліджень	31.01.22	
2	Ознайомлення з теоретичною базою, методами та засобами аналізу текстів природною мовою для вирішення задачі	04.04.22	
3	Аналіз існуючих засобів	18.04.22	
4	Розробка підходу до збирання даних. Формування набору даних	02.05.22	
5	Побудова моделі класифікації	12.05.22	
6	Адаптація моделі класифікації для роботи з даними українською мовою	15.05.22	
7	Розробка архітектури системи	18.05.22	
8	Побудова прототипу системи	24.05.22	
9	Перевірка запропонованого рішення	26.05.22	

Студент Богдан МИНЗАР

Науковий керівник Інна СТЕЦЕНКО

РЕФЕРАТ

Розмір пояснювальної записки — 77 аркушів, містить 18 ілюстрацій, 10 таблиць, 7 додатків, 4 розділи та 32 джерела.

Актуальність теми. Пропаганда — комунікаційна стратегія, що відома ефективністю та, зазвичай, намаганням ввести в оману для поширення певної точки зору, переконання, нав'язування поглядів. Певні категорії населення (наприклад, молодь) статистично є особливо вразливими. Характерною рисою пропаганди є використанням усталених прийомів та засобів, що вже досліджувались протягом XX та XXI сторіч. Тож актуальним є завдання створення програмного засобу, системи для автоматичного виявлення пропаганди в текстах, для інформування вразливих категорій населення.

Метою дослідження є розробка системи, що аналізує наданий текст на предмет наявності в ньому пропаганди та виводить інформацію про результати аналізу.

Об'єкт дослідження — обробка природної мови у контексті методів розпізнавання пропаганди в текстах.

Предмет дослідження — методи та засоби розпізнавання пропаганди в текстах.

Новизна роботи:

– Вперше сформовано набір даних для класифікації, що включає сучасні зразки державної пропаганди Російської Федерації в медіа за період повномасштабного вторгнення до України 2022 року. Проведено дослідження на цьому наборі даних.

– Надано подальшого розвитку способу виявлення пропаганди в текстах англійською мовою за рахунок її адаптації для роботи з українською мовою.

Практичне значення одержаних результатів. Практичні результати магістерської дисертації мають цінність в сфері аналізу текстів природною мовою.

Результати включають реалізацію прототипу запропонованої системи, що аналізує наданий текст, використовуючи навчену модель виявлення пропаганди та надає відповідь з результатами аналізу. Система, надана як сервіс, може бути використана для побудови процесу фільтрації контенту в інших системах.

Ключові слова: ОБРОБКА ПРИРОДНОЇ МОВИ, НЕЙРОННІ МЕРЕЖІ, РОЗПОДІЛЕНІ СИСТЕМИ, МІКРОСЕРВІСНА АРХІТЕКТУРА, PYTHON

ABSTRACT

Explanatory note size – 77 pages, contains 18 illustrations, 10 tables, 7 applications, 4 sections, 32 sources.

Topicality. Propaganda is a communication strategy known for its effectiveness and, as a rule, for its attempts to mislead the audience to spread some agenda, persuade, and impose certain viewpoints. Some population categories (e.g. youth) are statistically particularly vulnerable to it. A key feature of propaganda is the use of certain established techniques and tools researched during the XX and XXI centuries. Therefore, the task to design a software tool, a system for automatic detection of propaganda in texts, to inform vulnerable categories of the population is topical.

The purpose of the study is to develop a system that analyzes the text provided for the presence of propaganda in it and displays information about the results of the analysis.

The object of research is the natural language processing in the context of methods of propaganda detection in texts.

The subject of research is methods and means of recognizing propaganda in texts.

The novelty of work:

– For the first time, a new dataset for the classification has been formed. It includes modern examples of Russian Federation state propaganda in the media for the period of the full-scale invasion of Ukraine in 2022. The research has been conducted on this dataset.

– The method for propaganda detection in texts was further developed for processing the Ukrainian language.

The practical value of the obtained results. The practical results of the master's dissertation are valuable in the field of natural language processing.

The results include the implementation of a prototype of the proposed system that analyzes the provided text using a trained model of propaganda detection and provides a response with the results of the analysis. The system, provided as a service, could be used to build a content filtering process in other systems.

Keywords: NATURAL LANGUAGE PROCESSING, NEURAL NETWORKS, DISTRIBUTED SYSTEMS, MICROSERVICE ARCHITECTURE, PYTHON

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

API – Application Programming Interface; прикладний програмний інтерфейс

BERT – Bidirectional Encoder Representations from Transformers; методика машинного навчання для обробки природної мови

CSV – Comma-Separated Values; текстовий формат зберігання структурованих даних

GloVe – Global Vectors for Word Representation; векторна модель розподіленого представлення слів

HTML – HyperText Markup Language; мова розмітки веб-сторінок

TF-IDF – метрика для визначення “ваги” слова в певному текстовому документі

XML – Extensible Markup Language; формат розмітки та зберігання та передачі даних

Word2Vec – методика обробки природної мови з метою створення так званих “вкладень”

ВСТУП

В наш час гострою є проблема поширення неправдивої інформації про певні події та явища з метою впливу на суспільну думку, для позитивної репрезентації однієї сторони чи підриву довіри до іншої та втілення цих хибних уявлень до свідомості людей. Особливо виражене це явище в контексті ведення гібридної війни, що поєднує застосування конвенційних збройних сил та інформаційної війни, що в свою чергу включає поширення пропаганди [1, 2]. Зазвичай в пропаганді використовуються усталені наративи та прийоми [3], що можна розпізнавати. Тому, актуальним є завдання розробки методу та моделі автоматичного виявлення пропаганди в текстах, а саме в дописах та коментарях в соціальних мережах з метою їх позначення як пропагандистських чи фільтрації.

Пропаганда являє собою ефективну, але часто оманливу комунікаційну стратегію, яка використовується для просування певної точки зору, наприклад, у політичному контексті [4, 5]. Метою цієї комунікаційної стратегії є переконати аудиторію у вірності такої точки зору шляхом введення в оману та/або за допомогою використання часткових аргументів [3], що особливо шкідливо для більш вразливих верств населення (наприклад, молоді чи літніх людей) [5, с.160]. Тому здатність виявляти випадки пропаганди в політичному дискурсі та газетних статтях має основне значення, а методи та технології обробки природної мови відіграють головну роль у цьому контексті, вирішуючи завдання виявлення та класифікації пропаганди [6]. Важливо, зокрема, проінформувати цю вразливу групу про проблему та надати їй інструменти, що здатні підвищити їх обізнаність та розвивати критичне мислення.

Дослідження в області виявлення пропаганди наразі знаходяться на початковій стадії, незважаючи на ефективність пропаганди та необхідності протидії їй. Окрім того, існує досить багато проблем, з якими стикаються дослідники при розробці методів автоматичного виявлення пропаганди. Щоб досягти цієї амбітної мети, в даній роботі розглядається проєктування нового інструменту, а саме інтелектуальної системи для автоматичного визначення та класифікації пропаганди в текстах.

Метою дослідження є розробка системи, що аналізує наданий текст на предмет наявності в ньому пропаганди та виводить інформацію про результати аналізу. Об’єктом дослідження є обробка природної мови у контексті методів розпізнавання пропаганди в текстах. Предметом дослідження є методи та засоби розпізнавання пропаганди в текстах.

Наукова новизна роботи включає:

– Вперше сформовано набір даних для класифікації, що включає сучасні зразки державної пропаганди Російської Федерації в медіа за період повномасштабного вторгнення до України 2022 року. Проведено дослідження на цьому наборі даних.

– Надано подальшого розвитку способу виявлення пропаганди в текстах англійською мовою за рахунок її адаптації для роботи з українською мовою.

Практичні результати роботи включають реалізацію прототипу запропонованої системи, що аналізує наданий текст, використовуючи запропоновану модель виявлення пропаганди та надає відповідь з результатами аналізу.

1 ЗАДАЧА ВИЯВЛЕННЯ ПРОПАГАНДИ. ПІДХОДИ ДО ВИЯВЛЕННЯ ПРОПАГАНДИ В КОНТЕКСТІ ОБРОБКИ ПРИРОДНОЇ МОВИ

1.1 Постановка задачі виявлення пропаганди

Основна задача аналізу тексту, для виявлення в ньому пропаганди, може бути сформульована наступним чином: якщо надано текст, що є суб'єктивним висловлюванням щодо певного об'єкту, виявити характер тексту як одну з двох полярностей: пропагандистський або не пропагандистський. Одним з мінусів даного підходу є те, що певний текст не завжди можна однозначно класифікувати, тобто документ може містити як ознаки пропагандистської риторики, так і звичайні за характером твердження [11]. Альтернативним відображенням результатів аналізу може бути спектр, дійсне число від 0 до 1, що дає можливість отримати метрику у вигляді індексу — на який відсоток певний текст відповідає критеріям пропагандистського за оцінкою алгоритму.

1.2 Пропаганда

Щоб дослідити методику виявлення пропаганди у контексті обробки природної мови, слід спершу розглянути саме явище пропаганди, а також доступні методи та засоби пропаганди, що зазвичай використовуються для її створення.

1.2.1 Методи та засоби пропаганди

Пропаганда — навмисна, систематична спроба сформувати сприйняття, маніпулювати знаннями, та направляти поведінку на досягнення очікуваного результату пропагандиста [7]. Пропаганда існує в багатьох формах. Загалом, розпізнати її можливо по її наміру переконати певну аудиторію.

Серед ознак, зокрема, але не виключно: апелювання до порядку денного певної соціальної групи, використання хибних міркувань, а також вживання певних мовних конструкцій, фраз тощо. Загалом, кількість характерних ознак пропаганди та самі ознаки варіюються в різних джерелах. Серед ознак, корисними для автоматичного розпізнавання можна виділити наступні:

– Нечіткість, неоднозначність, намагання викликати сумніви. Тут мається на увазі, що використовуються слова, які спеціально можуть мати неоднозначне трактування, щоб читачі могли створювати власні інтерпретації подій.

– Повторення. Мається на увазі повторення одного й того ж посилу з метою врешті схилити аудиторію до його прийняття.

– "Перевантажене" мовлення (*loaded language*). Використання специфічних слів та фраз, що мають сильне емоційне забарвлення, навантаження.

– Апелювання до емоцій. Зображення подій та ситуацій з (занадто) сильним негативним чи позитивним емоційним забарвленням.

Також до перелічених вище факторів варто додати фактор контексту та наявності кліше, характерних для певного пропагандистського дискурсу. Наприклад, пропаганда СРСР чи більш сучасна пропаганда Російської Федерації спирається на відомі наративи та стереотипи, що поширюються протягом десятиліть. Зокрема, в дослідженні Львівського університету державної пропаганди Російської Федерації за 2014-2018 рік висвітлюється шаблонність такої пропаганди в новинах: "Окрім того, досить часто журналісти Pravda.ru використовують метод навішування ярликів. У матеріалах регулярно з'являються слова та назви, які зумовлюють негативні асоціації на підсвідомому рівні: «фашисти», «каратели», «каральні війська», «хунта», «народний мер», «народний губернатор» (мається на увазі самопроголошений), «головорізи», «ультраправі»" [9, с.4]

Отже, як висновок, для автоматичного виявлення пропаганди можна використати словник, зібраний з ключових слів пропагандистського дискурсу певного джерела відносно якогось суб’єкту чи явища. Більш складним є виявлення перелічених вище факторів, незалежних від виду, джерела чи контексту пропаганди, як-то апелювання до емоцій чи неоднозначність висловлювань.

1.2.2 Пропаганда в мережі Інтернет

Пропаганда в мережі Інтернет — доволі поширене явище. Зокрема, соціальна мережа Твіттер часто використовується як середовище поширення пропаганди. Twitter може використовуватись терористичними та екстремістськими групами. Найчастіше ж використовується для поширення державної пропаганди — особливо державної пропаганди Російської Федерації [10] та КНР.

Акаунти (облікові записи), що поширюють пропаганду в Twitter, можуть бути завідомо пропагандистськими, як російські медіа (рис. 1.1). З даного допису можна зробити висновок про його пропагандистський характер виходячи з наступних ознак:

– Наявність ключових слів «нацисты», «Народная милиция», «ДНР», характерних для російського пропагандистського дискурсу

– Апелювання до емоцій (див. підрозділ 1.2.1 Методи та засоби пропаганди)

Пропаганду можуть поширювати і звичайні користувачі, а не лише облікові записи медіа. Користувачі можуть виглядати як і завідомо підозріло (так звані «боти»), так і можуть бути на перший погляд обліковими записами реальних людей.

Рисунок 1.1 – Пропагандистський допис російського державного медіа "RT на русском"

Розпізнати умисно створений акаунт для поширення пропаганди чи певної дезінформації — доволі тривіальна задача. Серед ключових ознак, що можуть викликати підозру: дата реєстрації близька до певної події, що описується, мала кількість підписників або ж їх відсутність, відсутність фото профілю або ж використання чужого.

Розпізнати акаунт популярного користувача чи відомої людини, що поширює пропаганду, набагато складніше, особливо для людини, що не є обізнаною в контексті інформації, що поширюється. Пропаганду можуть поширювати навіть і так звані верифіковані акаунти, тобто такі, особа яких була підтверджена адміністрацією мережі Twitter. А також, пропаганду поширювати можуть відомі люди з великою кількістю підписників.

Для прикладу поширення пропаганди відомими обліковими записами розглянемо допис користувача Javier Couso від 8 квітня 2022 року (рис. 1.2). Допис стосується атаки ракетою типу «Точка-У» Краматорську, в якій загинули цивільні. Російські джерела пропаганди стверджують, що атака була здійснена збройними силами України, що було неодноразово спростовано. Допис створено верифікованим користувачем, тобто за свідченням Twitter, це реальна доволі відома особа, що керує обліковим записом, оскільки Twitter вирішив верифікувати його обліковий запис. Акаунт має 80,2 тисячі підписників, а допис від 10 квітня 2022 отримав 1 589 розповсюджень, 164 цитування, 2 319 вподобань. Невідомо, скільки людей його переглянуло, але з наведених вище даних можна припустити, що це десятки тисяч переглядів.

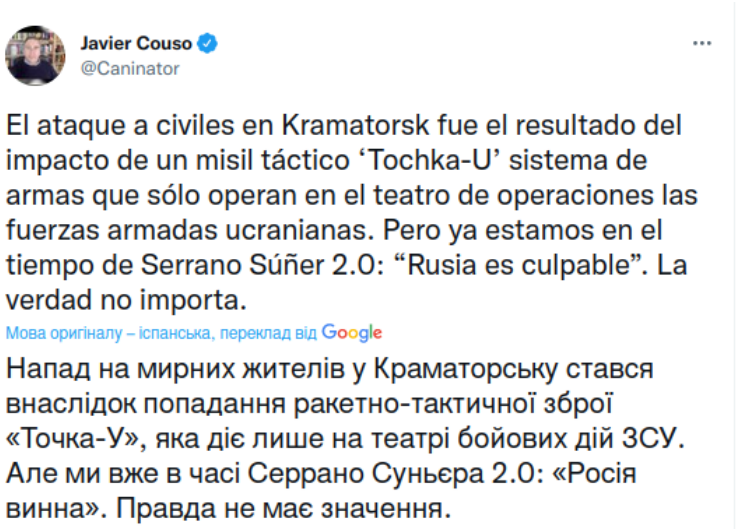


Рисунок 1.2 – Допис користувача Javier Couso в Twitter

Отже, щоб відповісти на питання, чи є певний допис пропагандистським, слід проаналізувати його вміст — зокрема, і в першу чергу, текстове наповнення відносно контексту. Слід перевірити текст допису на відповідність критеріям пропагандистської риторики та наявність відповідних ключових слів.

1.3 Існуючі дослідження в області автоматичного виявлення пропаганди

Велика кількість досліджень на тему виявлення пропаганди базуються на аналізі даних на різних онлайн-ресурсах для новин [7, 11, 12], а також на створених користувачами у соціальних мережах дописах.

Дослідження, опубліковане науковцями Сеульського національного університету 2021 року [11], розглядає саме процес виокремлення ознак для якісного виявлення пропаганди. Визначаються якісно описові ознаки та аналізується їх придатність для виявлення прийомів обману. Далі демонструється, що розроблені функції та моделі можна поєднати з вже попередньо підготовленими мовними моделями, що дає найкращі на момент публікації результати.

Інша робота від дослідників університету Ніцци (Université Côte d'Azur) розглядає побудову системи для збирання та попередньої обробки даних для подальшого використання з метою аналізу на пропаганду [14].

Найновіші підходи до виявлення пропаганди засновані на мовних моделях, які переважно включають архітектури на основі трансформерів. Підхід, який найкраще показав себе у задачі класифікації на рівні речень, базується на архітектурі BERT з налаштуванням гіперпараметрів без функції активації [12]. В цьому дослідженні зосередилися на етапах попередньої обробки, щоб надати більше інформації про мовну модель. Дослідники використовують архітектуру BERT, зводячи задачу до задачі маркування послідовності (sequence labeling).

1.4 Підходи до обробки природної мови. Попередня обробка даних.

Перед виконанням класифікації тексту зазвичай проводять попередню обробку тексту. Правильно проведена попередня обробка є такою ж важливою, як і правильний алгоритм для безпосередньої класифікації тексту.

Представимо загальний метод класифікації тексту у спрощеному вигляді як певну функцію $f(y, C)$

$$f(y, C) = \begin{cases} 1, y \in C \\ 0, y \notin C \end{cases} \quad (1.1)$$

$$y = p(x), \quad (1.2)$$

де x — це вихідний текст, що надходить до системи,

$p(x)$ — функція попередньої обробки тексту,

C — певна множина текстів, належність до якої потрібно виявити.

Аргумент функції класифікації y — це результат від функції попередньої обробки, тобто вхідні дані методу класифікації. В такому разі результат класифікації повністю залежить від попередньої обробки даних. Робимо висновок, що недосконало виконана попередня обробка може призвести до отримання невірних результатів, як і відповідно правильно підібрана під конкретну задачу, корпус текстів та мову функція може покращити результати відносно результатів без попередньої обробки чи з базовою обробкою.

Попередня обробка тексту зазвичай включає такі етапи, але не обов’язково всі (рис. 1.3):

- Нормалізація
- Токенізація
- Видалення стоп-слів
- Стемінг або лематизація

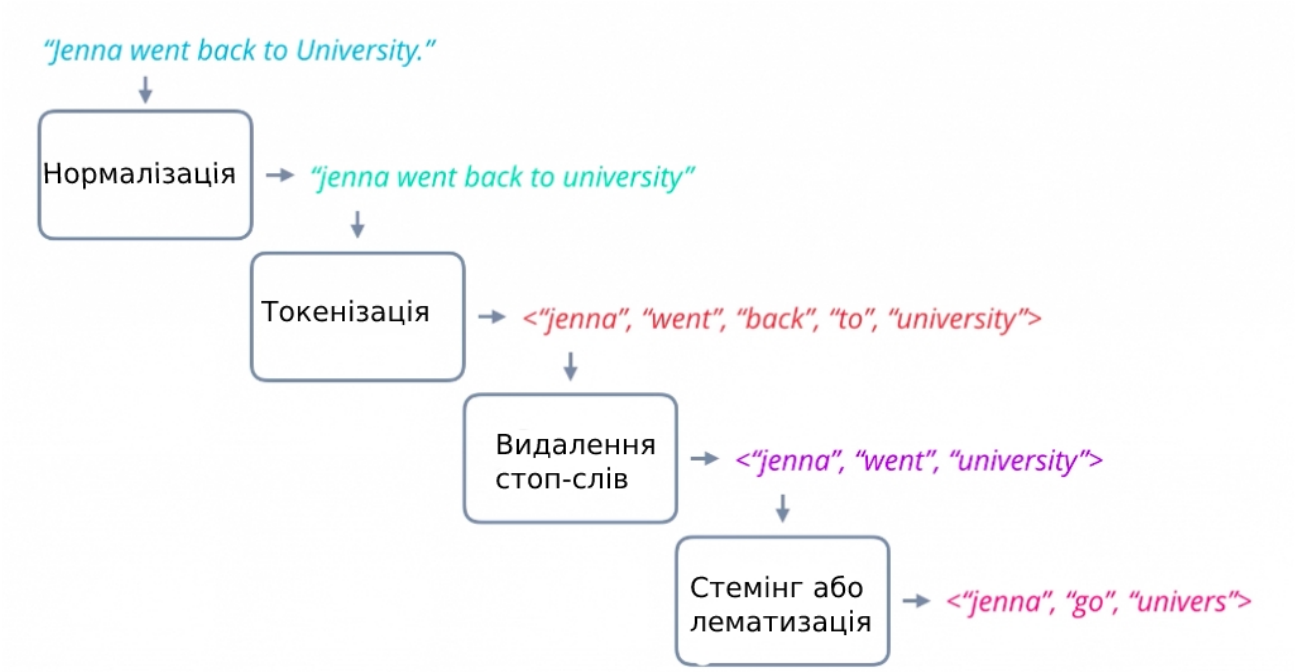


Рисунок 1.3 – Процес попередньої обробки тексту

Розберемо детальніше кожен з етапів попередньої обробки.

Нормалізація — процес, за допомогою якого послідовність слів приводиться до більш однорідної послідовності. Конвертуючи послідовність слів до їх певного стандартного вигляду, ми полегшуємо наступні кроки. Під час нормалізації зменшується можлива ентропія в тексті без втрати інформативності, що допомагає зменшити кількість інформації яку потрібно обробити комп’ютеру, і відповідно покращує ефективність алгоритму.

Токенізація — техніка обробки текстових даних, що передбачає розбиття тексту на окремі частини — токени. Токен — це частина послідовності символів в певних текстових документах, що об’єднані разом і розглядаються як окрема одиниця для подальшої обробки. Токенами можуть бути: слова, речення, або символи. В той же час токенізація передбачає відкидання певних символів, як-то розділових знаків, пробілів, переносів, інших спецсимволів. Загалом і найчастіше в прикладних задачах обробки природної мови кожен токен — це окреме слово.

Це досить звична техніка в задачах обробки природної мови. Для демонстрації різних способів токенізації розглянемо наступне речення: *“Токенізація — один з перших етапів в алгоритмах обробки природної мови. Задача токенізації — розділити текст на одиниці з семантичним значенням.”*

Токенізація з розбиттям по словах:
“Токенізація", “один”, “з”, “перших”, “етапів”, “в”, “алгоритмах”, “обробки”, “природної”, “мови”, “Задача”, “токенізації”, “розділити”, “текст”, “на”, “одиниці”, “з”, “семантичним”, “ значенням”

Токенізація з розбиттям по реченнях:
“Токенізація — один з перших етапів в алгоритмах обробки природної мови”, “Задача токенізації — розділити текст на одиниці з семантичним значенням”

Відповідно, в залежності від виду токенізації ми за атомарну одиницю беремо різні за розміром сегменти, що може бути корисним для тієї чи іншої задачі. Проблеми, що можуть виникнути при підборі правильного алгоритму токенізації:

– Чи слід розбивати текстову послідовність на токени, опираючись на всі символи, що не є літерами? (Тобто пробіли, тире, дефіси тощо)

– Правильна обробка апострофів (в залежності від мови, апостроф має різне значення)

– Правильна обробка складних слів, тобто таких, що складаються з кількох слів (наприклад Велика Британія)

Тож підбір чи побудова ефективного алгоритму токенізації залежить від мови та контексту. Для цього може використовуватись попереднє визначення мови для підбору та налаштування алгоритму токенізації.

Видалення *стоп-слів* — етап, що передбачає відкидання певних слів перед обробкою тексту природною мовою (або ж після, в залежності від задачі). Зазвичай стоп-слова — це слова, що найчастіше зустрічаються в корпусі текстів певною мовою, але при цьому не існує універсального списку стоп-слів, що підходив би одночасно для вирішення всіх задач з обробки природної мови. Найчастіше під стоп-словами розуміють сполучники (наприклад “а”, “і”, “би”, “же” тощо в українській мові), артиклі (“a”, “the” тощо в англійській мові), знаки пунктуації (“.”, “,”, “!”, “?” тощо), цифри. Іноді, в залежності від задачі, до списку стоп-слів також може бути включена нецензурна лексика.

Стемінг та лематизація потрібні для приведення слів до їх однієї базової форми. В текстах природною мовою може зустрічатись велика кількість різних форм одного слова (приклад: комп’ютер, комп’ютерний, комп’ютерна, комп’ютерні). Для ефективної алгоритмічної обробки текстів слід уніфікувати представлення таких слів, тобто до базової форми.

Стемінг — нормалізація окремо взятих слів до їх базової або кореневої форми. По своїй природі це значно простіший процес, ніж лематицзація. Зазвичай під стемінгом розуміють евристичну обробку слова, що “відрізає” закінчення слова, щоб привести його до базової форми. Приклад застосування стемінгу розглянуто в таблиці 1.1.

Таблиця 1.1 – Приклад стемінгу

Слово	Результат стемінгу
ортогональне	ортогонал
шкільна	шкіл
військовим	військов

Лематизація — зведення слова до його леми. Загалом, це більш складний процес ніж стемінг. Приклад результату лематизації розглянуто у таблиці 1.2. Лематизація передбачає використання словників та морфологічного аналізу слів, щоб надати базову словникову форму слова, що називається *лемою*, від якої й походить слово що піддається лематизації. Оскільки лематизація — це доволі складний процес, то зазвичай її застосування передбачає використання готових програмних бібліотек, наприклад, таких як *spaCy*, а не розробку власного рішення.

Таблиця 1.2 – Приклад результату лематизації

Слово	Лема слова
ортогональне	ортогональний
шкільна	шкільний
військовим	військовий

1.5 Підходи до обробки природної мови. Алгоритми класифікації.

Розглянемо підходи, що можуть допомогти вирішити задачу автоматичного виявлення пропаганди в контексті обробки природної мови.

1.5.1 Алгоритм на основі ключових слів

Суть підходу полягає в тому, що отримавши певний текст для перевірки, алгоритм перевіряє текст на наявність в ньому специфічних ключових слів або фраз, що були заздалегідь визначені. Для цього спершу потрібно проаналізувати наявний корпус пропагандистських текстів, виявити в ньому слова, що характерні саме для пропаганди та можуть ідентифікувати певний інший текст як належний до цієї категорії.

Переваги алгоритму – відносна простота реалізації, а отже і легке відлагодження, передбачуваність результатів, що цілком залежать від вхідних даних – тексту для перевірки та словника ключових слів. Серед недоліків можна відмітити складність наповнення такого словника – це потребує великої кількості ручної роботи. Потрібно провести попередню обробку текстових даних, далі зібрати статистику по найбільш вживаних словах та серед них вибрати ті, що мають пропагандистський характер.

Також, оскільки цей метод спирається на словник, то алгоритм буде прив'язаний до пропаганди щодо конкретного суб'єкту, конкретного джерела та періоду часу. Це пов'язано з тим, що з часом риторика може змінюватись, щодо інших явищ та об'єктів можуть використовуватись інші ключові слова.

Варто відмітити, що ймовірно цей алгоритм може мати доволі високий відсоток хибно-позитивних спрацювань у випадку, коли буде він буде застосований до тексту, в якому є цитування або ж непряма мова з пропагандистського джерела.

1.5.2 Аналіз тональності тексту

Аналіз тональності не є самостійним методом в контексті виявлення пропаганди, але може бути застосований як додаток для покращення точності інших методів. Мотивація використання цього підходу базується на характерності однієї з властивостей пропаганди, а саме — сильно поляризоване емоційне забарвлення тексту, що є технікою апелювання до емоцій (див. підрозділ 1.2.1).

За визначенням, аналіз тональності тексту (також відомий як сентимент-аналіз) – це використання методів обробки природної мови, аналізу текстів, комп'ютерної лінгвістики для визначення, вилучення, вимірювання та дослідження емоційного забарвлення та суб'єктивної інформації [15].

Аналіз тональності текстів – тема, що потребує окремої уваги. Для розв’язання цієї задачі існують різні методи, що відрізняються точністю, складністю реалізації та швидкодією. Загалом, останнім часом все більшої популярності набирають методи сентимент-аналізу на основі нейронних мереж та алгоритмів машинного навчання, оскільки вони дозволяють проводити аналіз доволі складних областей, наприклад, новин – новини зазвичай не відрізняються вираженим проявом емоцій авторів.

1.5.3 Алгоритм TF-IDF

TF-IDF (Term Frequency — Inverted Document Frequency) — статистична метрика, що визначає показник важливості певного слова у контексті окремо взятого документу.

– Term Frequency — важливість слова у межах окремо взятого документа.

– Inverted Document Frequency — загальна кількість слів у всіх документах текстового корпусу.

Таким чином, найбільшу вагу матимуть слова, що часто зустрічаються в певному документі, та рідко зустрічаються в інших. Ця техніка часто застосовується в системах повнотекстового пошуку, зокрема, Elasticsearch та Solr. Також алгоритм TF-IDF може бути використаний для виділення ключових слів з документів.

1.5.4 Методи на основі нейронних мереж

Word2Vec — алгоритм обробки природної мови, що базується на використанні нейронних мереж. Ця модель нейронної мережі навчається встановлювати асоціації між словами на великих об’ємах текстових даних. Модель відображає кожне слово як певний список номерів, що відповідно називають вектором.

Натренована нейронна мережа може виявляти слова-синоніми в тому контексті, що була натренована та може бути використана для визначення індексу семантичної подібності між різними словами [16].

GloVe — алгоритм машинного навчання без вчителя для отримання векторних представлень слів. Перевага моделі GloVe — на відміну від Word2Vec, ця модель використовує глобальну статистику замість того щоб спиратись лише на локально зібрані дані (тренувальну вибірку). GloVe натомість використовує як і локальну статистику корпусу текстів (тренувальної вибірки), так і глобальну статистику. Використання двох видів статистики має свої переваги.

Використання локальної статистики в Word2Vec дає перевагу в задачах пошуку синонімів для слів та аналогій, в той же час глобальна статистика має свою перевагу над локальною (краще відображення контекстної важливості слова).

Висновки до розділу

У даному розділі визначена та описана предметна область дослідження магістерської дисертації. Проведено огляд пропаганди як явища та засобу впливу на суспільну думку. Розглянуто характерні ознаки пропаганди та способи її поширення в медіа, зокрема, у соціальних мережах.

Проведено огляд загальної задачі аналізу текстів та задачі виявлення пропаганди в контексті обробки природної мови. Розглянуто, які засоби обробки природної мови зазвичай застосовуються дослідниками для вирішення подібних задач. Окрім того, описано, які наразі є існуючі дослідження в області автоматичного виявлення пропаганди в текстах.

2 ФОРМУВАННЯ НАБОРУ ДАНИХ

Щоб розробити алгоритм класифікації, спершу слід зібрати та підготувати набір даних. Для багатьох видів задач класифікації вже існують набори даних в вільному доступі, проте не для всіх. Також, не всі набори даних що описуються в наукових працях з виявлення пропаганди є у вільному доступі. Отже, було прийнято рішення зібрати набір даних самостійно. Для задачі потрібно зібрати набір даних, що буде складатись з текстів двох класів: пропаганда та не пропаганда. Розглянемо джерела, з яких можна зібрати дані що відповідають поставленій меті, а також підхід, методи та засоби збирання даних з цих джерел в наступних підрозділах.

2.1 Вибір джерел для збирання даних

Загалом, у новинних ресурсах, що розглядаються, є два різних формати: повноцінні статті на сайті та короткі дописи в соціальній мережі Twitter — твіти. Кожен твіт — це короткий допис довжиною до 280 символів. Оскільки ми розглядаємо один і той же медіаресурс, то твіти мають бути лаконічними відносно статей та загалом містити інформацію на тематику діяльності сайту з новинами, але у більш стислому вигляді. Слід розглянути та проаналізувати обидва формати: повноцінні статті та більш короткі дописи з Twitter та порівняти отримані результати. Для цього зберемо дані обох типів, щоб потім проаналізувати їх.

Потрібно зібрати дані двох класів. Для даних першого класу (пропаганда) як джерело оберемо державну пропаганду Російської Федерації в медіа. Ресурс RT (Russia Today) — російська державна медіакомпанія, що відповідно фінансується з державного бюджету РФ та орієнтується на західну — більшою мірою, англомовну, аудиторію. Окрім телеканалу, компанія має власний веб-сайт та акаунт в соціальній мережі Твіттер англійською та російською мовами.

RT заблоковано в Україні після нелегальної окупації Російською Федерацією території автономної республіки Крим. Литва та Латвія заблокували ресурс в 2020 році. Починаючи з лютого 2022 року, ресурс формально заблоковано на території ЄС, хоча доступ до нього все ще можна отримати. Деякі сервіси, зокрема, YouTube, теж заблокували RT. Цей ресурс вже неодноразово досліджувався та існують десятки праць, що аналізують та класифікують пропаганду, поширювану ним [17, 18]. Отже, можна стверджувати, що цей ресурс є пропагандистським, а тому візьмемо матеріали з нього як дані для класу “пропаганда”.

MFA Russia — офіційний акаунт Міністерства Зовнішніх Справ Російської Федерації в Twitter. Поширює пропаганду, зокрема, звинувачення військовослужбовців батальйону “Азов” у неонацизмі, а також і звинувачення всіх українців у неонацизмі. Використовуються специфічні фрази, характерні для російської пропаганди, зокрема “Kiev regime”, “Collective West” тощо.

Розглянемо кілька ресурсів, що будуть використовуватись для даних другого класу. Bloomberg News — підрозділ Bloomberg L.P., інтернаціонального новинного агенства зі штаб-квартирою в Нью-Йорку, що публікує новини політики. BBC World — інтернаціональний британський новинний ресурс, що покриває новини зі всього світу на різну тематику. The Kyiv Independent — український новинний ресурс англійською мовою, що висвітлює події в Україні та Східній Європі та фінансується за моделлю фандрейзингу, тобто фінансується читачами.

2.2 Розробка засобів збирання даних

Відповідно до задачі збору даних з джерел, визначених вище (див. підрозділ 2.1), потрібно зібрати дані з новинних ресурсів, а також з різних медіа-акаунтів в соціальній мережі Twitter. Дані будемо збирати до файлів у форматі CSV.

Деякі набори даних вже існують в відкритому доступі (наприклад, вибірка новин з BBC для класифікації), хоча й мають певні обмеження. Використаємо ці дані, а для решти ресурсів проведемо збирання даних в подальших пунктах.

2.2.1 Розробка скрейпера сайту RT.com

Готового актуального набору даних з веб-сайту RT у вільному доступі немає. Щоб зібрати дані з сайту, потрібно розробити так званий парсер або скрейпер (scraper). скрейпер — це програма, що збирає інформацію з веб-сторінок у форматі HTML та перетворює її у дані певної визначеної структури.

Веб-сторінки призначені для перегляду в браузерах, а тому найчастіше обмін інформацією браузера з сервером відбувається по протоколу HTTP/HTTPS даними в форматі HTML. HTML є мовою розмітки, і призначений для відображення контенту в браузері. Відповідно він містить метадані, інформацію про стилі для візуального відображення, а текстові дані розподілені по різних вузлах. Такий формат в загальному вигляді не є зручним для обробки, адже нас цікавить саме текст природною мовою.

Принцип робота скрейпера можна описати так: програмою спочатку завантажується сторінка, потім з неї добувається потрібна інформація за допомогою парсера HTML формату. Необхідні дані збираються, агрегуються, приводяться до потрібної схеми даних та зберігаються у файл або ж до бази даних для подальшого використання, передаючи з кожним запитом до серверу.

У сайту є так звана карта (sitemap) — список всіх наявних сторінок веб-сайту у форматі XML. Карти сайту зазвичай потрібні для правильної індексації сторінок пошуковими ресурсами на кшталт Google. Використаємо карту сайту для отримання всіх сторінок за поточний 2022 рік, оскільки цікавлять актуальні дані.

Розглянувши структуру карти сайту за 2022 рік (див. рис. 2.1), можна помітити, що кожен url вузол-нащадок кореневого вузла urlset xml-документу містить посилання на статтю — дочірній вузол loc та медіа-файли що належать до статті. Отже, зберемо всі посилання на статті з вузлів loc. Для цього потрібно розібрати xml-документ в представлення, зручне для маніпуляції. Скористаємось для цього бібліотекою BeautifulSoup для мови програмування Python. Це популярна бібліотека для парсингу (розбору) та проведення маніпуляцій з HTML та XML документами.

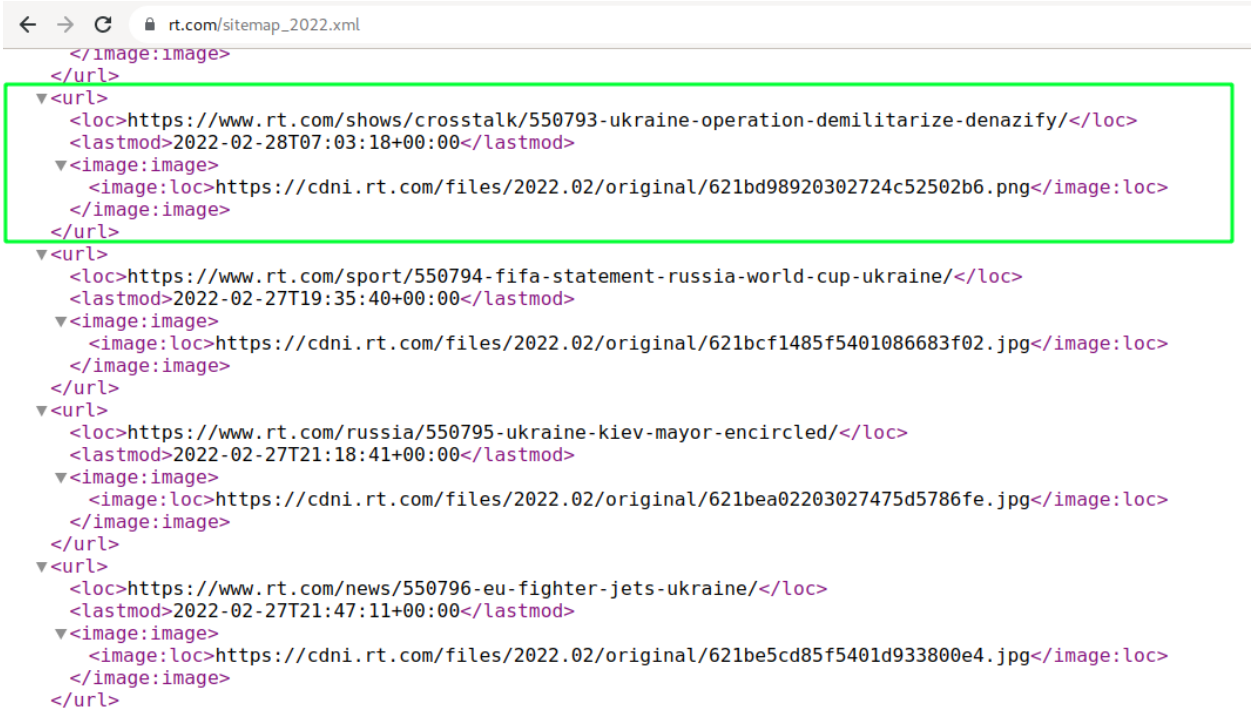


Рисунок 2.1 — Структура карти сайту rt.com (sitemap_2022.xml)

Далі потрібно завантажити вміст кожної веб-сторінки у форматі HTML, розібрати його та витягти текстову інформацію. Для розбору HTML-сторінки використаємо бібліотеку BeautifulSoup. Пошук елементів з інформацією будемо робити за допомогою CSS-селекторів.

Підхід з використанням CSS-селекторів, на відміну від підходу XPath, дозволяє мінімально прив’язуватись до розмітки сайту та продовжити коректно функціонувати при зміні структури дерева HTML веб-сторінки (при умові, що не зміняться CSS-селектори) — отже, є більш стійким та надійним. Всі статті на сайті мають однакову структуру, тому розглянемо пошук селекторів на прикладі однієї.

Зі статті з веб-сайту RT можна виділити такі селектори (рис. 2.2):

– *.article__heading* — заголовок статті;.
– *article__summary* — анотація до статті;
– *.article__text* — тіло (основна частина) статті;

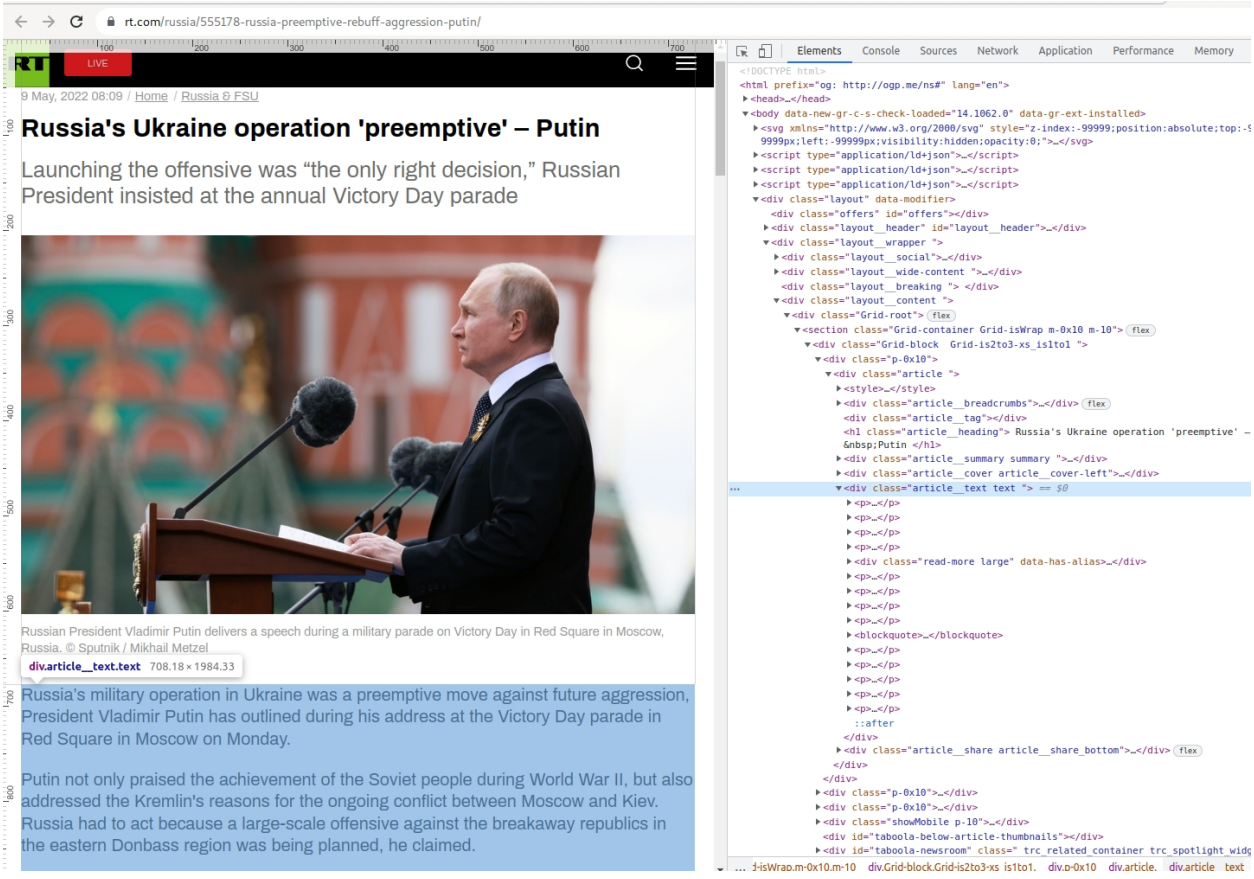


Рисунок 2.2 — Пошук CSS-селекторів в статті сайту rt.com

Підсумовуючи, загальний алгоритм роботи розробленого скрейпера можна описати як послідовність наступних високорівнених операцій:

1) Завантажити файл карти сайту sitemap_2022.xml з сайту rt.com.
2) Десеріалізувати файл карти сайту з формату XML у структури даних в пам’яті. Добути зі структури список всіх посилань на статті.
3) Для кожного посилання на статтю, завантажити HTML-вміст сторінки.
4) Для кожної статті, розібрати HTML-вміст сторінки за допомогою HTML-парсера. Добути заголовок, анотацію та вміст.
5) Зберегти зібрані дані як файл в форматі CSV на диск.

2.2.2 Збирання даних з соціальної мережі Twitter

Twitter — соціальна мережа та мікроблог. В Twitter зареєстрована велика кількість медіа-акаунтів, в тому числі й новинних, зокрема: Bloomberg, BBC, New York Times, The Guardian, CNN; є й російські державні, як-то RT, MFA Russia тощо. Особливість контенту Twitter та доцільність його аналізу розглядалась вище (див. підрозділ 2.1).

Twitter обмежує доступ до свого веб-застосунку автоматичним програмним засобам багатьма способами, тому підхід зі скрепінгом в даному випадку є ускладненим та субоптимальним. Натомість, Twitter надає офіційний API, Twitter API, як частину Twitter Developer Platform [19]. Для доступу до Twitter Developer Platform потрібно зареєструватись та подати запит на надання доступу з поясненням цілей та мотивації розробки чи дослідження, з якими планується використовувати Twitter API. Загалом, процес отримання доступу доволі ускладнений та зайняв кілька днів, враховуючи необхідність листування з техпідтримкою. Отримавши доступ, реєструємо застосунок з унікальним ідентифікатором та зберігаємо наступні ключі: API Key, API Key Secret та Bearer Token. Вони необхідні для виконання запитів до API.

Позитивною стороною збирання даних з Twitter є те, що вже існують бібліотеки для роботи з Twitter API [20] та розробки, які автоматизують поширені задачі взаємодіїї з API, зокрема, процес завантажевання дописів з певних акаунтів. Скористаємось одним з таких інструментів [21] для завантаження дописів від RT, MFA_Russia, BBC World, Bloomberg Politics, The Kyiv Independent.

Twitter API має й суттєве обмеження — можна завантажити лише 3 200 останніх дописів (при тому, що дописи відсортовані по даті створення в порядку спадання) від певного акаунту, тобто таким чином отримати можна лише останні дані. В контексті вирішуваної задачі це не є значною проблемою, адже емпірично було встановлено, що для обраних акаунтів вікно з 3 200 останніх дописів включає період найбільшої активності російської пропаганди, а саме вторгнення РФ до України 24 лютого 2022 року.

	id	created_at	text	is_propaganda
2000	1516348378048782338	2022-04-19 09:30:00+00:00	Zelensky’s statements influenced by what he drinks or sm...	True
1972	1514046239247151104	2022-04-13 01:02:07+00:00	⚡Shmygal: Ukraine begins seed sowing in all regions exc...	False
2516	1517604188724805633	2022-04-22 20:40:08+00:00	The Georgia state administrative hearing to determine if...	False
696	1524117070304096256	2022-05-10 20:00:00+00:00	Kiev intimidated Ukrainians from seeking refuge in Russi...	True
2573	1513109538529894401	2022-04-10 11:00:00+00:00	Kalibr missiles take out Ukrainian military infrastructu...	True
1220	1520591335278858242	2022-05-01 02:30:00+00:00	Russia offers India way of bypassing Western sanctions —...	True
2725	1508359220441272322	2022-03-28 08:23:56+00:00	⚡ No humanitarian evacuations to take place on March 28...	False
610	1524419768136212482	2022-05-11 16:02:49+00:00	⚡Czech Republic recognizes Russia's actions in Ukraine ...	False

Рисунок 2.3 — Фрагмент зібраного набору даних з Twitter

3 ОБРОБКА ДАНИХ. ПОБУДОВА АЛГОРИТМУ КЛАСИФІКАЦІЇ

3.1 Побудова процесу попередньої обробки даних

Для ефективної обробки природної мови, слід провести попередню обробку даних. Процес попередньої обробки даних та мотивація його застосування детально розглядались в попередній частині роботи (див. підрозділ 1.4).

Спершу розглянемо фрагмент набору даних, а саме три випадкових записи та вибірку з двох колонок: текст допису (колонка *text*) та колонку зі значенням булевого типу, що позначає чи є пропагандою (колонка *is_propaganda*). Фрагмент набору даних у оригінальному вигляді зображено в таблиці 3.1.

Таблиця 3.1 — Фрагмент набору даних в оригінальному вигляді

text	is_propaganda
Russia MoD claims that American scientists carried out harmful experiments between 2019 and 2021...	True
Plane damaged beyond repair after hard landing at UK airport https://t.co/gbx5UacKzy	False
RT @RusEmbSriLanka: Here is one of many accounts of the atrocities & crimes against civilians committed by the neo-Nazi Azov Battalion	True

У оригінальному вигляді текст може містити посилання на сторонні ресурси, довільну кількість пробілів та переносів, а також різні спецсимволи, зокрема, також, емодзі (emoji). Емодзі можуть бути корисними для розпізнавання тональності тексту, проте в контексті вирішуваної задачі розглядати семантичне значення емодзі не будемо.

Відфільтруємо емодзі разом зі спецсимволами. Також в процесі токенізуємо текст та проведемо його нормалізацію. Варто відмітити, що в цьому та в подальших етапах токенізація також включає нормалізацію тексту. Це базовий варіант попередньої обробки тексту, після чого з текстом вже можна працювати та класифікувати його програмним шляхом. Результат зображено в таблиці 3.2.

Таблиця 3.2 — Застосування токенізації до тексту

text_clean	is_propaganda
[russia, mod, claim, that, american, scientists, carried, out, harmful, experiments, between, 2019, and, 2021...	True
[plane, damaged, beyond, repair, after, hard, landing, at, uk, airport]	False
[rt, rusembsrilanka, here, is, one, of, many, accounts, of, the, atrocities, crimes, against, civilians, committed, by, the, neo, nazi, azov, battalion]	True

Доволі часто, хоча й не завжди, в задачах обробки природної мови проводять видалення стоп-слів – сполучників, артиклів, окремих літер, числел тощо (див. підрозділ 1.4). Доцільність цієї задачі залежить від конкретної вирішуваної задачі, тематики та іншої специфіки тексту, і ефективність слід перевіряти емпіричрним шляхом, що буде здійснено в подальших розділах. Результат токенізації та видалення стоп-слів на прикладі можна спостерігати в таблиці 3.3.

Таблиця 3.3 — Застосування токенізації та видалення стоп-слів до тексту

text_clean	is_propaganda
[russia, mod, claims, american, scientists, carried, harmful, experiments]	True
[plane, damaged, repair, hard, landing, uk, airport]	False
[rt, rusembsrilanka, accounts, atrocities, crimes, civilians, committed, neo, nazi, azov, battalion]	True

Більш складною технікою попередньої обробки є лематизація, що потребує морфологічного аналізу слова. Цей елемент попередньої обробки дозволяє привести слово до його словникової форми. Для лематизації скористаємось мовною моделлю en_core_web_sm з програмної бібліотеки spaCy. Варто відмітити, що цей процес займає істотно більше часу на обробку, ніж, наприклад, токенізація + нормалізація з попередніх кроків. Результат лематизації зображено в таблиці 3.4.

Таблиця 3.4 — Застосування токенізації та лематизації до тексту

text_clean	is_propaganda
[russia, mod, claim, that, american, scientist, carry, out, harmful, experiment, between, and]	True
[plane, damage, beyond, repair, after, hard, landing, at, uk, airport]	False
[rt, rusembsrilanka, here, be, one, of, many, account, of, the, atrocity, crime, against, civilian, commit, by, the, neo, nazi, azov, battalion]	True

Можливе також і комбінування описаних вище підходів. Розглянемо один можливий варіант комбінування, а саме токенізації, нормалізації, видалення стоп-слів та лематизації слів, результат якого зображено в таблиці 3.5. В результаті отримуємо максимально лаконічне представлення тексту природною мовою, але при цьому втрачається частина семантичного значення речення після застосування всіх операцій, а особливо видалення стоп-слів.

Таблиця 3.5 — Застосування токенізації та видалення стоп-слів до тексту

text_clean	is_propaganda
[russia, mod, claims, american, scientists, carried, harmful, experiments]	True
[plane, damaged, repair, hard, landing, uk, airport]	False
[rt, rusembsrilanka, accounts, atrocities, crimes, civilians, committed, neo, nazi, azov, battalion]	True

Щоб відповісти на питання, яка з наведених схем попередньої обробки даних є найкращою в контексті вирішуваної задачі, потрібно провести тестування з замірами показників ефективності та порівняти їх, що буде розглянуто в наступних підрозділах.

3.2 Застосування моделі Word2vec

Загалом, в задачах обробки природної мови є два способи створити модель представлення слів. Перший спосіб — використати попередньо навчену модель (навчену на великому корпусі текстів, згенерованому з різних джерел, наприклад, з Wikipedia чи соціальних мереж). Другий спосіб — створити власну модель на основі певного корпусу текстів.

Перевагою другого способу є те, що модель тренується на словнику доменної області, що досліджується, та вивчає зв’язки між словами в цій конкретній доменній області. Оскільки доменна область, що досліджується, а саме російська пропаганда за період повномасштабного вторгнення Російської Федерації до України 2022 року доволі специфічна, слід натренувати власну модель класифікації.

Для створення векторної моделі представлення слів скористаємось методом Word2vec, що вже розглядався в попередній частині роботи (див. підрозділ 1.5.4). Цей метод використовує модель машинного навчання, щоб встановити асоціації між словами в великому наборі текстових даних (корпусі). Word2Vec базується на дворівневих нейронних мережах, які розроблені для відтворення контексту слів. Однією з властивостей моделі Word2vec є можливість підказувати схожі по контексту слова.

```
λ Bogdan-Latitude-3410 propaganda-classification → λ git main* → pipenv run python3
Python 3.10.4 (main, Apr 20 2022, 23:20:54) [GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import pprint
>>> import gensim
>>>
>>> model = gensim.models.Word2Vec.load('w2v_model_rtcom')
>>>
>>> pprint.pprint(model.wv.most_similar('zelensky', topn=3))
[('volodymyr', 0.7912309765815735),
 ('tokayev', 0.7768572568893433),
 ('lukashenko', 0.7489634156227112)]
>>>
>>> pprint.pprint(model.wv.most_similar('bandera', topn=5))
[('stepan', 0.9659504294395447),
 ('collaborator', 0.9573060274124146),
 ('oun', 0.9039950370788574),
 ('insignia', 0.9011899828910828),
 ('fascist', 0.8968623280525208)]
>>>
>>> pprint.pprint(model.wv.most_similar('kiev', topn=3))
[('ukrainian', 0.690227746963501),
 ('ukraine', 0.6815356612205505),
 ('demilitarization', 0.6618202924728394)]
>>>
>>> pprint.pprint(model.wv.most_similar(['kiev', 'regime'], topn=3))
[('demilitarization', 0.8030099272727966),
 ('denazification', 0.7971296906471252),
 ('commit', 0.7894409894943237)]
>>>
```

Рисунок 3.1 — Пошук схожих за контекстом слів за допомогою Word2vec

Наприклад, з натренованої на пропагандистській вибірці моделі Word2vec можна отримати наступні результати пошуку схожих по контексту слів (див. рис. 3.1). Метод most_*similar* повертає список з максимум *topn* кортежів форми *(слово, індекс подібності),* відсортований по значенню індексу подібності в порядку спадання.

Також за допомогою моделі є можливість порівнювати різні слова та отримати індекс контекстної схожості слів — дійсне число від 0 до 1. Зокрема, з прикладу (рис. 3.2) можна спостерігати, що слова “ukrainian” та “nazi” мають значно більший індекс схожості, ніж слова “russian” та “nazi”, що цілком характерно для російської пропаганди. Також слова “zelensky” та “ukrainian” є більш контекстно сумісними, ніж “zelensky” та “british”, що є закономірним виходячи з того, на яких даних натренована модель.

```
>>> model.wv.similarity('zelensky', 'russian')
0.19594984
>>> model.wv.similarity('zelensky', 'british')
0.051697604
>>> model.wv.similarity('zelensky', 'ukrainian')
0.5411352
>>>
>>>
>>> model.wv.similarity('russian', 'nazi')
0.096823946
>>> model.wv.similarity('ukrainian', 'nazi')
0.47342068
>>>
```

Рисунок 3.2 — Порівняння контекстної схожості слів за допомогою Word2vec

Серед інших корисних методів варто відмітити метод *doesnt_match,* що дозволяє з переданого в якості параметра списку слів знайти таке слово, що не відповідає іншим, виходячи з контексту, на якому була натренована модель Word2vec. Приклад (рис. 3.3) демонструє закономірність — в зразках пропаганди, на який була натренована модель, систематично уникається вживання слів “war” та “invasion” разом зі словами “russia” та “ukraine”.

```
>>>
>>> model.wv.doesnt_match(['alexander', 'lukashenko', 'biden'])
'biden'
>>> model.wv.doesnt_match(['russia', 'war', 'ukraine'])
'war'
>>> model.wv.doesnt_match(['russia', 'invasion', 'ukraine'])
'invasion'
>>> model.wv.doesnt_match(['russia', 'war', 'special', 'operation' 'ukraine'])
'war'
>>>
>>>
```

Рисунок 3.3 — Пошук слова зі списку, що не підходить до решти слів

3.3 Класифікація за допомогою методу Random Forest

Для кожного слова, отриманого після всіх кроків попередньої обробки, створюється його векторне відображення за допомогою моделі Word2vec. Далі, для кожного окремого тексту, нам потрібно отримати один результуючий вектор зі всіх векторних представлень слів цього тексту. Для цього застосуємо векторну операцію середнього арифметичного до всіх векторів тексту. Результатом такої операції є певний вектор для кожного тексту і мітка для кожного з векторів. Далі ці значення усереднених векторів будуть використовуватись для навчання класифікатора.

Існує велика кількість класичних алгоритмів класифікації, що є усталеними та вже давно використовуються для класифікації текстів — зокрема, регресійні моделі класифікації, дерева прийняття рішень, Баєсів класифікатор, метод опорних векторів тощо. З ростом обчислювальних потужностей дедалі популярнішими стають методи, що є більш ресурсозатратними та потребують більше обчислень — зокрема, це метод класифікації Random Forest.

Алгоритм Random Forest використовує модель дерева прийняття рішень CART — Classification and Regression Tree. Створюється модель з випадково згенерованими параметрами та початковими значеннями. Функція класифікації Random Forest — це функція зі всіх передбачень кожного з дерев класифікації. Класифікатор Random Forest дозволяє отримати кращі результати, ніж звичайний метод CART, особливо тоді, коли потрібно класифікувати великий набір даних та/або коли є велика кількість змінних.

Класифікатор Random Forest тренується на основі векторів, отриманих в результаті застосування операції середнього арифметичного до векторів, що були створені в результаті навчання моделі вкладення слів Word2vec. Далі вже натренована модель Word2vec використовується для класифікації тексту.

Повний процес як послідовність кроків можна описати наступним чином (рис. 3.4):

1) Отримуємо тренувальний набір даних.

2) Проводимо попередню обробку тексту, застосовуючи токенізацію, лематизацію та видалення стоп-слів.

3) Тренуємо модель Word2vec, створюємо векторні представлення слів для тексту. На основі отриманих векторних представлень слів створюємо агрегований вектор для кожного тексту.

4) На основі агрегованих векторів з міткою тренуємо класифікатор Random Forest. Зберігаємо натреновану модель. Використовуємо вже натренований класифікатор Random Forest для класифікації текстів.

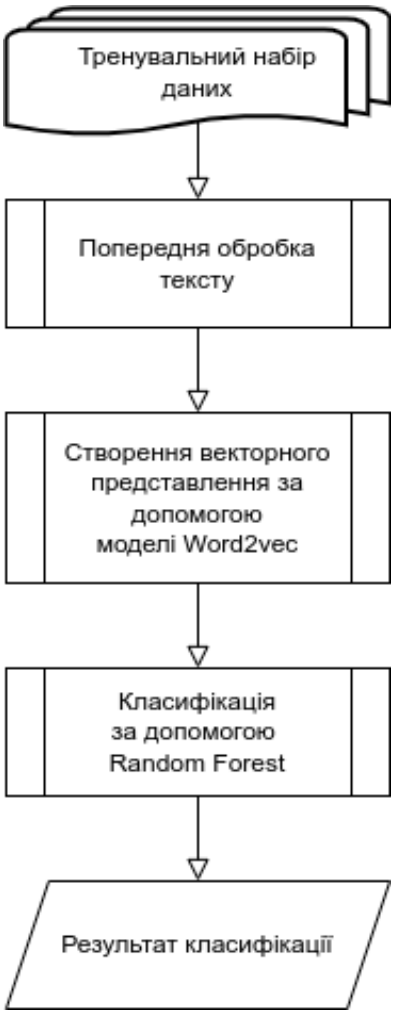


Рисунок 3.4 — Повний процес класифікації тексту

Продемонструємо покроково повний процес класифікації на прикладі одного з повідомлень. Використовуємо вже навчену на тренувальному наборі даних за попередньо описаною схемою модель Random Forest. Метод predict_proba класу RandomForestClassifier дозволяє отримати результат класифікації як ймовірність належності до кожного з двох класів (пропаганда та не пропаганда) як дійсне число від 0 до 1. На прикладі це відповідно [0,19 та 0,81], що означає, що повідомлення було класифіковане як пропаганда. Метод predict дає відповідь як значення булевого типу — True або False (True в даному випадку). Весь процес класифікації фрагменту тексту можна спостерігати на рисунку нижче (рис. 3.5).

```
>>> message = '#Zakharova: The West continues to provide the Kiev regime with lethal weaponry'
>>>
>>>
>>> preprocessed_message = preprocess_message_advanced(message)
>>>
>>> preprocessed_message
[' ', 'zakharova', 'west', 'continue', 'provide', 'kiev', 'regime', 'lethal', 'weaponry']
>>>
>>>
>>> agg_vectors = make_agg_vectors_from_text([preprocessed_message])
>>>
>>> agg_vectors
array([[[-0.02854842,  0.81745815,  0.19227745, ..., -1.4593288 ,
          0.09767188,  0.3459736 ],
        [-0.09611517,  0.49588934,  0.05124994, ..., -1.7117958 ,
         -0.22477023,  0.9998988 ],
        [-0.10238221,  0.44090638,  0.14987522, ..., -0.75037044,
          0.24358237,  0.23556803],
        ...,
        [-0.03686094,  0.25464115,  0.09207091, ..., -0.41909623,
          0.13931987,  0.10159288],
        [-0.01410565,  0.07108819,  0.02445323, ..., -0.09642744,
          0.04340363,  0.02248302],
        [-0.0092394 ,  0.06272655,  0.03612397, ..., -0.08579122,
          0.05192475,  0.01875964]]], dtype=float32)
>>>
>>>
>>> avg_vectors = make_avg_vectors(agg_vectors)
>>>
>>> avg_vectors
[array([-0.04909526,  0.38824722,  0.12162883, ..., -0.7259614 ,
        0.13593866,  0.21451782], dtype=float32)]
>>>
>>> random_forest.predict_proba(avg_vectors)
array([[0.19, 0.81]])
>>>
>>>
>>>
>>> random_forest.predict(avg_vectors)
array([ True])
>>>
```

Рисунок 3.5 — Приклад повного процесу класифікації

3.4 Оцінка ефективності алгоритму класифікації

3.4.1 Метрики ефективності алгоритмів класифікації

Спершу розглянемо метрики, за допомогою яких будемо оцінювати ефективність роботи алгоритму класифікації. *Accuracy* — проста, інтуїтивно зрозуміла метрика, що є долею вірних відповідей алгоритму серед всіх об’єктів вхідних даних. *Precision* — доля об’єктів, названих алгоритмом класифікації як ті що належать певному класу X, і що в дійсності є об’єктами класу X. Precision є показником здатності алгоритму відрізняти певний клас даних від іншого.

Recall — метрика, що відображає долю об’єктів класу X зі всіх об’єктів класу X, які виявив алгоритм. Recall є відображенням здатності алгоритму класифікації виявляти клас даних в цілому. На відміну від простої метрики Accuracy, метрики Recall та Precision не залежать від співвідношення класів і тому є особливо корисними, якщо вибірки незбалансовані. Зазвичай для вирішення практичних задач потрібно знайти оптимальний баланс між метриками Recall та Accuracy [22].

Метрики ефективності зазвичай описують в термінах помилок класифікації. Їх можна визначити, скориставшись матрицею помилок, що зображена в таблиці 3.6. Матриця помилок — це таблиця, що визначає ефективність певного алгоритму класифікації.

Таблиця 3.6 — Матриця помилок

	Передбачено 0	Передбачено 1
Фактично 0	TN	FP
Фактично 1	FN	TP

У таблиці 3.6 позначено:

– TP — частина істинно позитивних спрацювань алгоритму;
– TN — частина істинно негативних спрацювань;
– FP — частина невірних позитивних спрацювань;
– FN — частина невірних негативних спрацювань;

Виходячи з попереднього визначення показників помилок класифікації, математично метрики ефективності можна описати як такі рівняння:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

$$precision = \frac{TP}{TP + FP} \tag{3.2}$$

$$recall = \frac{TP}{TP + FN} \tag{3.3}$$

3.4.2 Перевірка ефективності алгоритму класифікації в залежності від схеми попередньої обробки даних

Порівняємо ефективність роботи алгоритму класифікації в залежності від виду попередньої обробки даних.

Оцінюватись будуть наступні види попередньої обробки даних:

1) Токенізація.

2) Токенізація + видалення стоп-слів.

3) Токенізація + лематизація.

4) Токенізація + видалення стоп-слів + лематизація.

Опишемо методику оцінювання ефективності як послідовність кроків. Спершу до пам’яті завантажується попередньо сформований набір даних, що включає тексти обох категорій. Набір даних рандомізується, тобто впорядковується у випадковому порядку. Далі набір даних розділяється на дві підмножини: тренувальні дані та тестові дані, таким чином, що частка тестових даних складає $\frac{1}{5}$ від розміру всього набору даних. В кожній отриманій категорії даних є дані обох класів — пропаганда та не пропаганда.

На тренувальних даних проводиться тренування моделі, а на тестових даних буде замірятись ефективність роботи вже натренованої моделі. Для кожного виду попередньої обробки даних, проводиться п’ять замірів ефективності, з яких береться середнє арифметичне значення. Результати замірів наведені в таблиці 3.7.

Таблиця 3.7 — Порівняння ефективності способів попередньої обробки

	Precision	**Recall**	**Accuracy**
1	0,786	0,674	0,743
2	0,762	0,701	0,737
3	0,793	0,696	0,76
4	0,767	0,718	0,749

Найкраще за метрикою Recall себе показав підхід з застосуванням токенізації, видалення слів та лематизації, з незначним погіршенням метрик Precision та Accuracy відносно підходу з застосуванням лише токенізації та лематизації.

Погіршення метрик Accuracy та Precision з видаленням стоп-слів можна пояснити тим, що разом із видаленням стоп-слів, втрачається і частина семантичного значення певного речення чи твердження. Отже, в випадку застосування видалення стоп-слів ми збільшуємо здатність алгоритму розпізнавати пропаганду в текстах, але натомість росте і кількість хибно-позитивних спрацювань (алгоритм частіше класифікує фрагмент тексту як пропагандистський, хоча він не є таким).

Графік кривої метрик precision-recall для четвертої схеми попередньої обробки даних (рис. 3.6) дає змогу оцінити компроміс між цими двома метриками для різних порогових значень. Метрика Average Precision (AP) — це середнє зважене значення всіх точностей (значень Precision), отриманих на різних порогових значеннях, що дає можливість швидко оцінити ефективність класифікатора.

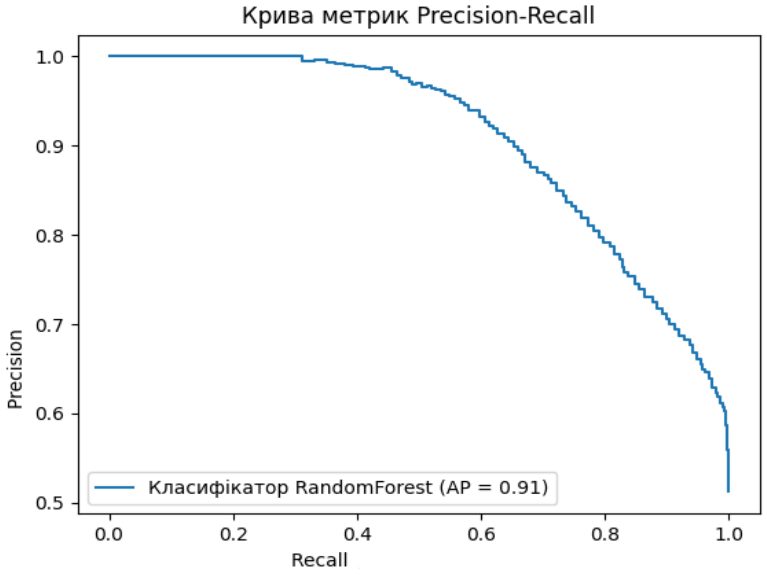


Рисунок 3.6 — Графічне зображення метрик Precision-Recall

Вибір між двома останніми підходами до попередньої обробки даних більшою мірою залежить від того, яких цілей ми хочемо досягти: виявити більше пропаганди шляхом збільшення хибно-позитивних спрацювань, чи зупинитись на менших показниках здатності алгоритму виявляти пропаганду, натомість отримавши менше хибно-позитивних результатів. В контексті вирішуваної задачі раціональним є зменшення саме хибно-позитивних спрацювань, тобто звинувачень в наявності пропаганди в тексті, коли в дійсності її немає. Отже, зупинимось на третьому підході.

3.5 Виявлення пропаганди в текстах українською мовою

В попередніх підрозділах розроблений алгоритм класифікації оперував текстами саме англійською мовою. Розглянемо також можливість класифікації пропаганди даною моделлю на текстах українською мовою.

3.5.1 Формування набору даних українською мовою

Для тренування та тестування ефективності моделі класифікації потрібен набір даних для тренування та тестування ефективності. Відповідно, для тестування моделі виявлення пропаганди в текстовій інформації українською мовою потрібен набір даних саме українською мовою. Створення повністю нового набору даних є доволі ресурсоємним процесом та відповідно займе багато часу. Більш раціональним є створення набору даних українською мовою на основі вже попредньо створеного набору даних англійською.

Можна виділити наступні методи перекладу набору даних:

1) Ручний переклад тексту в наборі даних
2) Напівавтоматичний переклад тексту в наборі даних за допомогою сервісів автоматичного перекладу, як-то Google Translate.
3) Створення програмного засобу (сценарію, або ж скрипта) для повністю автоматичного перекладу тексту в наборі даних.

Очевидно, що найбільше переваг для вирішення задачі надає саме третій спосіб з повністю автоматичним методом перекладу тексту, отже він є найбільш пріоритетним для розгляду. Розглянемо методи та засоби, за допомогою яких ми можемо автоматично перекласти текст в наборі даних. Компанія Google надає доступ до Google Translate як сервіс [23]. Існує програмна бібліотека googletrans для мови програмування Python [24], що звертається до API сервісу Google Translate та автоматизує рутинні операції з ним (рис. 3.7).

```
λ Bogdan-Latitude-3410 propaganda-classification → λ git main* → pipenv run python3
Python 3.10.4 (main, Apr 20 2022, 23:20:54) [GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> import googletrans
>>>
>>> translator = googletrans.Translator(raise_exception=True)
>>>
>>> translation = translator.translate('Ой у лузі червона калина похилилася', 'en')
>>>
>>> translation.text
'Oh, in the meadow the red viburnum bent down'
>>>
```

Рисунок 3.7 — Приклад програмного перекладу одиночної фрази

Реалізуємо скрипт мовою програмування Python, який буде брати існуючий набір даних з текстами англійською мовою та перекладати їх українською. Скрипт працює наступним чином: відкривається два файли — twitter_dataset.csv та twitter_dataset_translated.csv відповідно в режимі читання та запису. Вміст першого файлу зчитується та обробляється по одному рядку.

Для кожного рядка, за допомогою бібліотеки googletrans формується запит до Google Translate API зі значенням колонки з текстовими даними та ідентифікатором мови, на яку потрібно перекласти текст — «uk». Запит відправляється на сервер Google, потік очікує відповіді від сервера та в разі отримання відповіді з успішним-статус-кодом записує до другого файлу рядок з перекладеним текстом, отриманим в результаті попередньої операції.

Запис до вихідного файлу виконується з буферизацією, тобто спершу рядки записуються до буферу певного розміру в ОЗП, після чого дані з буферу періодично записуються на диск по його заповненню. Вказана послідовність дій застосовується до всіх рядків з файлу 1. Створений набір даних (рис. 3.8) використовується в подальших кроках для навчання моделі та оцінки її ефективності.

id	created_at	text	is_propaganda
1511225054876549121	2022-04-05 06:11:44+00:00	Британські батько і син загинули, інші отримали пораненн...	False
1510889918066614280	2022-04-04 08:00:02+00:00	Росія закликала до засідання Ради Безпеки ООН, оскільки ...	True
1514265090065568260	2022-04-13 15:35:01+00:00	#Захарова: Польська влада незаконно захопила російське д...	True
1522475271223398400	2022-05-06 07:16:05+00:00	«Це емоційний момент»Карін Жан-П'єр стала першим темношк...	False
1509379964088455175	2022-03-31 04:00:01+00:00	Росія висловила стурбованість у зв'язку з очевидним вико...	True
1523633145958080518	2022-05-09 11:57:04+00:00	Вартість біткойна впала на 50% з піку листопада	False
1522242324574740483	2022-05-05 15:50:26+00:00	Захід воює з дезінформацією чи інакомисленням? У міру зр...	True
1509235288551350285	2022-03-30 18:25:07+00:00	Будинки та підприємства на східному узбережжі Австралії ...	False

Рисунок 3.8 — Фрагмент набору даних, перекладеного українською мовою

3.5.2 Адаптація моделі виявлення пропаганди для роботи з українською мовою

Розглянемо компоненти описаної в попередніх підрозділах моделі виявлення пропаганди та можливі зміни, які можуть знадобитись для них. Модель створення векторних вкладень Word2vec є повністю мовно-незалежною, оскільки оперує векторним представленням слів. Тому можна застосувати модель Word2vec для отримання векторних представлень слів будь-якої мови в якій, власне, є слова. Відповідно, з українською мовою на цьому етапі не має виникнути проблем. Класифікатор Random Forest також не має обмежень до застосування мови, так як працює з векторами, отриманими в результаті застосування векторної операції середнього арифметичного до векторів. Отже, єдиною частиною, що залежить від мови, залишається процес попередньої обробки текстових даних.

Розглянемо процес попередньої обробки текстових даних та яких змін він може потребувати для проведення операцій над українською мовою. Щодо простих методів попередньої обробки (як токенізація та нормалізація) перехід на іншу мову є доволі тривіальним. Достатньо замінити регулярний вираз, що фільтром символів, на інший, який не буде виключати кириличні літери. Щоб виконати операцію видалення стоп-слів потрібно мати, власне, словник стоп-слів українською мовою, що потребує більше зусиль для адаптації алгоритму. Найскладнішим серед використаних етапів попередньої обробки текстових даних є лематизація, адже для цієї операції потрібно провести морфологічний аналіз слів, а тому потрібно мати мовну модель українською мовою.

Для побудови процесу попередньої обробки даних застосовувалась бібліотека spaCy. За допомогою засобів spaCy є можливість завантажувати мовні моделі для різних мов, і відповідно існують модулі з вже натренованими моделями. На жаль, для української мови готового модуля для підтримки української ще немає. Проте, є експериментальна можливість використання багатомовної моделі MultiLanguage, також відомої як xx_ent_wiki_sm. Ця модель може застосовуватись до багатьох різних мов, хоча є менш точною. Модель MultiLanguage вже успішно застосовувалась дослідниками для аналізу східноєвропейських мов, зокрема, польської, хорватської та чеської [25].

В мовній моделі MultiLanguage для аналізу української використовується морфологічний аналізатор PyMorhy2 з однойменної програмної бібліотеки для Python [26]. Аналізатор PyMorphy2 проєктувався для морфологічного розбору російської мови, але може застосовуватись також і для української.

3.5.3 Заміри ефективності на даних українською мовою

Методика проведення дослідження та застосовані метрики детально описані в попередній частині розділу (див. підрозділ 3.4.1, 3.4.2). Натомість, лише певною мірою були адаптовані кроки попередньої обробки для української мови, хоча принципово методики є однаковими.

Повторно наведемо застосовані види попередньої обробки даних:

1) Токенізація.

2) Токенізація + видалення стоп-слів.

3) Токенізація + лематизація.

4) Токенізація + видалення стоп-слів + лематизація.

Отримані результати замірів ефективності (табл. 3.8) свідчать про те, що описана модель є еквівалентно ефективною для виявлення пропаганди в текстах українською мовою, як і англійською.

Таблиця 3.8 — Порівняння ефективності способів попередньої обробки даних українською мовою

	Precision	**Recall**	**Accuracy**
1	0,795	0,641	0,732
2	0,816	0,691	0,762
3	0,845	0,718	0,791
4	0,816	0,732	0,783

Можна помітити, що показники деяких параметрів є незначною мірою кращими, порівняно з результатами на наборі текстів англійською мовою — зокрема, кращим є показники метрик Precision, що свідчить про те, що порівняно з результатами тестів на англійській мові, зменшилась частка хибно-позитивних спрацювань.

Проте спостерігається погіршення показника Recall, з чого можна зробити висновок, що загалом здатність моделі розпізнавати пропаганду в українській є меншою, аніж в англійській.

В цілому, також слід враховувати фактор автоматичного перекладу набору даних, оскільки завдяки машинному перекладу іноді може втрачатись певна емоційна або ж семантична складова речення. Щоб оцінити ефективність моделі загалом, скористаємось графіком кривої Precision-Recall (рис. 3.9). Можна помітити, що значення Average Precision (AP) є незначною мірою гіршим відносно такої ж метрики, отриманої на класифікації даних англійською мовою (0,88 проти 0,91).

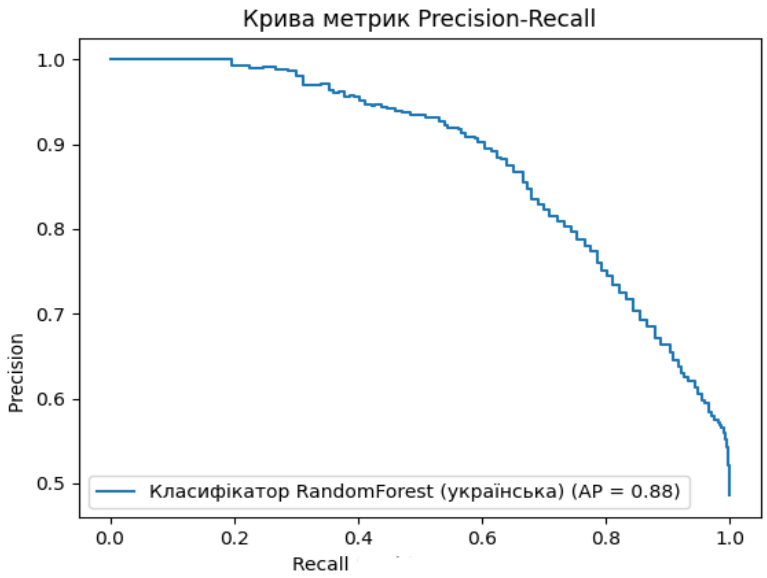


Рисунок 3.9 — Графічне зображення метрик Precision-Recall, отриманих у випадку класифікації даних українською мовою.

4. ПРОЄКТУВАННЯ ПРОГРАМНОГО ПРОДУКТУ

4.1 Аналіз вимог до системи

4.1.1 Веб-застосунок

Розглянемо вимоги до системи, щоб створити план подальшої розробки системи. По-перше, як основний варіант використання планується власний клієнтський веб-застосунок для того, щоб користувачі могли аналізувати контент що зустрічають в мережі Інтернет та бути проінформованими про наявність пропаганди в ньому. Окрім цього, на головній сторінці відображатиметься інформація про останні проаналізовані записи та їх результати, а також статистика використання сервісу.

Основний сценарій взаємодії користувача з веб-додатком можна описати як послідовність дій:

1) Користувач відкриває сторінку веб-застосунку.
2) Користувач копіює довільний текст з довільного ресурсу в буфер обміну.
3) Користувач вставляє текст з буферу обміну в текстове поле на головній сторінці веб-застосунку.
4) Користувач натискає кнопку “Проаналізувати” та отримує у відповідь результат у вигляді індексу пропаганди – дійсне число від 0 до 1.

Веб-додаток має бути створений за моделлю SPA та виконувати HTTP AJAX запити з клієнту (бразузера користувача) до API системи. Окрім популярності підходу SPA та зручності в розробці та підтримці, такий підхід викликаний необхідністю уніфікації API, що розглядається нижче.

4.1.2 Надання послуг за моделлю SaaS

Перспективним напрямком для подальшого розвитку запропонованої системи як продукту є надання API виявлення пропаганди як сервісу за моделлю SaaS (Software as a Service).

За такої моделі розробник програми розробляє API, розгортає її на сервері, якими керує сам та надає доступ клієнтам до неї. Розгонувши систему в хмарному сервісі, можна надавати доступ до неї іншим застосункам, які бажатимуть застосувати розроблену модель класифікації у зручний спосіб [27]. Так, наприклад, сервіс зможуть використовувати веб-сайти та додатки для фільтрації новинної стрічки або ж, можливо, сторонній розробник зможе використати наданий API для створення розширення для браузера, що фільтрує контент на певних ресурсах для користувача.

На початковому етапі для просування продукту планується надання послуг SaaS безкоштовно, але з обмеженням використання для кожного користувача. Далі, в залежності від використання продукту, та запитів клієнтів, можливий перехід на платну модель для комерційних користувачів.

Підсумуючи, виділимо два перспективних напрямки розробки:

– Розробка власного клієнтського веб-застосунку для надання користувачам можливості аналізувати власні тексти, а також для демонстрації можливостей продукту.

– Розробка API для надання послуг за моделлю SaaS.

4.2 Архітектура серверної частини системи

Виходячи з функціональних вимог до системи, описаних вище (див. підпункт 4.1), потрібно розробити серверну частину, що відповідатиме цим вимогам. Опишемо технічні вимоги до системи.

Технічні вимоги до системи:

– Можливість гнучко масштабуватись (scalability) та відповідати на зміни навантаження, оскільки кількість клієнтів API та кількість запитів від основного веб-застосунку може змінюватись [28, с.145].

– Повинна мати високу доступність (high availability) [28, с.145].

– Повинна мати зручний API, що може бути використаний багатьма кінцевими користувачами, зокрема веб-додатками, мобільними додатками, серверними додатками та сервісами.

– Має відповідати сучасним стандартам розробки, бути зручною для розширення та подальшої підтримки.

4.2.1 Мікросервісна архітектура системи. Міжсервісна взаємодія

Маючи технічні вимоги до системи, перейдемо до проєктування загальної серверної архітектури системи.

Оскільки нам потрібна можливість гнучкого масштабування, оберемо мікросервісну архітектуру системи. На початковому етапі виділимо 2 сервіси: головний сервіс, що також виконує функцію програмного API Gateway [29] та сервіс виявлення (класифікації) пропаганди.

Розділення сервісів дозволить гнучко масштабувати навантаження на систему. Це важливо, оскільки задачі сервісу виявлення пропаганди є більшою мірою CPU-bound через використання технік обробки природної мови, машинного навчання та класифікації, в той час як задачі головного сервісу є I/O-bound. Це два різних типи навантаження, що мають різні бажані вимоги до апаратних характеристик обчислювальних ресурсів, на яких вони будуть виконуватись. Так, CPU-bound задачі для ефективної роботи вимагають більшої обчислювальної потужності від ЦП, ніж I/O-bound задачі. Відповідно, для кожного сервісу можна буде підібрати належні обчислювальні ресурси для його ефективної роботи.

Розділення системи на мікросервіси також дозволяє інкапсулювати та ізолювати певні специфічні технології в рамках одного сервісу. Наприклад, для моделі класифікації потрібні наступні специфічні програмні бібліотеки: numpy, pandas, scikit-learn, spaCy, в той час як головному сервісу для обробки запитів вони не потрібні. Ізоляція дозволить зменшити розмір та складність кожного сервісу, що позитивно відобразиться на витратах на їх підтримку — як на обчислювальних ресурсах, так і на витратах програмних інженерів на розробку та підтримку системи.

Слід відмітити, що в подальшому з еволюцією системи кількість мікросервісів та їх призначення можуть змінюватись в залежності від зміни функціональних вимог до системи, проте на поточному етапі розвитку така гранулярність сервісів є цілком достатньою.

Для міжсервісної взаємодії використаємо технологію gRPC. Це рішення для побудови ефективної міжсервісної взаємодії, розроблене компанією Google. Технологія gRPC використовує протокол HTTP/2 в якості транспорту та протокол Protocol Buffers для опису структури повідомлень. Протокол є типізованим (потрібно явно вказувати структуру та тип даних, якими обмінюються сервіси) та бінарним, що дозволяє мінімізувати об’єм міжсервісного трафіку [28, с.135]. Для gRPC існують програмні бібліотеки для багатьох мов програмування, зокрема, і для мови програмування Python.

4.2.2 Збереження даних

Системі також потрібна база даних (БД) та СКБД для керування нею. Для того щоб обрати конкретне програмне рішення для БД, розглянемо характер даних, що буде зберігатись в ній. Дані, що будуть зберігатись в БД включають статистику анілізу текстів та останні проаналізовані тексти. В перспективі також можлива генерація підбірок та новинних стрічок з проаналізованих текстів.

Потрібно відмітити, що природа даних системи є нереляційною та немає вимоги щодо наявності жорстких гарантій цілісності даних що будуть зберігатись, адже певні порушення цілісності можуть вплинути лише на відображення статистики та історичні дані останніх проаналізованих текстів, і, в перспективі, новинну стрічку.

Оберемо найбільш потрібні характеристики БД, користуючись теоремою CAP, також відомої як теорема Брюера [28, с.336], яка стверджує, що з трьох ключових характеристик сховища даних можна обрати лише дві. Характеристики системи за теоремою CAP:

- Consistensy — консистентність;
- Availability — доступність;
- Partition tolerance — стійкість до розподілення;

Оскільки кількість даних що зберігається може суттєво зростати, оберемо стійкість до розподілення як необхідну характеристику. Залишається вибір між доступністю та узгодженістю даних (AP чи CP клас системи), що не є принциповим. Тому також врахуємо природу даних, що є нереляційною, та зупинимось на документо-орієнтованій БД MongoDB що належить до класу CP, оскільки вона є документо-орієнтованою та підтримує гнучку схему даних. Також MongoDB є достатньо популярним рішенням з відкритим програмним кодом, що має полегшити подальшу її підтримку.

Для прискорення доступу до списку останніх записів та статистики скористаємось кешуванням. Також це зменшить навантаження на читання з бази даних. Для кешування використаємо програмне забезпечення Memcached, що зберігає дані для швидкого доступу до них в ОЗП. Також Memcached можна застосовувати для збереження сесій користувачів.

4.2.3 Загальна архітектура системи

Розглянемо весь шлях, що проходить запит до системи на прикладі запиту на аналіз певного тексту:

1) Маючи доменне ім’я ресурсу, клієнт отримує IP-адресу вхідної точки системи, а саме балансувальника навантаження за допомогою DNS-запиту.

2) Клієнт встановлює TCP з’єднання з балансувальником навантаження за допомогою IP-адреси та порту 443. Відправляє запит аналізу по протоколу HTTPS.

3) Запит маршрутизується на один із серверних додатків, що обслуговують запити, кількість яких може динамічно змінюватись.

4) Серверний додаток формує запит за протоколом gRPC/Protocol Buffers та надсилає його до сервісу класифікації.

5) Запит gRPC маршрутизується на один з серверних додатків сервісу класифікації, кількість яких може динамічно змінюватись.

6) Сервіс проводить аналіз тексту за допомогою розглянутої попередньо моделі, формує та відправляє відповідь.

7) Отримавши відповідь від сервісу класифікації, проводиться оновлення даних в СКБД MongoDB, зокрема, проаналізованих записів та статистики. Формується відповідь на початковий HTTP запит та відправляється клієнту.

Загальна архітектура спроєктованої системи, її основні компоненти та зв’язки між ними зображені на рисунку нижче (рис. 4.1).

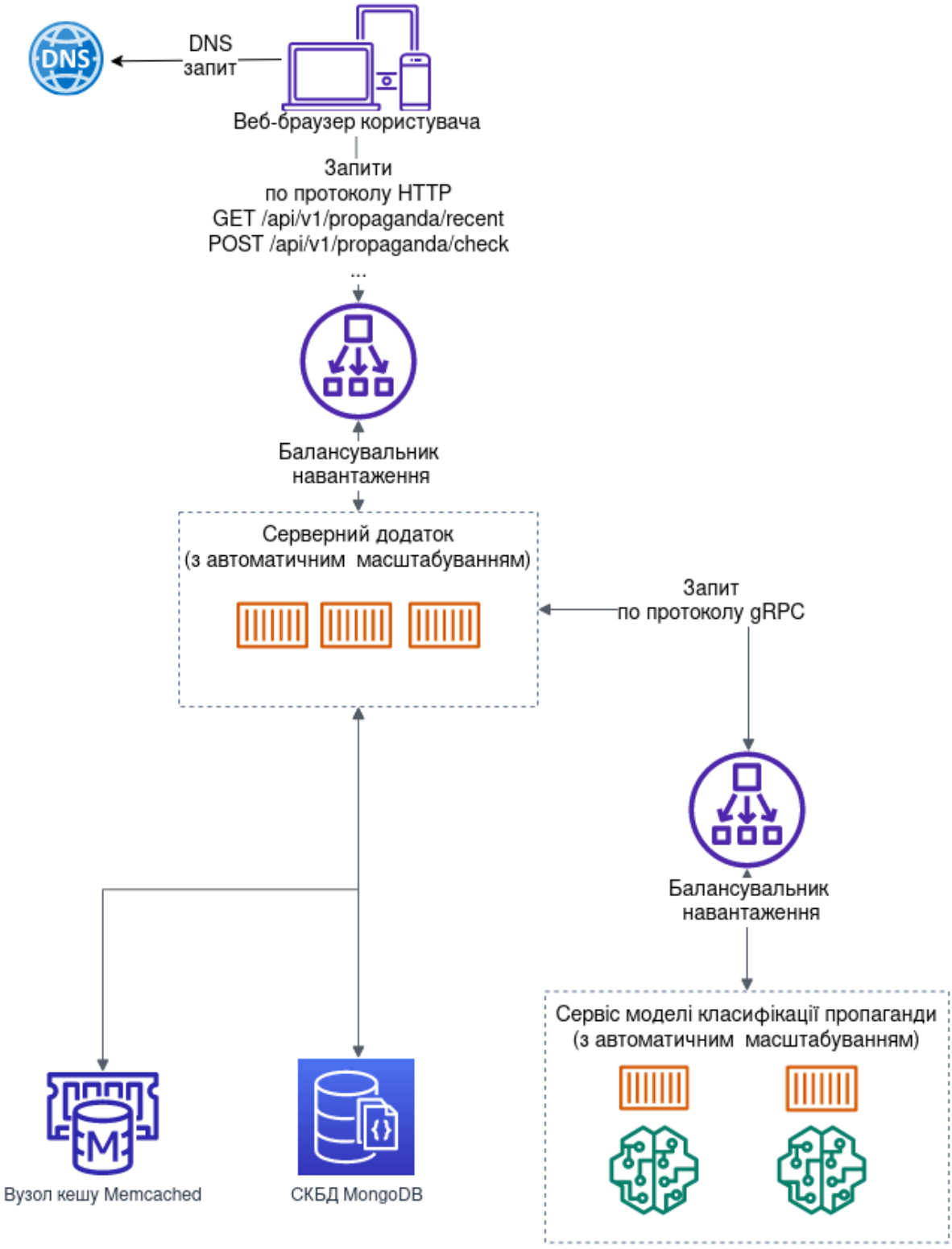


Рисунок 4.1 – Загальна архітектура запропонованої системи

4.3 Обґрунтування вибору мови програмування та програмних бібліотек

В якості основної мови програмування для серверної частини системи було обрано Python. Для побудови моделі виявлення пропаганди вже було попередньо використано бібліотеки numpy, pandas, scikit-learn та spaCy. Для розробки серверного додатка використовується фреймворк Flask.

Головна мотивація використовувати Python пов’язана з прагненням уніфікувати інструменти, засоби та середовище розробки, оскільки модель виявлення пропаганди вже побудована з використанням Python та наявних бібліотек та засобів. Уніфікація інструментів дозволяє зменшити загальну ентропію, складність системи, спростити підтримку системи та зменшити поріг входу при потребі залучення інших інженерів до розробки системи. Також, варто відмітити, що Python добре підходить для веб-розробки, а саме – для розробки серверних застосунків, тому і якість розробки не постраждає.

Серед недоліків Python можна відмітити порівняно низьку швидкодію в CPU-bound задачах – обчислювально-складних задачах, основне навантаження в яких є пов’язане саме з виконанням обчислень [30]. Проте, значна доля роботи системи є саме I/O-bound задачею, тобто основна робота пов’язана з введенням-виведенням даних [30]. CPU-bound частина системи – це модель класифікації, але й тут в засобах та бібліотеках, що використовуються, їх розробниками проведено оптимізації щодо підвищення швидкодії в Python. Так, наприклад, бібліотеки scikit-learn та spaCy для підвищення швидкодії використовують Cython, діалект Python, що генерує код мовою C [26, 31]. Бібліотека numpy має велику кількість коду, написаного мовою програмування C, до якого створені прив’язки для використання в мові програмування Python [32]. Всі перелічені вище заходи загалом дозволяють досягти достатньої швидкодії при обробці великої кількості даних та проведенні обчислень над ними.

4.4 Прототип API

Спроєктуємо прототип API системи. Інтерфейс системи має бути доступним для використання як із серверних додатків, так і з браузера. Отже, спроєктуємо API з використанням протоколу HTTP.

Розглянемо наступні ресурси API:

– api/v1/propaganda/check

– /api/v1/propaganda/check_batch

Ресурс POST /api/v1/propaganda/check дозволяє перевірити один текст. Ресурс приймає тіло запиту в форматі JSON, в якому міститься поле з текстом для аналізу. Використання методу POST викликано обмеженням методу GET, оскільки всі дані, що передаються в метод GET, мають передавати в URL запиту як параметри запиту. URL запиту має обмеження – максимум 2048 символів. Якщо використовувати метод GET, на довгих фрагментах тексту можуть виникнути проблеми з коректною передачею даних, тому використовується POST. Приклад виконання запиту в програмному засобі Insomnia зображено нижче (рис. 4.2). У відповідь на запит повертаються результати аналізу в форматі JSON в об’єкті analysisData, а саме:

– Відповідь булевого типу, що відповідає, чи містить текст пропаганду – поле isPropaganda.

– Значення індексу пропаганди, ймовірність того, що текст є пропагандистським – дійсне число від 0 до 1 – поле propagandaProbability.

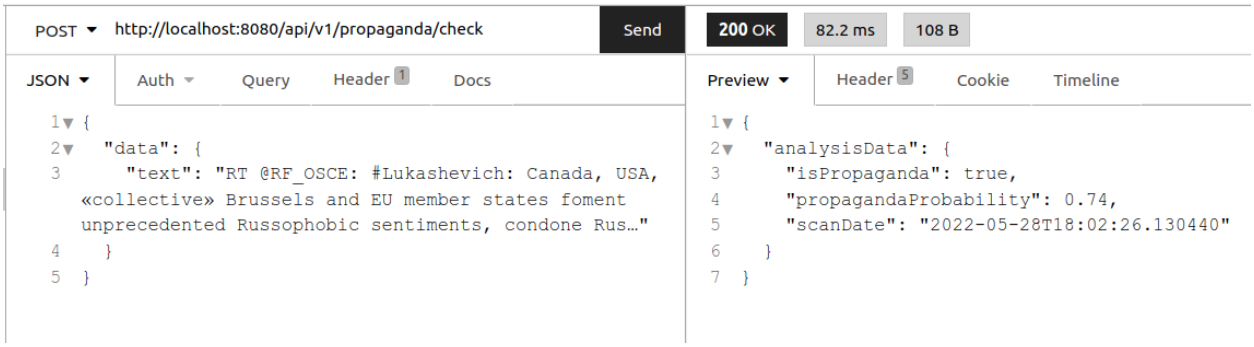


Рисунок 4.2 – Запит на перевірку до ресурсу /api/v1/propaganda/check

Ресурс POST /api/v1/propaganda/check_batch дозволяє перевірити декілька текстів за раз. Ресурс приймає список об’єктів з текстами. Відповідь має схожий формат з /api/v1/propaganda/check, проте адаптований для повернення багатьох відповідей за раз. Крім результатів аналізу, для кожного об’єкту результату повертається поле status, яке приймає значення “OK” або “ERROR”. Це поле дозволяє зрозуміти клієнту, що сталась помилка під час обробки конкретного тексту, та отримати результати для всіх інших текстів. Приклад виконання запиту в програмному засобі Insomnia зображено нижче (рис. 4.3).

Рисунок 4.3 – Запит на перевірку до ресурсу /api/v1/propaganda/check_batch

ВИСНОВКИ

Результатом даної роботи є реалізована модель виявлення пропаганди в текстах, робота та метрики ефективності якої були перевірені на наборах даних англійською й українською мовами та прототип системи виявлення пропаганди.

Сформовано новий набір даних зі зразками пропаганди РФ за 2022 рік та в подальшому перекладено даний набір даних українською мовою програмним способом. Проведено дослідження ефективності різних методів попередньої обробки природної мови та їх комбінацій. Обґрунтовано використання найефективнішого з розглянутих методів попередньої обробки даних у контексті вирішення поставленої задачі, а саме методу, що являє собою комбінацію операцій токенізації, нормалізації та лематизації тексту без видалення стоп-слів.

На сформованому наборі даних з Twitter англійською мовою отримано наступні показники ефективності моделі класифікації: Precision – 0,79, Recall – 0,7 та Average Precision – 0,91. На наборі даних, перекладеному програмним способом українською мовою отримано показники: Precision – 0,85, Recall – 0,72 та Average Precision – 0,88.

Завдяки застосуванню в запропонованому архітектурному рішенні системи мікросервісного підходу та стратегії масштабування Sharding у СКБД MongoDB, система має змогу гнучко масштабуватись. Як наслідок, зі зростанням кількості клієнтів системи, вона потребуватиме більшою мірою горизонтального масштабування, аніж вертикального, що є більш бажаною стратегією масштабування систем з інтенсивним використанням даних.

Перспективним напрямком подальшого розвитку є навчання моделі класифікації на наборі даних, що включає інші зразки пропаганди та навчання на змішаних наборах даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Giles K. Russia’s Hybrid Warfare: A Success in Propaganda [Електронний ресурс] / Keir Giles. – 2015. – Режим доступу до ресурсу: https://www.researchgate.net/publication/280922184.

2) Osmani S. War in Ukraine: Propaganda and disinformation [Електронний ресурс] / S. Osmani, A. Zeneli. – 2022. – Режим доступу до ресурсу: https://www.researchgate.net/publication/359894583.

3) Cole R. The Encyclopedia of propaganda / Robert Cole., 1998. – 504 с.

4) Koppang H. Social Influence by Manipulation: A Definition and Case of Propaganda : дис. докт. іст. наук / Koppang Haavard, 2009.

5) Disinformation Resilience in Central and Eastern Europe [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: http://prismua.org/wp-content/uploads/2018/06/DRI_CEE_2018.pdf.

6) Лассуел Г. Техніка пропаганди у світовій війні / Гарольд Лассуел., 1938.

7) Fine-Grained Analysis of Propaganda in News Articles : дис. канд. фіз.-мат. наук / Da San Martino Giovanni, 2019. – 12 с.

8) Jowett G. Propaganda & Persuasion 7th Edition / G. Jowett, V. O'Donnell., 2019. – 400 с.

9) Іваницька Б. ОСНОВНІ МЕТОДИ ПРОПАГАНДИ В РОСІЙСЬКОМУ ІНТЕРНЕТ-ЗМІ PRAVDA.RU: дис. канд. наук з соц. комун. / Іваницька Божена – Львів, 2018.

10) Cross-Platform State Propaganda: Russian Trolls on Twitter and YouTube during the 2016 U.S. Presidential Election : дис. докт. політ. наук / Golovchenko Yevgeniy, 2020. – 33 с.

11) Yu S. Interpretable Propaganda Detection in News Articles / Yu Seunghak, 2021.

12) Yoosuf S. Fine-Grained Propaganda Detection with Fine-Tuned BERT [Електронний ресурс] / S. Yoosuf, Y. Yang. – 2019. – Режим доступу до ресурсу: https://www.researchgate.net/publication/336999573.

13) Da San Martino G. Detection of Propaganda Techniques in News Articles / Da San Martino Giovanni – SemEval-2020, 2020.

14) Vorakitphan V. PROTECT: A Pipeline for Propaganda Detection and Classification / Vorakitphan Vorakit, 2022. – 8 с.

15) Liu B. Sentiment Analysis Mining Opinions, Sentiments, and Emotion / Bing Liu., 2015. – 451 с.

16) Menshawy A. Deep Learning By Example: A Hands-on Guide to Implementing Advanced Machine Learning Algorithms and Neural Networks / Ahmed Menshawy., 2018. – 452 с.

17) Потятиник Б. RUSSIAN PROPAGANDA MACHINE: NEW DIMENSION [Електронний ресурс] / Борис Потятиник. – 2021. – Режим доступу до ресурсу: https://www.researchgate.net/publication/353466643.

18) Carter E. Questioning More: RT, Outward-Facing Propaganda, and the Post-West World Order [Електронний ресурс] / E. Carter, B. Carter // University of Southern California. – 2021. – Режим доступу до ресурсу: https://www.researchgate.net/publication/349601626.

19) Use Cases, Tutorials, & Documentation | Twitter Developer Platform [Електронний ресурс] – Режим доступу до ресурсу: https://developer.twitter.com.

20) Tweepy: Twitter for Python [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/tweepy/tweepy

21) A script to download all of a user's tweets into a csv [Електронний ресурс] – Режим доступу до ресурсу: https://gist.github.com/yanofsky/5436496

22) Zheng A. Evaluating Machine Learning Models / Alice Zheng., 2015.

23) Translation AI. Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://cloud.google.com/translate#section-5.

24) Googletrans – Free Google Translate API for Python [Електронний ресурс] – Режим доступу до ресурсу: https://pypi.org/project/googletrans/.

25) Erjavec T. MULTEXT-East: Morphosyntactic resources for Central and Eastern European languages [Електронний ресурс] / Tomaž Erjavec. – 2012. – Режим доступу до ресурсу: https://www.researchgate.net/publication/257546903.

26) spaCy — Models & Languages [Електронний ресурс] – Режим доступу до ресурсу: https://spacy.io/usage/models#languages.

27) Marinescu D. Cloud Computing: Theory and Practice / Dan C. Marinescu., 2013. – 415 с.

28) Kleppmann M. Designing Data-Intensive Applications / Martin Kleppmann., 2017. – 613 с.

29) Pattern: API Gateway / Backends for Frontends [Електронний ресурс] – Режим доступу до ресурсу: https://microservices.io/patterns/apigateway.html.

30) Guide to the “CPU-bound” and “I/O bound” Terms [Електронний ресурс] – Режим доступу до ресурсу: https://www.baeldung.com/cs/cpu-io-bound.

31) Numpy – the fundamental package for scientific computing with Python. [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/numpy/numpy.

32) scikit-learn: machine learning in Python [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/scikit-learn/scikit-learn.

ДОДАТОК А СКРЕЙПЕР

```
import csv
import json
import logging
from typing import Generator, Iterable

import requests
import bs4

log = logging.getLogger(__name__)

SITEMAP_FNAME = 'sitemap_2022.xml'
REQUEST_HEADERS_FNAME = 'request_headers.json'
OUTPUT_FNAME = 'rt_articles.csv'



def load_request_headers() -> dict:
    with open(REQUEST_HEADERS_FNAME, 'r') as fp:
        return json.load(fp)



REQUEST_HEADERS = load_request_headers()



def get_sitemap_urls() -> list[str]:
    # response = requests.get('https://www.rt.com/sitemap_2022.xml')
    # content = response.text
    with open(SITEMAP_FNAME, 'rt') as f:
        content = f.read()

    soup = bs4.BeautifulSoup(content, 'lxml')

    urls = []

    for loc in soup.select('loc'):
        url = loc.text
        urls.append(url)

    return urls



def get_article_content(body: str) -> dict:
    html_parser = bs4.BeautifulSoup(body, 'html.parser')

    header_node = html_parser.select_one('.article__heading')
    summary_node = html_parser.select_one('.article__summary')
    body_node = html_parser.select_one('.article__text')

    title_inner_text = (
        header_node.get_text(strip=True)
        if header_node
        else None
    )
    summary_inner_text = (
        summary_node.get_text(strip=True)
        if summary_node
        else None
    )
    body_inner_text = (
        body_node.get_text(strip=True)
        if body_node
        else None
```

```
    )

    return {
        'title': title_inner_text,
        'summary': summary_inner_text,
        'body': body_inner_text,
    }


def get_articles(articles_urls: Iterable[str]) -> Generator[dict, None, None]:
    for i, _url in enumerate(articles_urls):

        response = requests.get(_url, headers=REQUEST_HEADERS)
        log.info(response)
        if not response.ok:
            log.error(
                f'got response with error: {response.status_code=}. '
                'trying to skip...'
            )
            continue

        log.info(f'got {i} article out of {len(articles_urls)}')
        content = get_article_content(response.text)

        yield content


def write_articles_to_file(articles: Iterable[dict]) -> None:
    log.info('start writing tweets...')

    with open(OUTPUT_FNAME, 'w') as fp:
        writer = csv.writer(fp)

        header = ['title', 'summary', 'body']
        writer.writerow(header)

        for article in articles:
            writer.writerow(
                [article['title'], article['summary'], article['body']]
            )

    log.info('finished writing tweets')


def main() -> None:
    urls = get_sitemap_urls()

    log.info('got urls, processing articles...')

    articles_stream = get_articles(urls)
    write_articles_to_file(articles_stream)

    log.info('job done!')


if __name__ == '__main__':
    main()
```

ДОДАТОК Б СЦЕНАРІЙ СТВОРЕННЯ НАБОРУ ДАНИХ

```
import pandas as pd
from sklearn.utils import shuffle

pd.set_option('display.max_colwidth', 100)


def load_twitter_messages() -> pd.DataFrame:
    ok_messages = pd.concat([
        pd.read_csv('BBCWorld_tweets.csv', encoding='utf-8'),
        pd.read_csv('bpolitics_tweets.csv', encoding='utf-8'),
        pd.read_csv('KyivIndependent_tweets.csv', encoding='utf-8'),
    ])
    ok_messages['is_propaganda'] = False

    rus_media_messages = pd.concat([
        pd.read_csv('rt_com_tweets.csv', encoding='utf-8'),
        pd.read_csv('mfa_russia_tweets.csv', encoding='utf-8'),
    ])
    rus_media_messages['is_propaganda'] = True

    ok_messages_fraction = min(1.0, len(rus_media_messages) / len(ok_messages))

    messages = pd.concat([
        ok_messages.sample(frac=ok_messages_fraction),
        rus_media_messages,
    ])

    messages = shuffle(messages)

    return messages


def main():
    messages = load_twitter_messages()
    messages.to_csv('twitter_dataset.csv')


if __name__ == '__main__':
    main()
```

ДОДАТОК В МОДЕЛЬ КЛАСИФІКАЦІЇ

```
import re

import gensim
import numpy
import pandas
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import (
    precision_score,
    recall_score,
)
from sklearn.utils import shuffle
import spacy
import matplotlib.pyplot as plt
from sklearn.metrics import PrecisionRecallDisplay

pandas.set_option('display.max_colwidth', 60)



def load_twitter_messages() -> pandas.DataFrame:
    return shuffle(pandas.read_csv('twitter_dataset.csv'))



def preprocess_message(x: str) -> list[str]:
    if not isinstance(x, str):
        return []

    x = re.sub("[^A-Za-z']+", ' ', x).lower()
    return gensim.utils.simple_preprocess(x)



nlp = spacy.load('en_core_web_sm', disable=['ner', 'parser'])



def preprocess_message_advanced(x: str) -> list[str]:
    if not x or not isinstance(x, str):
        return []

    document = nlp(x)

    clean_tokens = [
        token._lemma.lower()
        for token in document
        if not token.is_stop and not token.is_punct
    ]

    return clean_tokens



def make_agg_vectors_from_text(
    word2vector_model: gensim.models.Word2Vec,
    text_data: pandas.Series,
) -> numpy.ndarray:
    existing_words_in_model = set(word2vector_model.wv.index_to_key)

    return numpy.array([
        numpy.array([
            word2vector_model.wv[word]
            for word in words
            if word in existing_words_in_model
        ])
```

```
        for words in text_data
    ])


def make_mean_vector(v: numpy.ndarray) -> numpy.ndarray:
    if v.size:
        return v.mean(axis=0)
    else:
        return numpy.zeros(100, dtype=float)


def make_mean_vectors(vectors: numpy.ndarray) -> list[numpy.ndarray]:
    return list(map(make_mean_vector, vectors))


messages = load_twitter_messages()

messages['text_processed'] = messages['text'].apply(preprocess_message_advanced)
messages['text_processed'].dropna()


X_train, X_test, y_train, y_test = train_test_split(
    messages['text_processed'],
    messages['is_propaganda'],
    test_size=0.2,
)


word2vector_model = gensim.models.Word2Vec(
    messages['text_processed'],
    vector_size=100,
    window=5,
    min_count=2
)

word2vector_model.save('w2v_model_propaganda_twitter')


X_train_vectors = make_mean_vectors(make_agg_vectors_from_text(word2vector_model, X_train))
X_test_vectors = make_mean_vectors(make_agg_vectors_from_text(word2vector_model, X_test))


random_forest = RandomForestClassifier()
random_forest.fit(X_train_vectors, y_train.values.ravel())


y_pred = random_forest.predict(X_test_vectors)


def accuracy_score(y_true: numpy.ndarray, y_pred: numpy.ndarray) -> float:
    return (y_pred == y_true).sum() / len(y_pred)


precision = round(precision_score(y_test, y_pred), 3)
recall = round(recall_score(y_test, y_pred), 3)
accuracy = round(accuracy_score(y_test, y_pred), 3)

print(
    f'Precision: {precision}\n'
    f'Recall: {recall}\n '
    f'Accuracy: {accuracy}'
)
```

```
rfc_display = PrecisionRecallDisplay.from_estimator(
    random_forest,
    X_test_vectors,
    y_test.values.ravel(),
    name="Класифікатор RandomForest"
)
_ = rfc_display.ax_.set_title("Крива метрик Precision-Recall")
handles, labels = rfc_display.ax_.get_legend_handles_labels()

plt.show()
```

ДОДАТОК Г СЦЕНАРІЙ ПЕРЕКЛАДУ НАБОРУ ДАНИХ УКРАЇНСЬКОЮ

```
import logging
import csv
import googletrans


log = logging.getLogger(__name__)


def main():
    translator = googletrans.Translator(raise_exception=True)

    with open('twitter_dataset.csv', 'r') as in_file, \
            open('twitter_dataset_translated.csv', 'w') as out_file:

        reader = csv.DictReader(
            in_file,
            fieldnames=[None, 'id', 'created_at', 'text', 'is_propaganda'],
        )

        writer = csv.DictWriter(
            out_file,
            fieldnames=['id', 'created_at', 'text', 'is_propaganda']
        )

        for row in reader:
            try:
                translated = translator.translate(row['text'], 'uk')
            except Exception as e:
                log.exception('Something went wrong')
                continue

            writer.writerow({
                'id': row['id'],
                'created_at': row['created_at'],
                'is_propaganda': row['is_propaganda'],
                'text': translated.text,
            })

            log.info('processing...')


if __name__ == '__main__':
    main()
```

ДОДАТОК Д ФРАГМЕНТ НАБОРУ ДАНИХ ЗІБРАНОГО З TWITTER

id,created_at,text,is_propaganda

880,1521836191829667841,2022-05-04 12:56:37+00:00,EU plans Russian oil ban and war crimes sanctions https://t.co/Tivzk8RJGO,False
57,1527282678285729793,2022-05-19 13:39:00+00:00,"RT @RusEmbSriLanka: Here is one of many accounts of the atrocities & crimes against civilians committed by the neo-Nazi Azov Battalion, w…",True
1034,1506503438816980996,2022-03-23 05:29:43+00:00,RT @RusEmbUSA: Anatoly #Antonov: Pumping the #Kiev regime with weapons and sending foreign mercenaries to the territory is irresponsib…,True
600,1514564990635757570,2022-04-14 11:23:27+00:00,"This is Snezhana. She was born on 14.04.2014 - the very day that the Kiev regime started its ""anti-terrorist operat… https://t.co/jhTvb2ZP1K",True
1290,1518602144949186562,2022-04-25 14:45:40+00:00,Marine Le Pen concedes French election but still counts a win https://t.co/DwY0Txw3zl,False
3080,1515595339864150018,2022-04-17 07:37:42+00:00,"Israeli police have entered the Al-Aqsa Mosque compound, a sensitive Jerusalem holy site, two days after clashes wi… https://t.co/9ZL8e6QToV",False

1023,1506864464372633600,2022-03-24 05:24:19+00:00,"RT @RussiaUN: #Nebenzia: To ensure that #Ukraine no longer poses a threat to #Russia, we need to denazify and demilitarize , and those ar…",True
2354,1511219519078940674,2022-04-05 05:49:45+00:00,They were found in places where Russian forces had been stationed. Russians withdrew from Sumy region on April 3.,False

1695,1517926276438274049,2022-04-23 18:00:00+00:00,US doesn’t want peace in Ukraine – Russia https://t.co/wwKHkSg8g7,True
87,1527595516212793351,2022-05-20 10:22:07+00:00,Argentina found guilty of massacre of Qom and Moqoit people https://t.co/9uQO8zhaYD,False
878,1522609929613455361,2022-05-06 16:11:10+00:00,"⚡ Russian top official says Russia is in Kherson 'forever.'

“There will be no return to the past,” Andrey Turcha… https://t.co/U2IVIYMVYd",False
1384,1517555595326468096,2022-04-22 17:27:03+00:00,Inside Kyiv's trench defences https://t.co/vdydxjmXU5,False
2010,1520040456272363521,2022-04-29 14:01:00+00:00, TUNE IN: Arguments were made for and against a controversial Trump-era immigration policy. What do the courts hav… https://t.co/18KH513rJS,False
2720,1507089386755792899,2022-03-24 20:18:04+00:00,Spotify paid 130 artists more than $5m last year https://t.co/mNIpieKPHn,False
1486,1497370314828914695,2022-02-26 00:37:57+00:00,"RT @RussiaUN: #Nebenzia at UNSC meeting on #Syria: Starting from December 2018, 364 Russian children have been repatriated from conflict…",True
727,1523905678980616194,2022-05-10 06:00:01+00:00,Eternal Flame and giant Saint George Ribbon in Mariupol https://t.co/G0746Fc0pr,True
1511273520944328708,2022-04-05 09:24:20+00:00,Italy expels 30 Russian diplomats citing security concerns – Reuters https://t.co/it0kbEFQRY,False
1511267404382363654,2022-04-05 09:00:01+00:00,"The third stage of a Chinese rocket has fallen back to Earth, blazing a path through the night sky over Maharashtra… https://t.co/RE4mJ76ORV",False
1511261620265271296,2022-04-05 08:37:02+00:00,Denmark expels 15 Russian diplomats – FM Jeppe Kofod https://t.co/DaKGaWOH2u,False
1462,1519308273865928706,2022-04-27 13:31:34+00:00,Russia expels 8 Japanese diplomats https://t.co/gnPa81IgoY,True
1983,1516413323641765902,2022-04-19 13:48:04+00:00,Russia expels four Austrian embassy staffers https://t.co/mkLwo4oRix,True
1527892599352213504,2022-05-21 06:02:37+00:00,Authorities in Beijing have sent all residents of a large housing complex into quarantine after a total of 26 cases… https://t.co/hGlhtvAYgy
1527874959133052928,2022-05-21 04:52:31+00:00,Boris Johnson is bracing for a final day of scrutiny over the illegal parties in Downing Street during the pandemic https://t.co/6OF8WqzBhH
218,1523697135476035584,2022-05-09 16:11:20+00:00,"On May , 1945, the Medal “For the Victory Over Germany in the Great Patriotic War 1941–1945” was established.

… https://t.co/2U2cCw5N0V",True

ДОДАТОК Ж ФРАГМЕНТ НАБОРУ ДАНИХ ЗІБРАНОГО З НОВИННИХ САЙТІВ

text,is_propaganda

"China had role in Yukos split-up

China lent Russia $6bn (Â£3.2bn) to help the Russian government renationalise the key Yuganskneftegas unit of oil group Yukos, it has been revealed.

The Kremlin said on Tuesday that the $6bn which Russian state bank VEB lent state-owned Rosneft to help buy Yugansk in turn came from Chinese banks. The revelation came as the Russian government said Rosneft had signed a long-term oil supply deal with China. The deal sees Rosneft receive $6bn in credits from China's CNPC.

According to Russian newspaper Vedomosti, these credits would be used to pay off the loans Rosneft received to finance the purchase of Yugansk. Reports said CNPC had been offered 20% of Yugansk in return for providing finance but the company opted for a long-term oil supply deal instead. Analysts said one factor that might have influenced the Chinese decision was the possibility of litigation from Yukos, Yugansk's former owner, if CNPC had become a shareholder. Rosneft and VEB declined to comment. ""The two companies [Rosneft and CNPC] have agreed on the pre-payment for long-term deliveries,"" said Russian oil official Sergei Oganesyan. ""There is nothing unusual that the pre-payment is for five to six years.""

The announcements help to explain how Rosneft, a medium-sized, indebted, and relatively unknown firm, was able to finance its surprise purchase of Yugansk. Yugansk was sold for $9.3bn in an auction last year to help Yukos pay off part of a $27bn bill in unpaid taxes and fines.

The embattled Russian oil giant had previously filed for bankruptcy protection in a US court in an attempt to prevent the forced sale of its main production arm. But Yugansk was sold to a little known shell company which in turn was bought by Rosneft. Yukos claims its downfall was punishment for the political ambitions of its founder Mikhail Khodorkovsky. Once the country's richest man, Mr Khodorkovsky is on trial for fraud and tax evasion.

The deal between Rosneft and CNPC is seen as part of China's desire to secure long-term oil supplies to feed its booming economy. China's thirst for products such as crude oil, copper and steel has helped pushed global commodity prices to record levels. ""Clearly the Chinese are trying to get some leverage [in Russia],"" said Dmitry Lukashov, an analyst at brokerage Aton. ""They understand property rights in Russia are not the most important rights, and they are more interested in guaranteeing supplies."" ""If the price of oil is fixed under the deal, which is unlikely, it could be very profitable for the Chinese,"" Mr Lukashov continued. ""And Rosneft is in desperate need of cash, so it's a good deal for them too.""
",0
"Orange colour clash set for court

A row over the colour orange could hit the courts after mobile phone giant Orange launched action against a new mobile venture from Easyjet's founder.

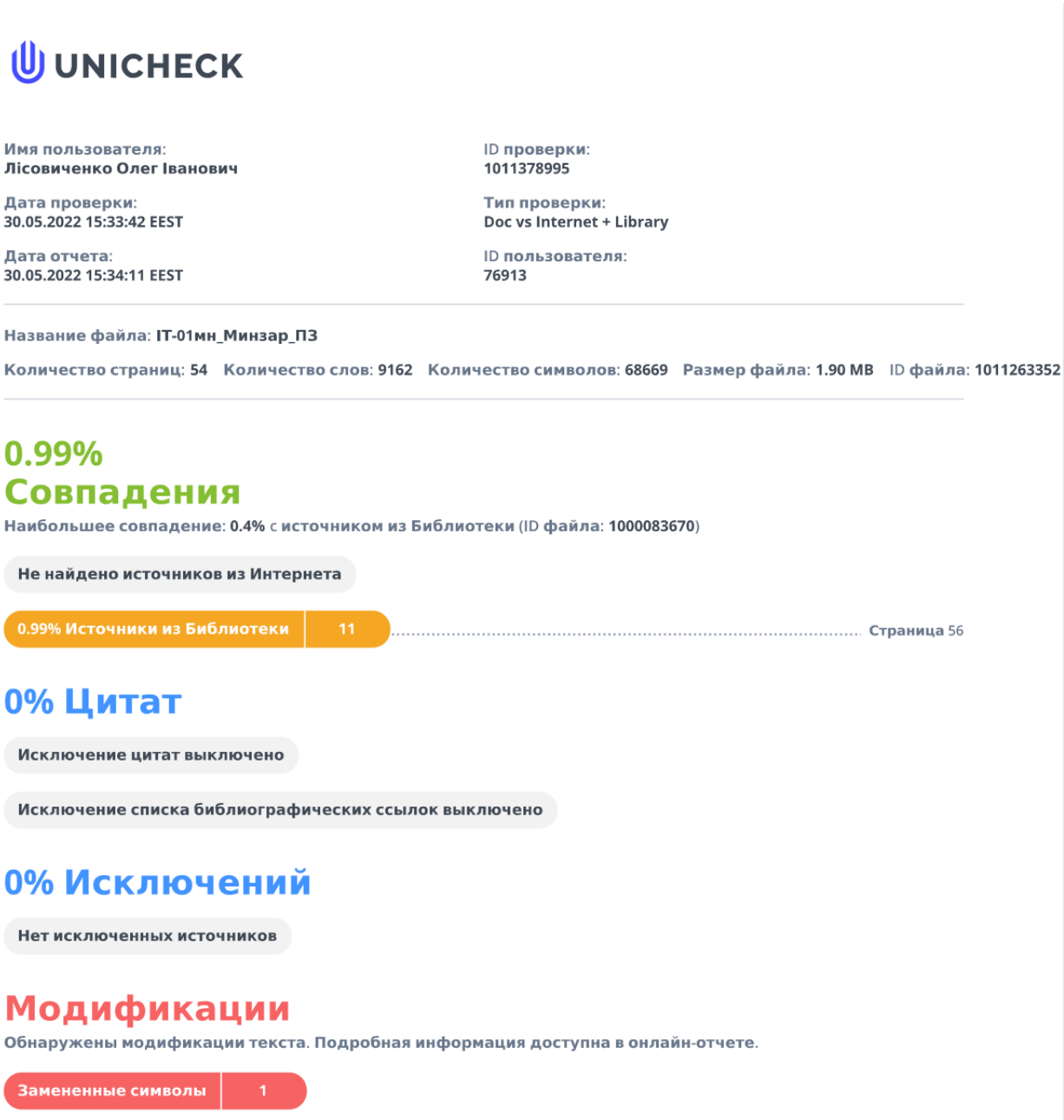
Orange said it was starting proceedings against the Easymobile service for trademark infringement. Easymobile uses Easygroup's orange branding. Founder Stelios Haji-Ioannou has pledged to contest the action. The move comes after the two sides failed to come to an agreement after six months of talks. Orange claims the new low-cost mobile service has infringed its rights regarding the use of the colour orange and could confuse customers - known as ""passing off"".

""Our brand, and the rights associated with it are extremely important to us,"" Orange said in a statement. ""In the absence of any firm commitment from Easy, we have been left with no choice but to start an action for trademark infringement and passing off."" However, Mr Haji-Ioannou, who plans to launch Easymobile next month, vowed to fight back, saying: ""We have nothing to be afraid of in this court case. ""It is our right to use our own corporate colour for which we have become famous during the last 10 years."" The Easyjet founder also said he planned to add a disclaimer to the Easygroup website to ensure customers are aware the Easymobile brand has no connection to Orange. The new service is the latest venture from Easygroup, which includes a chain of internet cafes, budget car rentals and an intercity bus service. Easymobile will allow customers to go online to order SIM cards and airtime - which will be rented from T-Mobile - for their existing handsets.

",0
"The newly recognised Donbass republics in Lugansk and Donetsk have formally asked Russia for military assistance in letters published on Wednesday.In them, their leaders claim that Ukrainian“aggression”has only increased since Moscow recognized the regions as independent states, earlier this week.The heads of Donetsk People's Republic (DPR) and the Lugansk People's Republic (LPR) wrote to Russian President Vladimir Putin separately, but both letters were dated Tuesday, February 22.The DPR’s Denis Pushilin and his LPR counterpart Leonid Pasechnik invoked articles three and four of their newly ratified treaties on cooperation and mutual aid with Russia, asking Moscow to“render aid in repelling the military aggression of the Ukrainian regime,”which they claim is waging war against them.READ MORE:Moscow warns Kiev against ‘adventures’ in Donbass“Ukrainian aggression is increasing,”Pushilin wrote citing the alleged increase in artillery bombardment targeting critical civilian infrastructure and reportedly leaving 300,000 people without water after the republic's main waterworks were hit. The DNR leader claimed Ukraine is continuing what he called a“genocide”of the civilians, which apparently has forced the evacuation of over 40,000 people so far.“Actions of the Kiev regime testify that they have no desire to carry out the Minsk agreements and stop the war in Donbass,”wrote Pasechnik, adding that Ukraine is receiving military aid from the US and other Western countries and is“oriented towards ending the conflict with the LPR by force.”Pasechnik also noted that over 51,000 people have been evacuated from Lugansk so far, more than half of them children.Ukrainian President Volodymyr Zelensky has insisted there was no military offensive aimed at the two regions, which Kiev considers“temporarily occupied”renegade territories. Ukraine has also accused the DPR and LPR of staging“false flag”incidents against their own civilians to justify a“Russian invasion.”READ MORE:‘Without Russia, we won't stop a full-on Ukrainian offensive’: What people in Donbass say about the ongoing military crisisDonetsk and Lugansk declared their independence from Ukraine in 2014, after the West-backed protests ended with a coup ousting the democratically elected government in Kiev. The Ukrainian military’s attempts to subjugate the area by force failed, resulting in the uneasy ceasefires signed in Minsk, Belarus – first in September 2014, then in February 2015.Moscow had long refused to recognize the two republics, pointing to Minsk and calling the conflict an internal Ukrainian matter. On Monday, however, Putin said that Kiev had openly refused to comply with the Minsk agreements and signed a decree on the“long overdue”recognition of Donetsk and Lugansk.",1

"Russia finds it hard to believe that NATO is a purely defensive bloc, given its bombing campaign in Yugoslavia in 1999, Russian Foreign Minister Sergey Lavrov told RT on Friday.The minister made his comments while speaking about the talks between German Chancellor Olaf Scholz and Russian President Vladimir Putin in Moscow on Tuesday.“Scholz and others NATO officials say that NATO is a defensive alliance. Putin reminded Scholz at a joint press conference about the bombing of Yugoslavia in 1999,”Lavrov remarked.“[Scholz] said that NATO had intervened in order to prevent the genocide of Kosovar Albanians. That it was a success, and now the region is prospering. It is far from prospering.”“Kosovo and some other parts of the Western Balkans are becoming a hotbed of crime. There are terrorists, drug dealers. Mercenaries are recruited there for military conflicts ignited by the US, among others,”the minister said.“There is information that militants from Kosovo, Albania, and Bosnia and Herzegovina are being recruited to knock Russia off balance, which includes sending them to Donbass [in eastern Ukraine]. We are working to verify it right now.”Read moreLavrov labels Western ‘Russia invasion’ claims ‘propaganda, fakes and fiction’“To say that NATO invaded Yugoslavia with noble goals is incorrect and unethical, to say the least,”Lavrov said.In 1999, NATO launched a 78-day bombing campaign, claiming that it was protecting civilians against atrocities committed by Serbian troops and police during an insurgency of ethnic Albanians in Kosovo. UN peacekeepers were deployed in the region after Serbian forces left.Kosovo’s independence, unilaterally declared in 2008, has been recognized by nearly 100 countries, including the US. Russia still considers Kosovo a part of Serbia.“The situation described by Scholz – when Yugoslavia was bombed – has nothing in common with genocide. International courts have not made rulings on that matter,”Lavrov argued.The minister pointed to the high-profile incident in the Kosovar village of Racak, where dozens of bodies were found in 1999. Western countries claimed that they were civilians massacred by the Serbs and used it as a pretext for military action. Their account was disputed by Belgrade, which said that the bodies belonged to Albanian militants.In 2006, the UN-backed International Criminal Tribunal for the former Yugoslavia decided not to use the evidence related to the Racak incident in order to“improve the expeditiousness of the proceedings while ensuring that they remain fair.”The court ultimately convicted several Serbian officials of war crimes and crimes against humanity committed in Kosovo, including murders and deportations. Several Albanians were convicted of war crimes against the Serbs as well.",1
"If you listen to Western leaders or the corporate media, you’d believe that Russia is by far the most evil and regressive country on the planet and therefore needs to be sanctioned back to the Stone Age as punishment. But how much of this rhetoric is based on fact rather than the fear of the Russian bear coming out of hibernation into our interdependent multipolar world?Ross Ashcroft is joined by professor of Slavic studies Vladimir Golstein and filmmaker Andrei Nekrasov to discuss Russia’s past, present, and future.YOUTUBEChannelRenegade Inc.LIKERenegade Inc. onFacebook hereFOLLOWRenegade Inc. at@Renegade_IncPODCASTRenegade",1

ДОДАТОК З РЕЗУЛЬТАТИ ПЕРЕВІРКИ НА СПІВПАДІННЯ



UNICHECK

Имя пользователя:
Лісовиченко Олег Іванович

ID проверки:
1011378995

Дата проверки:
30.05.2022 15:33:42 EEST

Тип проверки:
Doc vs Internet + Library

Дата отчета:
30.05.2022 15:34:11 EEST

ID пользователя:
76913

Название файла: IT-01мн_Минзар_ПЗ

Количество страниц: 54 **Количество слов:** 9162 **Количество символов:** 68669 **Размер файла:** 1.90 MB **ID файла:** 1011263352

0.99%
Совпадения

Наибольшее совпадение: 0.4% с источником из Библиотеки (ID файла: 1000083670)

Не найдено источников из Интернета

0.99% Источники из Библиотеки 11 ..... Страница 56

0% Цитат

Исключение цитат выключено

Исключение списка библиографических ссылок выключено

0% Исключений

Нет исключенных источников

Модификации

Обнаружены модификации текста. Подробная информация доступна в онлайн-отчете.

Замененные символы 1