НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ

кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

"На правах рукопису"
УДК 004.912

«До захисту допущено»
Завідувач кафедри
_____ Наталія АУШЕВА
"___" _____2026 р.

Магістерська дисертація

на здобуття ступеня другого (магістерського) рівня вищої освіти
за освітньою програмою "Комп'ютерні науки"
зі спеціальності 122 "Комп'ютерні науки"

на тему __Методи резюмування документів на основі моделей-трансформерів

Виконав: студент 2 курсу, групи ТР-41 мн
Новицький Костянтин Віталійович_____ _____
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник: професор, д.т.н., професор Шушура О.М. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Рецензент: професор, д.т.н., професор Гаврилко Є.В. _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Н.контроль: провідний інженер Людмила КУЗЬМІНА _____
(підпис)

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

Студент _____
(підпис)

Київ - 2026

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ
кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

Рівень вищої освіти – другий (магістерський)

спеціальність 122 “Комп’ютерні науки”

Освітньо-наукова програма “Комп’ютерні науки”

ЗАТВЕРДЖУЮ
Завідувач кафедри ЦТЕ
_____ Наталія АУШЕВА
«___»_____2026 р.

ЗАВДАННЯ

на магістерську дисертацію студенту

Новицькому Костянтину Віталійовичу

(прізвище, ім’я, по батькові)

1. Тема роботи Методи резюмування документів на основі моделей-трансформерів

керівник роботи *Шушура Олексій Миколайович, д.т.н., професор*
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «30» березня 2026 р. № 1326-с

2. Термін подання студентом дисертації 05.05.2026

3. Об’єкт дослідження: процес автоматичного резюмування текстових документів

4. Вихідні дані: мова програмування Python, мова програмування .NET, бібліотеки для роботи з моделями-трансформерами, перевірочний набір даних

5. Перелік завдань: проаналізувати існуючі підходи до автоматичного резюмування, приділивши увагу методам обробки довгих текстів, розробити власний метод гібридного резюмування документів великого обсягу, обґрунтувати

вибір інструментів та створити програмне забезпечення для практичної реалізації методу, провести експериментальне дослідження ефективності розробленого методу та порівняти його з наявними підходами.

6. Орієнтований перелік ілюстративного матеріалу: архітектура та узагальнений алгоритм системи

7. Орієнтований перелік публікацій: одні тези-доповіді на конференції та одна стаття у фаховому технічному журналі

6. Дата видачі завдання «15» вересня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ № з/п	Назва етапів підготовки магістерської дисертації	Термін виконання	Примітка
1	Аналіз методів дослідження за літературними джерелами	06.02.2026	виконано
2	Вибір засобів реалізації методів/моделей	13.02.2026	виконано
3	Програмна реалізація	28.02.2026	виконано
4	Проведення тестування та валідації системи	06.03.2026	виконано
5	Аналіз результатів експериментів	13.03.2026	виконано
6	Захист створенного програмного забезпечення	23.03.2026	виконано
7	Оформлення магістерської дисертації	16.04.2026	виконано

Студент _____ **Костянтин НОВИЦЬКИЙ**
(підпис) (ім’я, ПРІЗВИЩЕ)

Керівник роботи _____ **Олексій ШУШУРА**
(підпис) (ім’я, ПРІЗВИЩЕ)

АНОТАЦІЯ

Дипломна робота виконана на 109 сторінках, містить 19 рисунків, 5 таблиць, 2 додатки, 43 джерела в переліку посилань.

У сучасних умовах стрімкого зростання обсягів текстової інформації дедалі більшого значення набувають методи автоматичного узагальнення змісту документів. Це стосується наукових статей, урядових матеріалів та інших довгих текстів, довжина яких перевищує розмір контекстного вікна багатьох моделей-трансформерів. За таких умов пряме резюмування часто призводить до втрати важливої інформації, зниження зв'язності та погіршення якості підсумкового тексту. Тому розробка методів, орієнтованих саме на обробку документів великого обсягу, є актуальною задачею сучасної обробки природної мови.

Метою дослідження є підвищення якості резюмування довгих документів шляхом розробки гібрибного методу на основі моделей-транформерів з використанням структурно-семантичної сегментації документа, кластеризації текстових блоків, виділення ключових тверджень та локального абстрактивного і подальшого глобального резюмування.

Завдання дослідження:

– проаналізувати існуючі підходи резюмування документів;

– розробити метод гібридного резюмування;

– обрати засоби та розробити програмне забезпечення для реалізації методу;

– провести порівняльне дослідження застосування розробленого методу з існуючими підходами до резюмування документів.

Об'єктом дослідження є процес автоматичного резюмування текстових документів.

Предметом дослідження є методи та інформаційні технології автоматичного резюмування документів на основі моделей-трансформерів.

Методи та засоби: у роботі використано методи обробки природної мови, семантичного векторного подання тексту, кластеризації, генеративного

резюмування на основі моделей-трансформерів, а також експериментальне порівняння за метриками ROUGE, BERTScore та часом виконання. Практична реалізація виконана у вигляді програмної системи з модулем резюмування на Python, серверною частиною на ASP.NET Core та клієнтською частиною на Angular.

Наукова новизна полягає у створенні багатоступеневого методу автоматичного резюмування документів великого обсягу, який поєднує структурно-семантичну сегментацію документа, відбір інформативних текстових блоків, кластеризацію змістово близьких фрагментів, виділення ключових тверджень, локальне резюмування та подальше глобальне узагальнення. На відміну від однорівневих підходів, такий метод дозволяє краще зберігати зміст документа при роботі з довгими текстами та зменшує навантаження на фінальну генеративну модель.

Практичне значення роботи: розроблена програмна система резюмування документів, яка забезпечує повний цикл взаємодії користувача із системою: аутентифікацію, завантаження документа, запуск побудови резюме та перегляд отриманого результату через веб-інтерфейс. Практична цінність системи підтверджується експериментально: її застосування забезпечує скорочення часу побудови резюме та підвищення якості результату. На вибірці BookSum запропонований підхід показав приріст BERTScore приблизно на 13% та зменшення часу обробки приблизно на 48% порівняно з використанням моделі.

Апробація результатів дослідження. Основні результати роботи висвітлено у статті, опублікованій у першому випуску за 2026 рік фахового журналу «Вісник Херсонського національного технічного університету», а також представлено на XXIII Міжнародній науково-практичній конференції молодих вчених та студентів «Сучасні проблеми наукового забезпечення енергетики: безпека, стійкість, ІТ та екологічний моніторинг (до 40-річчя Чорнобильської катастрофи)».

Ключові слова: резюмування тексту, обробка природної мови, моделі-трансформери, Sentence-BERT, Qwen, автоматичне резюмування, програмна система.

ABSTRACT

Thesis consists of 109 pages and includes 19 figures, 5 tables, 2 appendices, and 43 references.

Under the current conditions of the rapid growth of textual information, methods for automatic summarization of document content are becoming increasingly important. This applies to scientific articles, government materials, and other long texts whose length exceeds the context window size of many transformer models. Under such conditions, direct summarization often leads to the loss of important information, reduced coherence, and lower quality of the final summary. Therefore, the development of methods specifically oriented toward the processing of long documents is an actual task in modern natural language processing.

The purpose of the research is to improve the quality of summarization of long documents by developing a hybrid method based on transformer models using structural-semantic document segmentation, clustering of text blocks, extraction of key statements, and local abstractive and subsequent global summarization.

Objectives of the work are:

– to analyze existing approaches to document summarization;

– to develop a hybrid summarization method;

– to select tools and develop software for implementing the method;

– to conduct a comparative study of the application of the developed method in comparison with existing approaches to document summarization.

The object of the research is the process of automatic summarization of text documents.

The subject of the research is methods and information technologies for automatic document summarization based on transformer models.

Methods and tools: the work uses methods of natural language processing, semantic vector representation of text, clustering, generative summarization based on transformer models, as well as experimental comparison using ROUGE, BERTScore, and

execution time. The practical implementation was carried out in the form of a software system with a summarization module in Python, a server-side component in ASP.NET Core, and a client-side component in Angular.

The scientific novelty lies in the creation of a multi-stage method for automatic summarization of large-volume documents, which combines structural-semantic document segmentation, selection of informative text blocks, clustering of semantically related fragments, extraction of key statements, local summarization, and subsequent global generalization. Unlike single-level approaches, such a method makes it possible to better preserve the content of a document when working with long texts and reduces the load on the final generative model.

The practical significance of the work lies in the developed document summarization software system, which provides a complete cycle of user interaction with the system: authentication, document upload, summary generation, and viewing of the obtained result through a web interface. The practical value of the system is confirmed experimentally: its application ensures a reduction in the time required to generate a summary and an improvement in the quality of the result. On the BookSum dataset, the proposed approach showed an approximately 13% increase in BERTScore and an approximately 48% reduction in processing time compared with using the model.

Approbation of the research results. The main results of the work are presented in an article published in the first issue of 2026 of the professional journal “Visnyk of Kherson National Technical University”, and were also presented at the XXIII International Scientific and Practical Conference of Young Scientists and Students “Modern Problems of Scientific Support of Energy: Safety, Sustainability, IT and Environmental Monitoring (Dedicated to the 40th Anniversary of the Chornobyl Disaster)”.

Keywords: text summarization, natural language processing, transformer models, Sentence-BERT, Qwen, automatic summarization, software system.

ЗМІСТ

ВСТУП

У сучасних інформаційних системах обсяг текстових даних безперервно зростає. Наукові публікації, технічні звіти, патентна документація, нормативні акти, аналітичні матеріали та корпоративні документи формують великі масиви тексту, повне опрацювання яких потребує значних витрат часу. У таких умовах автоматичне резюмування стає важливим інструментом, що дозволяє швидко отримати стислий виклад основного змісту документа без необхідності детального прочитання всього тексту.

Попри значний розвиток моделей-трансформерів, автоматичне резюмування довгих документів залишається складною задачею. Більшість сучасних генеративних моделей демонструє високі результати на коротких і середніх текстах, однак при роботі з великими документами виникають обмеження, пов’язані з довжиною контекстного вікна, обчислювальною складністю та зниженням якості підсумкового тексту.

У таких випадках пряме подання повного документа до моделі часто супроводжується втратою частини важливої інформації, погіршенням зв’язності резюме або надмірним часом обробки. Це зумовлює необхідність побудови спеціалізованих багатоступеневих підходів, у яких великі документи попередньо структуруються, скорочуються та подаються до генеративної моделі у більш керованому вигляді.

Одним із перспективних напрямів розв’язання цієї проблеми є використання гібридних підходів до резюмування, у межах яких поєднуються екстрактивні та абстрактивні методи. Це дозволяє підвищити керованість процесу побудови резюме та краще зберігати зміст документа під час роботи з довгими текстами.

Метою дослідження є підвищення якості резюмування довгих документів шляхом розробки гібридного методу на основі моделей-транформерів з використанням структурно-семантичної сегментації документа, кластеризації

текстових блоків, виділення ключових тверджень та локального абстрактивного і подальшого глобального резюмування.

Для досягнення поставленої мети в роботі вирішуються задачі аналізу сучасних підходів до автоматичного резюмування, розроблення власного багатоступеневого методу, вибору засобів реалізації, створення програмної системи та проведення експериментального порівняння запропонованого підходу з базовими варіантами резюмування.

Досягнення цієї мети здійснювалося через побудову багатоступеневої схеми обробки документа, яка включає структурно-семантичний поділ тексту, кластеризації текстових блоків, виділення ключових тверджень, локальне резюмування та подальше фінальне узагальнення. Отримані результати показують, що подібні системи можуть бути ефективними для автоматичного резюмування довгих документів в умовах обмеженого контекстного вікна генеративних моделей.

1 ОГЛЯД МЕТОДІВ АВТОМАТИЧНОГО РЕЗЮМУВАННЯ ТЕКСТУ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

На сьогоднішній день резюмування тексту є однією з актуальних задач обробки природної мови, оскільки сучасні інформаційні системи працюють із великими обсягами текстових даних, які потребують швидкого та змістовного узагальнення.

Перед розробкою власного методу та програмної системи необхідно проаналізувати існуючі підходи до автоматичного резюмування, архітектури моделей, способи обробки довгих документів і підходи до оцінювання якості отриманих резюме.

У цьому розділі розглядаються основні підходи до автоматичного резюмування тексту, архітектура моделі-трансформера, методи обробки документів, а також метрики оцінювання якості резюме. На основі проведеного аналізу формулюється постановка задачі дослідження.

1.1 Автоматичне резюмування тексту: визначення, класифікація та основні підходи

Автоматичне резюмування тексту – це напрям обробки природної мови, що вивчає методи автоматичного скорочення тексту для отримання змістовного та зв’язного резюме, яке зберігає ключові ідеї вихідного документа.

Актуальність цієї задачі зумовлена стрімким зростанням обсягів текстової інформації в науковій, медичній, юридичній, корпоративній та медійній сферах, де автоматичне резюмування дає змогу зменшити витрати часу на опрацювання великих масивів даних [1].

У сучасній літературі автоматичне резюмування зазвичай класифікують за способом формування підсумкового тексту на:

- екстрактивне;
- генеративне (абстрактивне);
- за кількістю джерел на однодокументне та багатодокументне.

Така класифікація є базовою для аналізу сучасних систем резюмування та використовується в більшості оглядових праць останніх років [2].

Сенс екстрактивного резюмування полягає в відборі найбільш інформативних речень або абзаців з вихідного документа та їх об’єднанні у фінальний резюме-текст без переформулювання.

У сучасних оглядах до класичних екстрактивних методів відносять підходи на основі графів, TF-IDF, ранжування речень і центральності, а типовими представниками такого напряму вважаються алгоритми на кшталт TextRank. Вони мають наступні переваги та недоліки:

Переваги:

- відносно висока фактична надійність, оскільки резюме формується безпосередньо з фрагментів вихідного джерела;
- простота реалізації та інтерпретації результатів;
- менший ризик спотворення змісту порівняно з абстрактивними методами.

Недоліки:

- уривчастість і слабка зв’язність фінального тексту;
- відсутність переформулювання, через що резюме може виглядати менш природним;
- у результаті часто отримуються незмінені речення, взяті з різних частин тексту.

Генеративне (абстрактивне) резюмування відрізняється тим, що модель формує новий текст, який може містити переформульовані думки, узагальнення змісту оригінального документу, не тільки копії фрагментів тексту. Такий підхід

ближчий до людського способу написання резюме й потенційно забезпечує більш природний і читабельний результат.

Проте генеративні методи є складнішими з точки зору моделювання та більш чутливими до проблем фактичної неузгодженості й галюцинацій, тобто появи тверджень, не підкріплених джерелом [3].

Переваги:

- дає змогу формувати новий, цілісний і більш природний текст;
- забезпечує вищу читабельність резюме;
- дозволяє переформульовувати думки та узагальнювати зміст документа;
- є ближчим до людського способу написання резюме.

Недоліки:

- є складнішим з точки зору моделювання;
- більш чутливе до фактичної неузгодженості;
- може породжувати галюцинації, тобто твердження, не підкріплені джерелом.

З точки зору масштабу вхідних даних розрізняють однодокументне та багатодокументне резюмування. В однодокументному резюмуванні система працює з одним документом і формує його стислий виклад.

У багатодокументному резюмуванні на вхід до моделі подається кілька тематично пов’язаних джерел, а система має сформувати узагальнений резюме-текст, що охоплює ключові положення з усіх документів. Цей сценарій є складнішим, оскільки вимагає узгодження потенційно суперечливої інформації та визначення відносної важливості окремих джерел.

Варто відмітити таку категорію задач як резюмування текстів великого обсягу. До таких документів належать наукові статті, юридичні договори, технічні звіти, корпоративна документація та інші тексти, довжина яких значно перевищує стандартні вхідні межі багатьох трансформерних моделей.

Сучасні великі мовні моделі мають великий контекст, який би теоретично мав би вміщувати весь великий документ, але в такому разі як правило модель може

губити контекст [4]. Також, довгі документи, як правило, мають внутрішню ієрархічну структуру, розподіл змісту між секціями та наявність важливих залежностей між віддаленими частинами тексту, що ускладнює їхнє резюмування.

У сучасних дослідженнях підкреслюється, що стандартні архітектури на основі моделей-трансформерів гірше працюють з довгими документами через кілька причин: обмеження довжини вхідної послідовності, значну обчислювальну вартість механізму уваги, нерівномірний розподіл важливої інформації по документу та схильність до втрати суттєвих деталей при сильному резюмуванні тексту.

Саме ці фактори стимулювали появу моделей з великим контекстом, ієрархічних схем і багатоступеневих підходів для резюмування довгих документів.

1.2 Трансформерні архітектури у задачах резюмування

Моделі-трансформери стала фундаментом сучасних систем резюмування, оскільки механізм самоуваги дозволяє моделі враховувати контекстні зв’язки між елементами послідовності значно ефективніше, ніж класичні рекурентні архітектури [5]. У сучасних оглядових роботах трансформери розглядаються як базова архітектурна основа для більшості стандартних підходів до автоматичного резюмування.

У типовому варіанті трансформер для задачі резюмування реалізується як архітектура «енкодер-декодер». Енкодер формує контекстні представлення вхідного тексту, а декодер покроково генерує резюме, спираючись як на попередньо згенерований контекст, так і на приховані стани енкодера. Саме така конфігурація є природною для задачі перетворення повного документа у стислий підсумковий текст.

Ключовим механізмом цієї архітектури є механізм масштабованої точкової уваги. Для трьох матриць – запитів Q, ключів K та значень V вихід уваги обчислюється за формулою:

$$Attention\ (Q,\ K,\ V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) \cdot V,$$

де d_k – розмірність вектора ключа [6].

Ділення на $\sqrt{d_k}$ запобігає проблемі насичення функції *softmax* при великих значеннях скалярного добутку, що виникає при великих розмірностях.

Повноцінна реалізація уваги в трансформерах використовує концепцію багатоголовної уваги, яка дозволяє моделі паралельно звертати увагу на різні підпростори представлень. Обчислення багатоголовної уваги та визначення окремої головки проводиться за наступними формулами:

$$MultiHead\ (Q, K, V) = Concat\ (head_1, \ldots, head_h)\ W^O,$$

$$head_i = Attention\ (QW_i^Q, KW_i^K, VW_i^V),$$

де h – кількість головок,

$W^O \in R^{(h \cdot d_v \times d_{model})}$ – матриця вихідної проєкції,

$W_i^Q \in R^{(d_{model} \times d_k)}$, $W_i^K \in R^{(d_{model} \times d_k)}$, $W_i^V \in R^{(d_{model} \times d_v)}$ – матриці проєкцій для i-тої головки, $i = 1 \ldots h$,

d_{model} – загальна розмірність моделі,

d_k – розмірність ключів і запитів,

d_v – розмірність значень.

Завдяки паралельній обробці h модель здатна фіксувати різноманітні аспекти семантичної залежності між елементами послідовності одночасно. На рисунку 1.1 зображена схема роботи моделі-трансформера.

Як вже було сказано, архітектура трансформера складається з декодера та енкодера, кожен з яких має шість однакових шарів.

Вхідний текст спочатку токенізується, перетворюється на векторні подання та доповнюється позиційним кодуванням, та перетворюється на контекстуалізовані вектори на виході енкодера. Декодер, використовуючи механізм багатоголової уваги та враховуючи інформацію з виходів енкодера, генерує фінальний текст резюме.

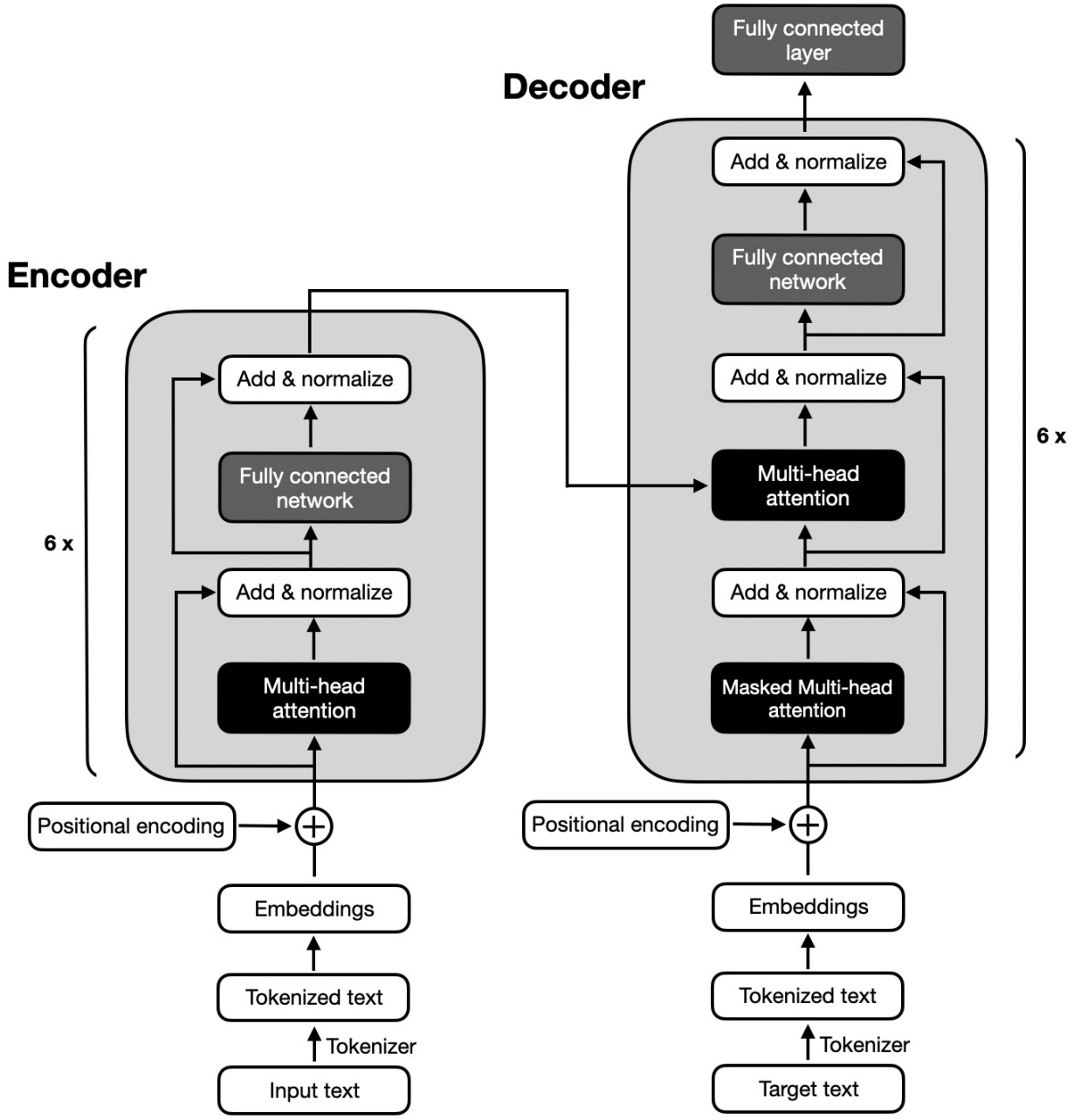

Рисунок 1.1 – Схема роботи моделі-трансформера [7]

Переваги:

- здатність враховувати контекст усього вхідного тексту;
- ефективне моделювання залежностей між далекими словами в послідовності;
- використання механізму багатоголової уваги, що дає змогу одночасно аналізувати різні аспекти тексту;

– можливість отримання контекстуалізованих векторних подань що пришвидшує роботу.

Недоліки:

– висока обчислювальна складність;
– значні вимоги до обсягу пам’яті та ресурсів;
– потреба у великій кількості даних для ефективного навчання;
– складність архітектури порівняно з простішими моделями;
– можливе уповільнення роботи на довгих послідовностях.

Серед найвідоміших трансформерних моделей для резюмування важливе місце посідає PEGASUS, запропонована у 2020 році [8]. Відмінною особливістю цієї моделі є спеціальна стратегія попереднього навчання, за якої модель навчається відновлювати вилучені з документа речення. Завдяки цьому PEGASUS стала однією з базових моделей для генеративного резюмування, особливо після передтренуванні на цільових наборах даних.

Моделі GPT є за своїм змістом декодер-архітектурами, попередньо навчені шляхом авторегресійного прогнозування наступного токена. У сучасних версіях, зокрема GPT-5, такі моделі демонструють високу якість текстової генерації, здатність дотримуватися інструкцій, що робить їх придатними і для задач резюмування, особливо в інструктивному або режимі без попередніх прикладів [9]. Логотип GPT-5 наведено на рисунку 1.2.

Рисунок 1.2 – GPT-5 лого [10]

Qwen це сімейство великих мовних моделей, розроблених компанією Alibaba Cloud. Згідно з технічним звітом Qwen2.5, моделі цього сімейства демонструють сильні результати в задачах інструктивної генерації, довгого генерування тексту та дотримання структурних вимог до вихідного тексту [11]. Завдяки цьому моделі Qwen є перспективними як компоненти фінальної генерації резюме у багатоступеневих системах. Логотип Qwen наведено на рисунку 1.3.

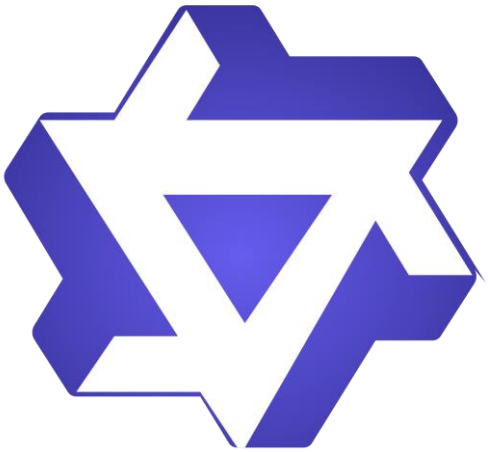

Рисунок 1.3 – Qwen лого [12]

Gemma - це сімейство великих мовних моделей, розроблених компанією Google. Згідно з технічними описами, моделі цього сімейства демонструють високі результати в задачах генерації тексту, дотримання інструкцій і роботи з мовними шаблонами. Вони здатні формувати зв’язний і читабельний текст, що робить їх перспективними для використання в задачах узагальнення та резюмування, зокрема як компоненти фінальної генерації резюме у багатоступеневих системах.

Водночас моделі Gemma, як і інші великі мовні моделі, мають певні обмеження: вони можуть допускати фактичні неточності або галюцинації, їхня ефективність значною мірою залежить від якості сформульованого запиту, а використання більших версій потребує суттєвих обчислювальних ресурсів. Крім того, під час роботи зі спеціалізованими або вузькопрофільними текстами може

виникати потреба в додатковому контролі якості результатів. Логотип Gemma наведено на рисунку 1.4.

Рисунок 1.4 – Gemma лого [13]

LongT5 є спеціалізованою модифікацією сімейства моделі-трансформера T5, адаптованою до роботи з довгими послідовностями [14]. Ця модель орієнтована на обробку значно довших вхідних текстів, ніж стандартні енкодер-декодер моделі. У контексті довгих документів LongT5 розглядається як одна з базових моделей для резюмування довгих документів.

Одним з фундаментальних обмеженням трансформерних архітектур є контекстне вікно, тобто максимальна довжина вхідної послідовності до моделі. Навіть моделі з розширеним контекстом не завжди здатні обробити весь зміст великих документів.

Для резюмування довгих документів ця проблема безпосередньо впливає на якість покриття змісту документа та стимулює використання різних підходів, такі як ієрархічний та поділ на структури [15]. Таким чином, проблема обмеження контекстного вікна безпосередньо мотивує до розробки альтернативних підходів до резюмування довгих документів, зокрема методів розбиття документа на фрагменти, ієрархічних схем обробки та багатоступеневих стратегій відбору і генерації тексту.

Окремою практичною проблемою є час виконання моделей, особливо великих мовних моделей, що містять кілька мільярдів параметрів або більше. Зі збільшенням розміру моделі зростають обчислювальні витрати як на етапі обробки вхідного тексту, так і під час генерації підсумкового резюме. Це призводить до збільшення затримки отримання результату, підвищення вимог до апаратних ресурсів і ускладнює використання таких моделей у системах, де важливо забезпечити швидку відповідь.

Крім того, тривалий час виконання стає особливо відчутним у задачах резюмування довгих документів, оскільки в таких випадках модель повинна обробляти великі обсяги тексту або декілька фрагментів послідовно. У результаті проблема часу виконання є не лише технічним, а й практичним обмеженням, яке впливає на масштабованість, вартість використання моделі та доцільність її застосування в тому числі в дослідницьких умовах.

Локально запускати подібні моделі може бути проблематично через технічні обмеження – недостатня кількість VRAM, низька потужність процесора, високий час очікування, тощо. На даний момент для використання доступні хмарні сервіси, де можна розгорнути модель або скористатися публічним API однієї з компаній які надають доступ до моделей(такі як GPT, Claude, Gemini) але кількість таких моделей обмежено і вимагає більших фінансових затрат аніж локальне розгортання.

1.3 Методи обробки документів великого обсягу

Для подолання проблеми обмеженого контекстного вікна у задачах резюмування довгих документів у сучасній літературі пропонується кілька базових стратегій. До них належать урізання тексту, ковзне вікно, ієрархічні моделі та багатоступеневі системи. Кожна з цих стратегій має власні переваги й обмеження, що визначають доцільність її застосування в конкретному сценарії.

Найпростішим підходом є урізання тексту, при якому документ обрізається до максимально допустимої довжини моделі. Такий підхід може працювати але він часто призводить до втрати ключових положень, розташованих у середині або наприкінці тексту. Тому урізання вважається лише базовим, але не достатнім підходом до резюмування довгих документів.

Метод ковзного вікна є більш просунутою альтернативою усіченню. У цьому випадку документ ділиться на перекривні фрагменти фіксованої довжини, які обробляються незалежно, а отримані часткові результати далі агрегуються. Такий підхід дозволяє охопити більшу частину документа, але сам по собі не гарантує глобальної зв’язності фінального резюме та може втрачати залежності між секціями. Він був описаний в статі разом з моделлю BART-Large-CNN і показав покращення, порівняно з базовими моделями та іншими гібридними методами [16]. Проте, він недостатньо гарно працює з більш потужними моделями через втрату зв’язності.

Більш теоретично обґрунтованим напрямом є ієрархічні підходи. Їхня ключова ідея полягає у розділенні процесу обробки довгого документа на кілька рівнів: спочатку текст ділиться на речення, абзаци або секції, а потім формується глобальний текст, який використовується для генерації підсумкового тексту. Ієрархічні схеми дозволяють краще поєднувати локальну та глобальну інформацію, але є складнішими в реалізації та налаштуванні [15].

Також важливим підходом є багатоступеневі схеми обробки. На відміну від єдиної архітектури, такі системи розбивають задачу на послідовність етапів: попередній відбір релевантних частин документа, локальне резюмування змісту, побудову проміжного представлення та фінальну генерацію резюме.

Разом із тим, стандартні варіанти мають характерні недоліки. Якщо на етапі відбору система працює лише з ізольованими реченнями, а не зі зв’язними, фінальна генеративна модель отримує фрагментований вхід. У результаті резюме може виглядати не як цілісний текст. Сучасні дослідження показують, що перехід до зв’язних блоків і проміжного планування покращує читабельність підсумкового тексту [17].

1.4 Структурно-орієнтовані та блочні методи резюмування

Одним із ключових спостережень сучасної літератури з резюмування довгих документів є те, що текст не є просто лінійною послідовністю речень, він має складну ієрархічну організацію [17]. У наукових статтях, технічних звітах, юридичних актах та інших великих документах суттєву роль відіграють розділи, підрозділи, заголовки, перехідні абзаци та функціональні секції. Кожна з них виконує власну унікальну змістову роль. Якщо модель не враховує межі секцій і локальні тематичні переходи, вона може переоцінити вступні або методичні фрагменти, при цьому недооцінивши результати чи висновки. Саме це у сучасних дослідженнях розглядається як головна причина слабкої якості резюмування в простіших підходах.

У відповідь на ці обмеження були запропоновані структурно-орієнтовані методи резюмування. Їхня ключова ідея полягає в тому, що процес сегментації, відбору змісту та подальшої генерації має спиратися на структурні межі документа. У нових роботах [15] показано, що така організація дає наступні переваги:

- полегшує збереження загальної змістової повноти;
- значно покращує узгодженість підсумкового тексту.

Близьким до структурно-орієнтованих підходів є блочне резюмування. У ньому базовою одиницею обробки виступає не окреме речення, а контекстуально зв’язний текстовий блок. Це дозволяє краще зберігати локальний контекст і формувати більш надійне проміжне представлення для моделі для подальшої генерації. Таким чином забезпечується тематичне різноманіття обраних блоків, а не лише представлення центральних фрагментів тексту.

Наступним важливим компонентом сучасних генеративних систем є планування змісту. Такий план виконує роль каркасу для майбутнього резюме: він визначає, які теми мають бути висвітлені, у якому порядку та з яким ступенем деталізації. Завдяки наявності явного плану вдається розділити дві задачі відбір

змісту та формування зв’язного тексту, що, згідно з сучасними дослідженнями, підвищує контрольованість процесу генерації [18].

Спільним для всіх зазначених підходів є використання того факту, що для якісного резюмування довгих документів недостатньо лише збільшити розмір моделі. Необхідно або розширювати контекстне вікно, або вводити ієрархічні механізми обробки, або ж поєднувати структурні й семантичні сигнали при відборі змісту.

1.5 Метрики оцінювання якості резюме

Оцінювання якості автоматично згенерованих резюме є складним завданням. Якісне резюме повинно одночасно задовольняти цілу низку критеріїв: бути інформативним, точним, стислим, зв’язним і читабельним. Жодна існуюча автоматична метрика не здатна охопити всі ці властивості відразу. Саме тому в сучасних дослідженнях [19] рекомендується використовувати набір метрик, які оцінюють різні аспекти якості тексту.

1.5.1 ROUGE як метрика збігу з еталонним текстом

ROUGE досі залишається найбільш вживаною метрикою для оцінювання резюме. У сучасних оглядах її розглядають як базову метрику, яка вимірює ступінь лексичного збігу між згенерованим та еталонним текстами.

Зазвичай така оцінка здійснюється через порівняння спільних слів, словосполучень або найдовших спільних підпослідовностей між двома текстами [19].

Найбільш поширеними є такі її варіанти:

- ROUGE-1 – відображає збіг за уніграмами, тобто окремими словами;
- ROUGE-2 – фіксує збіг за біграмами, тобто парами сусідніх слів;

– ROUGE-L – оцінює найдовшу спільну підпослідовність, тобто найдовшу послідовність слів що трапляється в обох текстах у тому самому порядку.

Таке поєднання дає змогу виміряти як локальний лексичний збіг, так і загальну подібність структури резюме до еталону. Зокрема, ROUGE-1 дає уявлення про збіг ключових слів, ROUGE-2 частково враховує зв'язки між словами та їх порядок, а ROUGE-L дозволяє оцінити, наскільки збережено загальну послідовність викладу. На практиці результати зазвичай подаються через показники точності, повноти та F1-міри.

Однак варто зазначити й недоліки: сучасні огляди підкреслюють, що ця метрика занадто чутлива до довжини тексту і майже не враховує семантичної еквівалентності перефразованих фрагментів. Іншими словами, два речення можуть бути близькими за змістом, але отримати низький показник ROUGE через різне формулювання. Тож вона є недостатньою метрикою при оцінці абстрактивного резюме.

1.5.2 BLEU метрика

BLEU є однією з найвідоміших метрик автоматичного оцінювання якості згенерованого тексту[20]. Спочатку її було запропоновано для задач машинного перекладу, однак згодом її почали використовувати і для оцінювання резюме.

У сучасних оглядах BLEU розглядають як метрику, що вимірює ступінь лексичного збігу між згенерованим та еталонним текстами на основі n-грам [21].

Найбільш поширеними є такі її варіанти:

– BLEU-1 – відображає збіг за уніграмами, тобто окремими словами;
– BLEU-2 – фіксує збіг за біграмами, тобто парами сусідніх слів;
– BLEU-3 – оцінює збіг за триграмами, тобто послідовностями з трьох слів;
– BLEU-4 – враховує збіг за чотириграмами, тобто послідовностями з чотирьох слів.

Таке поєднання дає змогу оцінити, наскільки згенерований текст збігається з еталонним не лише на рівні окремих слів, а й на рівні коротких словосполучень і локальних синтаксичних конструкцій. Зокрема, нижчі порядки n-грам, як-от уніграми та біграми, відображають загальний лексичний збіг, тоді як триграми й чотириграми краще враховують порядок слів і зв'язність фраз.

На практиці BLEU, зазвичай, подають як інтегральний показник, що обчислюється на основі модифікованої точності n-грам із додаванням штрафу за надто короткий текст.

Однак варто зазначити й недоліки: сучасні огляди підкреслюють, що BLEU, як і ROUGE, орієнтується насамперед на поверхневий лексичний збіг і недостатньо враховує семантичну подібність перефразованих фрагментів.

Крім того, ця метрика є більш придатною для задач, де важливе точне відтворення формулювання, ніж для абстрактивного резюмування, у якому можливі різні коректні способи передавання того самого змісту. Тож BLEU також не можна вважати достатньою метрикою для повноцінного оцінювання якості абстрактивного резюме [22].

1.5.3 BERTScore та семантичні метрики подібності

Сучаснішою альтернативою ROUGE виступає BERTScore. На відміну від простого лексичного порівняння, вона враховує саме семантичну подібність між текстами. Для цього використовуються контекстні векторні подання токенів від моделей сімейства BERT, що дозволяє краще розпізнавати синоніми або перефразовані вислови [23].

Принцип роботи BERTScore полягає в обчисленні косинусної близькості між контекстуальними векторними представленнями токенів згенерованого й еталонного текстів. Далі, як і в ROUGE, формуються оцінки точність, повнота і F1. Завдяки такому підходу BERTScore вважається придатнішою метрикою для оцінювання систем які генерують абстрактивні резюме, де поверхневий збіг слів вже не є ключовим критерієм.

1.5.4 Метрики зв’язності

Зв’язність у сучасних роботах розглядають як характеристику, що відображає логічну послідовність і семантичну пов’язаність речень у згенерованому тексті [24].

На практиці автоматичного аналізу тексту для її оцінювання часто використовують показники локальної зв’язності, зокрема подібність між сусідніми реченнями, а також між реченнями, розташованими на невеликій відстані одне від одного.

Такі показники можуть обчислюватися через косинусну подібність між векторними представленнями речень [25].

Найбільш поширеними є такі її варіанти:

– зв’язність першого порядку – оцінює локальну зв’язність тексту через подібність між сусідніми реченнями;
– зв’язність другого порядку – фіксує зв’язність між реченнями, що розташовані через одне, тобто дає змогу оцінити ширшу тематичну узгодженість тексту.

Таке поєднання дає змогу виміряти як локальну логічну зв’язність, так і загальну послідовність викладу в резюме. Зокрема, зв’язність першого порядку показує, наскільки плавно текст переходить від одного речення до наступного, тоді як зв’язність другого порядку допомагає оцінити, чи зберігається тематична єдність у ширшому контексті.

Ці показники обчислюються через косинусну подібність між векторними представленнями відповідних речень.

Однак варто зазначити й недоліки: ця метрика не оцінює, наскільки резюме є фактично правильним або повним щодо змісту оригінального документа. Високий рівень зв’язності може мати навіть текст, який є добре написаним, але пропускає важливу інформацію або містить змістові викривлення. Тому зв’язність доцільно використовувати разом з іншими метриками, що оцінюють змістову відповідність і семантичну точність резюме.

1.5.5 Точність відтворення фактів

Метрики фактичної достовірності або фактичної опори перевіряють, чи узгоджуються твердження у згенерованому резюме з вихідним документом. Для генеративних систем цей аспект є важливим, адже абстрактивне резюмування завжди несе ризик появи тверджень, які не підкріплені джерелом або взагалі йому суперечать.

Автоматично оцінити достовірність значно складніше, ніж порахувати ROUGE чи BERTScore. Проте він значно важчий в реалізації аніж попередньо наведені метрик. Для цього залучають моделі перевірки текстових імплікацій, залучення людей або спеціалізовані метрики.

1.6 Постановка задачі дослідження

На основі проведеного аналізу можна сформулювати задачу даного дослідження. Встановлено, що існуючі підходи до автоматично резюмування не завжди забезпечують належну якість при роботі з документами більшого обсягу, аніж контекстне вікно моделі. Таким чином, виникає необхідність розробити метод та програмну систему для автоматичного резюмування документів великого обсягу. Така система повинна вміти працювати з текстами, що перевищують стандартне контекстне вікно загальних трансформерних моделей, гарантуючи при цьому змістову повноту, зв’язність та прийнятний рівень фактичної достовірності.

Метою дослідження є підвищення якості резюмування довгих документів шляхом розробки гібрибного методу на основі моделей-транформерів з використанням структурно-семантичної сегментації документа, кластеризації текстових блоків, виділення ключових тверджень та локального абстрактивного і подальшого глобального резюмування.

Для досягнення цієї мети необхідно розв’язати такі завдання:

1) проаналізувати існуючі підходи до автоматичного резюмування, приділивши увагу методам обробки довгих текстів;

2) розробити власний метод гібридного резюмування документів великого обсягу;
3) обґрунтувати вибір інструментів та створити програмне забезпечення для практичної реалізації методу;
4) провести експериментальне дослідження ефективності розробленого методу та порівняти його з наявними підходами.

У межах дослідження передбачається, що ефективне резюмування документів досягається не вибором однієї моделі, а завдяки багатоступеневій схемі, яка в даному випадку поєднує:

- ієрархічне структурно-семантичне поділу тексту;
- побудову репрезентативних текстових блоків;
- кластеризацію
- виділення ключових тверджень;
- фінальну генерацію резюме на основі одночасного використання контексту й ключових фактів.

У межах постановки задачі також приймаються такі обмеження та припущення:

- у межах дослідження не розглядається задача розпізнавання тексту зі сканованих зображень;
- оцінювання ефективності методу виконується на фіксованому дослідному наборі документів.

Критеріями успішності запропонованого методу визначено:

- здатність системи стабільно обробляти документи в межах дослідного набору;
- отримання конкурентоспроможних результатів за метриками лексичного та семантичного збігу;
- отримання конкурентоспроможних результатів за метриками часу;
- підвищення показників абстрактивності та зв’язності порівняно з базовими підходами.

Висновки до розділу 1

У першому розділі виконано огляд теоретичних і прикладних засад автоматичного резюмування тексту. Розкрито зміст самої задачі резюмування, наведено її основні різновиди та охарактеризовано особливості роботи з документами різної структури і довжини. Увагу зосереджено на тому, що зі збільшенням обсягів текстових даних зростає потреба в інструментах, здатних швидко виділяти головний зміст документа без повного ручного опрацювання.

У межах розділу проаналізовано трансформерні архітектури, які сьогодні становлять основу більшості сучасних систем резюмування. Розглянуто механізм самоуваги, багатоголову увагу та їх значення для врахування контекстних зв’язків у тексті.

Окремо розглянуто підходи, які використовуються для обробки документів великого обсягу. До них віднесено урізання тексту, ковзне вікно, ієрархічні схеми, багатоступеневі методи, а також структурно-орієнтовані та блочні способи організації вхідного тексту.

Проведений аналіз показує, що при роботі з довгими документами недостатньо покладатися лише на безпосередню генерацію резюме з повного тексту. Більш обґрунтованими є підходи, у яких враховується внутрішня будова документа, тематичні межі між його частинами та послідовне скорочення змісту через проміжні подання.

Проведене опрацювання літератури дало підстави стверджувати, що наявні методи не завжди забезпечують достатню якість резюмування у випадку документів, довжина яких перевищує допустимі межі стандартних трансформерних моделей. Це зумовлює необхідність розробки власного методу, орієнтованого саме на великі документи.

2 РОЗРОБКА МЕТОДУ РЕЗЮМУВАННЯ ДОКУМЕНТІВ

Розроблення методу резюмування документів потребує переходу від однокрокового підходу до багатоступеневої схеми, у якій відбір змісту, формування проміжного представлення та побудова фінального резюме розглядаються як окремі, але узгоджені етапи [26].

Як вже зазначалось, запропонований у роботі метод орієнтований на ситуацію, коли повний документ або не вміщується у контекстне вікно генеративної моделі, або при прямій генерації створює нестійкий результат через нерівномірний розподіл важливої інформації по тексту.

Метод, розглянутий у магістерській роботі, є розвитком підходу, попередньо викладеного у статті про автоматичне резюмування документів [16]. У статті було сформульовано базову ідею багатоступеневого резюмування довгих текстів, а в дипломній роботі її використано як основу подальшого дослідження.

Цей підхід спирається на такі принципи: структурне подання довгого документа як системи пов’язаних блоків, кластеризація, відбір ключових фактів як змістових опор; подання фінальній моделі одночасно контексту й опорних тверджень, поетапне скорочення тексту без руйнування логіки документа [27].

У цьому розділі послідовно розглядаються структурно-семантична сегментація документа, формування текстових блоків, механізм пошуку ключових тверджень та кластеризація, принцип генерації та повний алгоритм роботи методу.

2.1 Структурно-семантична сегментація документа та побудова текстових блоків

Перший етап запропонованого методу безпосередньо пов’язаний з напрацюваннями, отриманими у статті [16]. У попередньому дослідженні для роботи з довгими науковими документами використовувалося розбиття тексту на

перекривні фрагменти, що дозволяло подавати великі тексти до моделі частинами. У цій роботі ця ідея розвивається – замість простого поділу за довжиною документ розглядається як система структурно й змістовно пов’язаних блоків, що дає змогу точніше зберігати логіку документа на етапі подальшого аналізу.

Під таким блоком розуміється мінімальна відносно завершена частина документа, яка зберігає локальну тему, не розриває межі між заголовком і підлеглим йому текстом та не змішує кілька різних змістових зон. Необхідність такого підходу пояснюється особливостями довгих документів.

У наукових статтях, технічних звітах, юридичних документах і урядових матеріалах зміст розподілений нерівномірно. Вступні секції задають контекст, окремі розділи описують метод або аргументацію, а результати та висновки можуть знаходитись далеко від початку документа і якщо ігнорувати цю будову і механічно різати текст по довжині, модель отримує фрагменти, які починаються або закінчуються всередині думки.

Це знижує якість подальшого змістового аналізу і погіршує фінальне резюме. На рисунку 2.1 зображений механізм структурно-семантичної сегментації та побудови текстових блоків.

Спочатку документ аналізується як структурований текст, виділяються заголовки, підзаголовки, абзаци, короткі списки, межі розділів. На цій основі формуються структурні блоки документа.

Якщо окремий блок виявляється занадто великим, він додатково ділиться вже не механічно, а за семантичними межами всередині блока. Для цього текст блока розбивається на речення, для речень обчислюються семантичні векторні подання, після чого аналізуються зміни змісту між сусідніми реченнями [28].

Поєднання структурного поділу та семантичного уточнення меж дає змогу сформувати фрагменти, які одночасно є структурно коректними, змістовно цілісними і придатними для подальшого оцінювання та відбору.

Кожен попередньо виділений фрагмент, а також окремі речення всередині нього, подаються у вигляді числових векторів, які відображають їхній загальний зміст. Далі порівнюється смислова близькість між сусідніми частинами тексту.

Рисунок 2.1 – Схема структурно-семантичної сегментації та побудови текстових блоків

Якщо два сусідні фрагменти мають близькі векторні подання, то вони вважаються такими, що продовжують одну і ту саму локальну тему, і можуть бути об’єднані в межах одного змістового блоку. Якщо ж між ними спостерігається помітне зниження смислової близькості, це розглядається як ознака тематичного переходу, і в такій точці доцільно провести межу між блоками.

Такий підхід особливо важливий для документів великого обсягу, оскільки в них структурні межі не завжди точно збігаються зі змістовими. Один розділ може

містити кілька підтем, а короткий абзац між двома великими частинами може, навпаки, логічно належати до попереднього фрагмента. Тому у запропонованому методі структурний аналіз задає початкову схему поділу, а смислове уточнення коригує її так, щоб фінальні блоки були придатними для подальшого відбору ключових тверджень і побудови підсумкового резюме.

Після побудови первинних структурних блоків система фільтрує їх для визначення найголовніших.

Критерії важливості текстового блоку:

- текстовий блок зберігає змістову цілісність;
- текстовий блок не повинен розривати заголовок і фрагмент тексту, на який цей заголовок поширюється;
- текстовий блок має бути достатньо коротким для обробки, але не настільки малим, щоб втратити контекст.

Окремим кроком є фільтрація допоміжних секцій. Реальні документи часто містять фрагменти, які формально є частиною тексту, але не повинні впливати на зміст підсумкового резюме. До них належать бібліографічні списки, подяки, додатки, технічні примітки, підписи до рисунків, довгі таблиці та інші службові частини. Якщо такі фрагменти не прибирати, модель може помилково завищити їхню вагу через велику кількість доменно специфічних слів.

Для контролю тематичної близькості між блоками використовується семантичне векторне подання. Кожен блок перетворюється на векторне подання, після чого для пари блоків обчислюється косинусна подібність.

Використання семантичного векторного подання та косинусної подібності в цьому підході також спирається на результати, отримані у статті [16]. Якщо в статті ця міра застосовувалася для усунення семантичних дублікатів між фрагментами тексту, то в цій роботі вона використовується також як інструмент контролю тематичної близькості між текстовими блоками та керування їх змістовим різноманіттям [29].

Якщо позначити вектори двох блоків як b_i та b_j, то міра їхньої близькості визначається за формулою:

$$sim(b_i, b_j) = (b_i \cdot b_j) / (||b_i|| \cdot ||b_j||)$$

Суть цієї формули полягає в тому, що вона оцінює не абсолютну довжину текстів, а подібність напрямів їхніх векторних подань. У запропонованому методі ця міра використовується не як єдиний критерій важливості, а як інструмент керування різноманіттям.

Висока подібність між двома вже відібраними блоками означає, що новий фрагмент, імовірно, дублює вже представлену тему. Відповідно, при формуванні фінального набору текстових блоків система підтримує баланс між інформативністю та новизною змісту.

2.2 Кластеризація та схема виділення ключових тверджень

Після формування текстових блоків система переходить до наступного етапу, на якому необхідно узгодити їх між собою за змістом і підготувати локальні змістові опори для подальшої генерації. На цьому етапі потрібно зберегти тематичне покриття документа, і визначити, які саме фрагменти та речення всередині них найкраще передають основний зміст відповідних частин документа.

Для цього відібрані фрагменти документа групуються за допомогою кластеризації [30], а в межах сформованих тематичних груп визначаються фрагменти та речення, що найкраще передають їхній основний зміст.

Схему кластеризації відібраних фрагментів та механізму виділення ключових тверджень зображено на рисунку 2.2.

Підхід до виділення ключових тверджень у даній роботі продовжує екстрактивну фазу, закладену в авторській статті [16].

У статті семантичні векторні представлення Sentence-BERT використовувалися для відбору найбільш інформативних речень із документа. У цій роботі цю ідею поглиблено — спочатку оцінюються самі текстові фрагменти, а далі всередині відібраних фрагментів формується набір змістових опор для подальшої генерації.

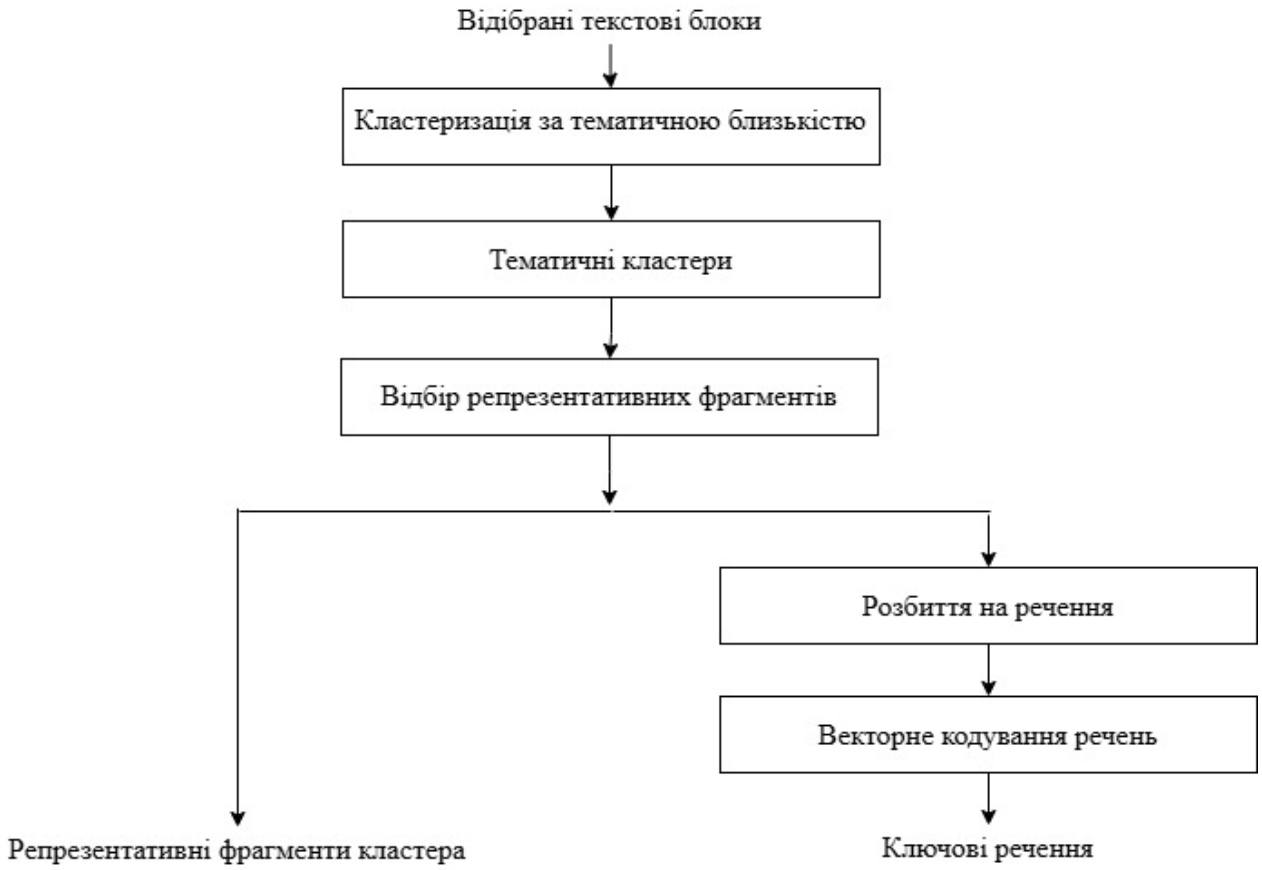

Рисунок 2.2 – Схема кластеризації відібраних фрагментів і виділення ключових тверджень

Близькі за змістом блоки групуються між собою за допомогою кластеризації. Для кожного текствого блоку береться його текст і перетворюється на векторні подання у спільному семантичному просторі. На їх основі виконується кластеризація. За допомогою мінімізації суми квадратів відстаней від кожної точки до центра її кластера, знаходяться фрагменти, які були б максимально схожі всередині а між групами максимально різними.

У результаті формуються тематичні кластери, кожен із яких об’єднує фрагменти, що стосуються однієї підтеми або близького змістового аспекту документа. Кількість кластерів є динамічною і залежить від довжини документа, якби вона була б фіксованою, то для коротших документів кластери могли б бути штучно роздроблені, а для довгі – занадто великими.

Для кожного фрагменту (кількість фрагментів n) є векторне подання x_i та задається число кластерів k. Далі ітеративно робляться дві дії – призначається кожен фрагмент до найближчого центру, який обчислюється за формулою:

$$c_i = \arg\min \left\| x_i - \mu_j \right\|^2, \text{ [30]}$$

де c_i – номер кластера, до якого належить фрагмент x_i, $i = 1 \ldots n$,

arg min – операція вибору такого кластера j, для якого відстань між фрагментом і центром є мінімальною,

x_i – векторне подання i-го фрагмента документа, $i = 1 \ldots n$,

μ_j – центр j-го кластера, $j = 1 \ldots k$,

$\left\| x_i - \mu_j \right\|^2$ – квадрат відстані між фрагментом x_i та центром кластера μ_j.

Кожен фрагмент документа відноситься до того кластера, центр якого є найближчим до його векторного подання. Це дозволяє згрупувати семантично близькі фрагменти в межах одного тематичного кластера.

Потім переобчислюється центр кожного кластера, як середнє всіх точок у ньому:

$$\mu_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i \text{ .[30]}$$

де μ_j – оновлений центр j-го кластера, $j = 1 \ldots k$,

C_j – множина фрагментів, які належать до j-го кластера, $j = 1 \ldots k$,

$|C_j|$ – кількість фрагментів у кластері C_j, $j = 1 \ldots k$,

x_i – векторне подання окремого фрагмента, що входить до кластера,

$\sum_{x_i \in C_j} x_i$ – сума всіх векторів фрагментів, які належать до кластера C_j.

Новий центр кластера визначається як середній вектор усіх фрагментів, що були віднесені до цього кластера. Завдяки цьому центр поступово зміщується до найбільш характерної області відповідної групи фрагментів.

Для кластера відбираються репрезентативні фрагменти. Вони використовуються як основа подальшого локального подання змісту кластера. Такий відбір дозволяє не передавати до моделі весь текст тематичної групи, але зберегти її основний зміст [25]. При цьому враховується змістова важливість

текстових блоків, яка була порахована в попередньому пункті 2.1, близькість до центру кластера [31]. Такий вибір було зроблене через те, що фрагмент, найближчий до центру кластера є зазвичай найкращим представником теми, але не завжди це відповідає дійсності, тому також враховується позиційна важливість, адже головні ідеї тексту можуть бути розділені в різних темах.

Додатково забезпечується позиційне покриття кластера. Спочатку виділяється підмножина фрагментів, чия змістова важливість є близькою до максимальної в межах кластера, після чого серед них визначаються перший і останній фрагменти за їхнім порядком у документі. Такий підхід дає змогу включити до репрезентативного набору також ті сильні фрагменти, що представляють різні позиційні частини кластера. Завдяки цьому зберігається і тематичне ядро кластера, і послідовність розвитку цієї теми, якщо важливі елементи теми розподілені в різних частинах документа.

Ключові речення виділяються не з усього документа, а з репрезентативних фрагментів кластера. Це дозволяє зосередити подальшу обробку на тих частинах тексту, які найкраще передають зміст відповідної тематичної групи [31]. У довгих документах такий підхід є важливим, оскільки глобальний відбір речень часто переоцінює загальні вступні формулювання і недооцінює локальні, але змістово важливі твердження.

Репрезентативний фрагмент розбивається на речення. Кожне речення далі подається до модуля векторного кодування, який створює для нього семантичне представлення за допомогою Sentence-BERT.

На основі цих представлень визначається, наскільки речення пов’язане з іншими змістовими реченнями того самого фрагмента – чим вища така семантична пов’язаність, тим важливішим вважається речення для передавання основної думки. Після цього отримана оцінка уточнюється з урахуванням трьох додаткових чинників: відповідності речення заголовку фрагмента, його фактологічної насиченості та його позиції у фрагменті [32]. Фактологічна насиченість речення оцінюється за сукупністю формальних ознак, що вказують на високу концентрацію предметної інформації. Для цього враховуються кількість чисел, абревіатур, слів з

великої літери та довгих термінологічних слів у реченні. Отримане значення нормується відносно довжини речення, завдяки чому вищу оцінку отримують речення, що містять більше фактів, назв, позначень і термінів у компактній формі. У результаті відбираються короткі речення, які найкраще відображають центральний зміст відповідного фрагмента та можуть бути використані як змістові опори на наступному етапі генерації.

У результаті цього етапу формуються два пов’язані компоненти. Перший це репрезентативні фрагменти кластера, які зберігають основний контекст відповідної тематичної групи. Другий компонент це короткі ключові речення, виділені з цих фрагментів, які уточнюють, конкретизують або підтримують їхній основний зміст [33]. Такий поділ потрібний, щоб на наступному етапі модель отримувала не лише основний текстовий контекст, а й короткі змістові опори, які допомагають точніше зберегти головні твердження.

2.3 Принцип генерації фінального резюме

Принцип фінальної генерації в цій роботі також пов’язаний із підходом, апробованим у статті [16]. У статті абстрактивний етап виконувався після попереднього екстрактивного скорочення документа. У цій роботі ця логіка зберігається, однак вона розширюється за рахунок кластеризації змістово близьких фрагментів, побудови локальних резюме окремих груп і подальшого формування фінального глобального резюме документа.

Структура набору, який іде на генерувальну модель:

- відібрані текстові блоки;
- ключові факти, визначені всередині цих блоків.

Важливою частиною є інструкції, які подаються на генерувальну модель, так як вона представлена архітектурою тільки декодер і може реагувати на відповідні інструкції в повідомленні. Інструкції дещо відрізняються в залежності від обраного

бажаного резюме(короткий, середній, довгий), містять в собі інформацію про бажаний формат резюме, про те, що текст має містити тези зі всіх локальних резюме.

Узагальнену схему генерації наведено на рисунку 2.3.

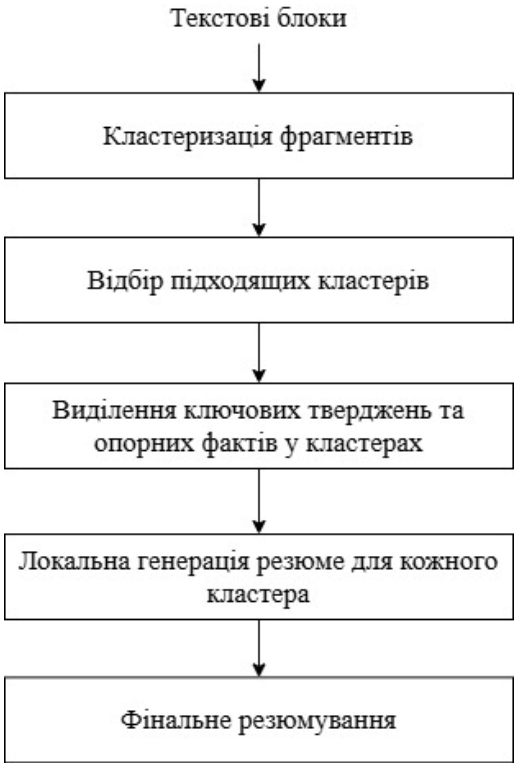

Рисунок 2.3 – Узагальнена схема генерації фінального резюме

На виході цього етапу система вже не працює з повним документом як з однорідним текстом. Документ представляється у вигляді відібраних і згрупованих фрагментів, далі у вигляді локальних резюме окремих тематичних груп які є основою для побудови фінального підсумкового тексту.

Необхідність такого підходу зумовлена пошуком компромісу між повнотою та керованістю. Якщо генерувальна модель бачить тільки текстові блоки, то в неї є локальний контекст, але немає явних головних змістових тем, тому вона може зміститися в бік другорядних деталей. Якщо ж модель отримує тільки стислий план або лише набір коротких тез, то втрачається природна зв’язність і зростає ризик занадто механічного, конспектного тексту.

Такий підхід також підвищує контрольованість системи. За потреби можна аналізувати, який саме текстовий блок став джерелом певного фрагмента резюме, які твердження були обрані як ядро змісту і чи не втрачено одну з важливих тем. Це робить метод придатним не лише для генерації, а й для подальшого дослідницького аналізу.

2.4 Алгоритм роботи методу та порівняння з однорівневими підходами

Враховуючи описані компонентів повний метод можна подати як послідовність взаємопов’язаних етапів. Кожен наступний етап працює з більш компактним, але більш керованим поданням документа. Основна логіка підходу подана на рисунку 2.4.

Послідовність взаємопов’язаних етапів:

1) документ завантажується, нормалізується та очищується від технічного шуму;
2) на основі заголовків, абзаців і списків формуються первинні структурні блоки;
3) для великих фрагментів виконується семантичне уточнення меж;
4) виконується оцінювання та відбір кандидатних фрагментів, при цьому виконується фільтрація;
5) кандидатні фрагменти групуються у кластери за змістовою близькістю;
6) усередині кожного кластера обираються репрезентативні фрагменти;
7) від репрезентативних фрагментів виділяються короткі ключові речення;
8) для кожного кластера формується локальний промпт, що містить репрезентативні фрагменти та ключові речення, після чого будується локальне резюме;

9) локальні резюме кластерів об’єднуються і подаються на фінальний етап генерації, де формується підсумкове резюме всього документа.

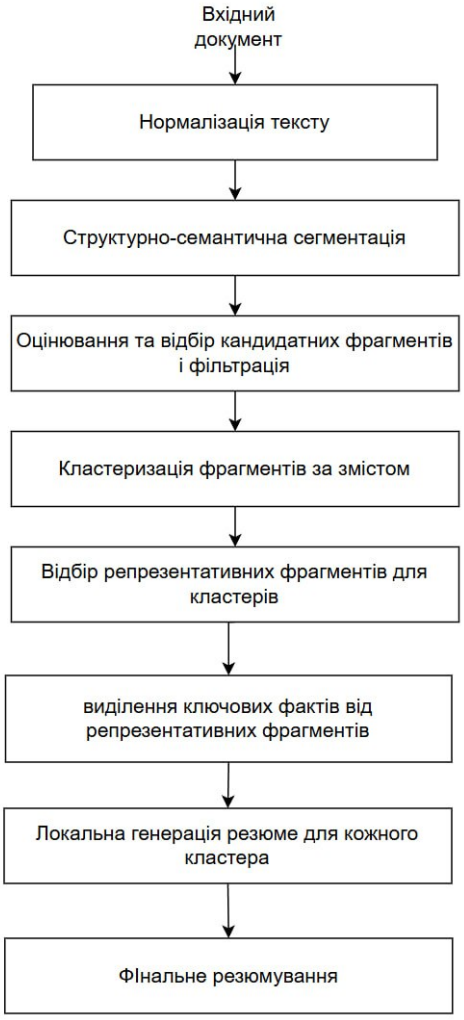

Рисунок 2.4 – Загальна логічна схема багатоступеневого методу резюмування

Така послідовність дозволяє не передавати великий документ у модель безпосередньо, але водночас не втрачати зв’язок між локальним контекстом і підсумковим текстом.

Порівняння з не-гібридними підходами:

– на відміну від простого скорочення тексту, метод не відкидає автоматично інформацію з середини або кінця документа;
– на відміну від прямої генерації по всьому документу, він зменшує навантаження на фінальну модель і робить процес більш керованим;

– на відміну від чисто екстрактивних систем, він не обмежується механічним набором речень, а дає можливість побудувати зв’язний фінальний текст;

Висновки до розділу 2

У другому розділі подано опис запропонованого методу резюмування документів великого обсягу, побудованого як послідовність кількох взаємопов’язаних етапів. Запропонований метод враховує обмеження генеративних моделей при роботі з довгими текстами і дозволяє краще контролювати, яка саме інформація використовується для формування підсумкового резюме.

У роботі запропоновано формувати текстові блоки з урахуванням структури документа та семантичних меж між його частинами. Далі виконується оцінювання та відбір найбільш змістовних фрагментів, які групуються за тематичною близькістю. Усередині сформованих кластерів обираються репрезентативні фрагменти, а вже в межах цих фрагментів виділяються короткі ключові речення, що використовуються як змістові опори для подальшої генерації. Такий підхід дозволяє передати моделі більш впорядковане проміжне подання змісту документа.

Після цього формуються локальні резюме окремих тематичних груп і фінальне глобальне резюме всього документа. У результаті запропонований метод поєднує структурно-семантичну сегментацію, відбір змістовно важливих фрагментів, кластеризацію змістово близьких частин документа, виділення ключових речень у репрезентативних фрагментах, локальне резюмування та фінальне узагальнення. Така організація методу становить основу його подальшої програмної реалізації в наступному розділі.

3 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМНОЇ СИСТЕМИ РЕЗЮМУВАННЯ ДОКУМЕНТІВ

Після розробки методу резюмування документів великого обсягу наступним етапом є побудова програмної системи, у межах якої цей метод може бути реалізований на практиці.

У цьому розділі розглянуто архітектуру програмної системи, склад її основних компонентів, їхню взаємодію під час обробки документа, а також технічні особливості реалізації. Основну увагу приділено організації проходження документа через систему, від завантаження користувачем до отримання готового резюме.

3.1 Архітектура програмної системи резюмування та взаємодія її компонентів

Розроблена система резюмування документів побудована за модульним принципом [34]. Такий підхід дає змогу розділити окремі функції системи між самостійними компонентами і спростити як її реалізацію, так і подальший розвиток.

Система складається з інтерфейсу користувача, серверної частини, підсистеми автентифікації та керування доступом, модуля резюмування і підсистеми збереження даних. Кожен із цих компонентів відповідає за окрему частину загального процесу, а разом вони утворюють єдиний цикл роботи з документом – від завантаження до отримання готового резюме.

Основні принципи проектування архітектури програмної системи резюмування документів:

1) модульність побудови системи – архітектура системи побудована за модульним принципом, відповідно до якого загальна функціональність

поділяється на окремі самостійні компоненти. Такий підхід спрощує реалізацію системи, її тестування, супровід і подальше вдосконалення;

2) розподіл відповідальності між компонентами – кожен компонент системи виконує чітко визначені функції. Інтерфейс користувача відповідає за взаємодію з користувачем, серверна частина – за координацію обробки запитів, підсистема аутентифікації – за безпеку та контроль доступу, модуль резюмування – за змістове опрацювання документа, а підсистема збереження даних – за роботу з файлами та даними користувачів;
3) централізація керування через серверну частину – серверна частина виступає центральним координаційним вузлом системи[35]. Саме через неї проходять основні потоки даних, що дозволяє уникнути прямої взаємодії між користувацьким інтерфейсом і внутрішніми обчислювальними модулями та забезпечує цілісність роботи системи;
4) відокремлення користувацького рівня від рівня обробки даних – інтерфейс користувача не виконує внутрішнього аналізу документа, а лише забезпечує доступ до функцій системи. Це дозволяє відокремити логіку представлення даних від логіки їх обробки, що позитивно впливає на керованість архітектури;
5) відокремлення підсистеми безпеки від прикладної логіки – функції аутентифікації та керування доступом винесені в окремий компонент. Такий підхід забезпечує незалежність механізмів безпеки від основного процесу резюмування та дозволяє окремо керувати правами доступу користувачів до документів;
6) ізоляція модуля резюмування як ключового обчислювального компонента - модуль резюмування реалізований як окрема підсистема, у межах якої виконується основне змістове опрацювання документа [36]. Це дає можливість змінювати, вдосконалювати або замінювати алгоритми резюмування без необхідності перебудови всієї програмної системи;
7) підтримка розширюваності системи – архітектура спроєктована таким чином, щоб окремі компоненти можна було модифікувати або доповнювати

новими функціями без суттєвого впливу на інші частини системи. Це створює основу для подальшого розвитку програмного забезпечення.

Загальну схема архітектури програмної системи резюмування документів зображено на рисунку 3.1.

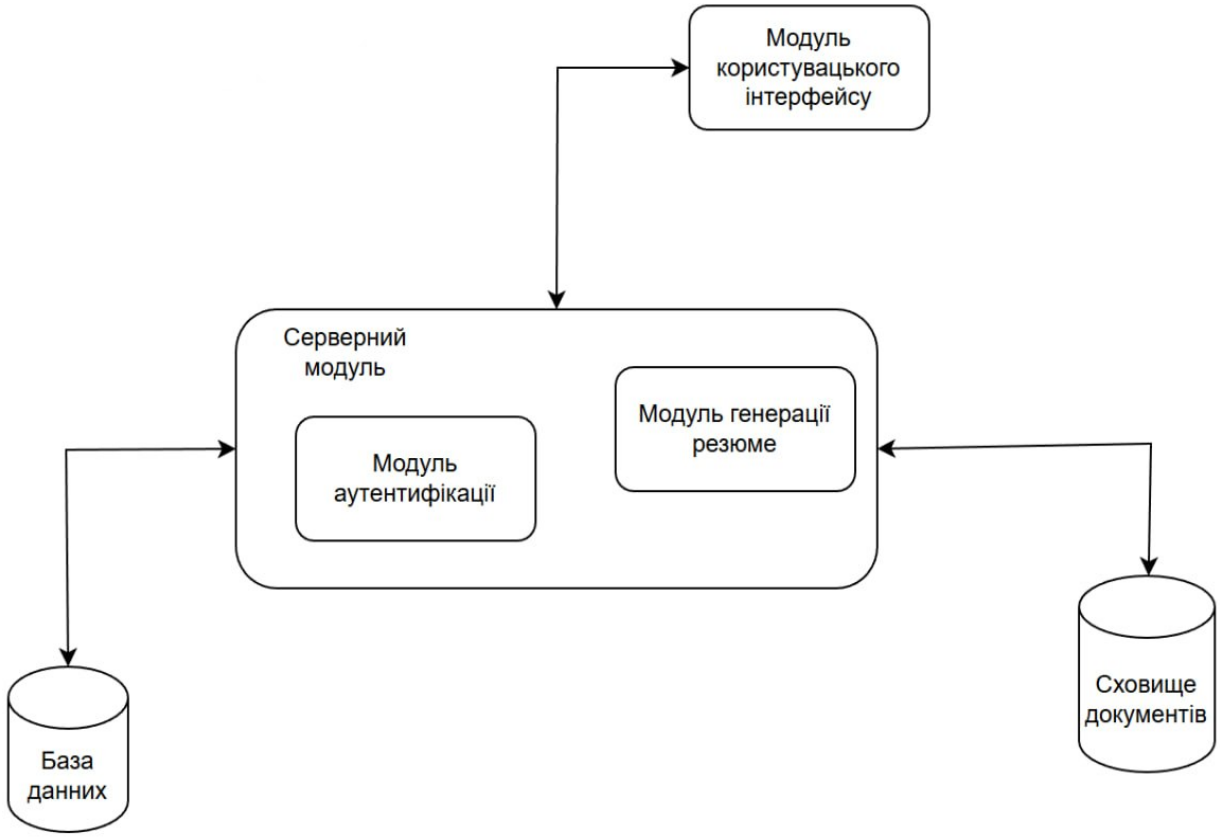

Рисунок 3.1 – Загальна схема архітектури програмної системи резюмування документів

До складу системи входять такі основні компоненти:

– інтерфейс користувача, через який здійснюється вхід до системи, завантаження документа, задання параметрів обробки, запуск побудови резюме та перегляд результату;
– серверна частина, яка координує роботу інших модулів, обробляє запити від інтерфейсу користувача та забезпечує передавання даних між окремими частинами системи;

- підсистема аутентифікації та керування доступом, що використовується для перевірки користувача і контролю доступу до документів;
- модуль резюмування, у якому виконується структурно-семантичний поділ документа відповідно до запропонованого методу та формується фінальне резюме.

Інтерфейс користувача є зовнішньою частиною системи, через яку відбувається вся взаємодія з користувачем. Саме на цьому рівні користувач виконує вхід до системи, завантажує документ, задає параметри обробки, ініціює запуск побудови резюме та переглядає отриманий результат. Інтерфейс не виконує внутрішнього змістового аналізу документа, а забезпечує доступ до функцій системи.

Серверна частина є центральним компонентом системи. Вона координує роботу інших модулів, обробляє запити від інтерфейсу користувача та забезпечує передавання даних між окремими частинами системи. Також координує роботу з сховищем документів та базою даних.

Через сервер проходить основний потік даних, тому саме він поєднує між собою інтерфейс користувача, підсистему автентифікації та модуль резюмування. Така побудова спрощує керування системою і дозволяє уникнути прямого зв’язку між користувацьким рівнем та внутрішніми обчислювальними модулями.

Модуль аутентифікації та керування доступом потрібна для перевірки користувача і контролю доступу до документів. Винесення функцій аутентифікації в окремий компонент дозволяє відокремити питання безпеки від логіки опрацювання документа.

У результаті система стає більш керованою, а права доступу до документів і побудованих резюме можна контролювати окремо. Дані користувачів зберігаються в базі даних.

Ключовим внутрішнім компонентом системи є модуль резюмування. Саме в ньому виконується основне опрацювання документа відповідно до методу, описаного в попередньому розділі. На вхід цей модуль отримує текст документа та параметри запуску, а на виході формує фінальне резюме.

Відокремлення модуля резюмування в самостійний компонент дає змогу вдосконалювати алгоритм побудови резюме без зміни інших частин системи. Документ для резюмування модуль загружає з сховища документів.

Побудова системи за такою схемою дає змогу розділити функції між окремими рівнями: інтерфейс відповідає за взаємодію з користувачем, серверна частина – за координацію роботи системи, підсистема доступу – за безпеку, модуль резюмування – за змістове опрацювання документа. Це спрощує розробку та відладку системи, оскільки окремі компоненти можна змінювати або розширювати без повної перебудови всієї архітектури.

3.2 Опис програмної реалізації системи

Програмна реалізація системи виконувалася з дотриманням загальноприйнятих принципів якісного проєктування програмного забезпечення. Під час розробки особлива увага приділялася модульності, розподілу відповідальності між компонентами, мінімізації жорстких залежностей та можливості подальшого розширення системи без повного переписування її окремих частин.

У побудові програми враховувались ідеї, близькі до принципів SOLID, зокрема орієнтація кожного модуля на чітко визначене коло функцій, відокремлення інтерфейсного, серверного та алгоритмічного рівнів, а також побудова компонентів таким чином, щоб їх можна було змінювати або замінювати з мінімальним впливом на інші частини системи.

Такий підхід дозволив зробити систему більш зрозумілою, керованою, придатною до тестування та зручною для подальшого розвитку.

Були також враховані принципи, а саме слабкої зв'язності і високої згуртованості.

3.2.1 Опис реалізації модуля резюмування

Модуль резюмування в системі реалізовано мовою Python у вигляді окремого модульного пакета. Діаграма основних класів розробленого модуля представлена на рисунку 3.2.

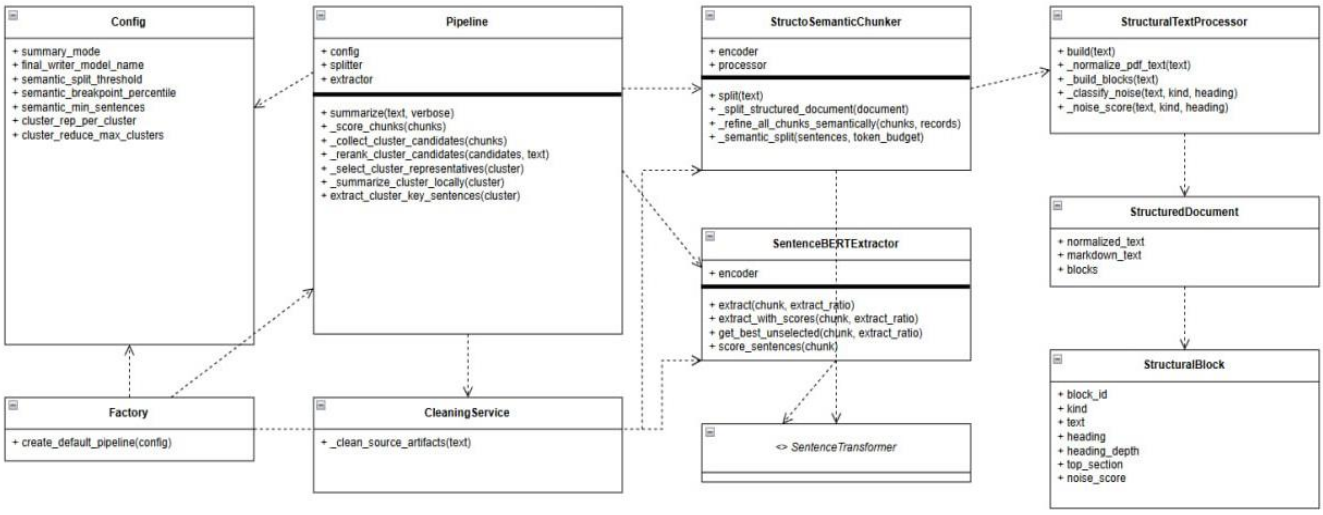

Рисунок 3.2 – Діаграма основних класів модулю резюмування

Центральним елементом є клас Pipeline, який координує повний цикл обробки документа: оцінювання фрагментів, відбір кандидатів, їх додаткове ранжування, кластеризацію, вибір репрезентативних елементів і побудову локальних резюме.

Побудова конвеєра виконується через Factory на основі параметрів, заданих у класі Config. За структурно-семантичний поділ відповідає StructoSemanticChunker, який використовує StructuralTextProcessor для формування внутрішнього структурного подання документа. Результатом роботи StructuralTextProcessor є об’єкти StructuredDocument і StructuralBlock, у яких зберігаються нормалізований текст документа, набір структурних блоків і їхні основні атрибути.

Оцінювання речень і фрагментів виконує SentenceBERTExtractor, а побудову семантичних подань забезпечує SentenceTransformer.

Основні класи системи:

– Config – зберігає параметри конфігурації конвеєра;
– Pipeline – координує основні етапи обробки документа;
– SentenceBERTExtractor – оцінює речення та виділяє опорні елементи;
– Factory – створює та налаштовує конвеєр обробки;
– StructoSemanticChunker – виконує структурно-семантичний поділ документу;
– SentenceTransformer – забезпечує побудову семантичних подань тексту.

3.2.2 Опис реалізації серверної частини

Серверну частину системи реалізовано на платформі ASP.NET Core. Її основним призначенням є організація взаємодії між клієнтським інтерфейсом, файловим сховищем та Python-модулем резюмування. На рисунку 3.3 зображена діаграма класів серверної частини.

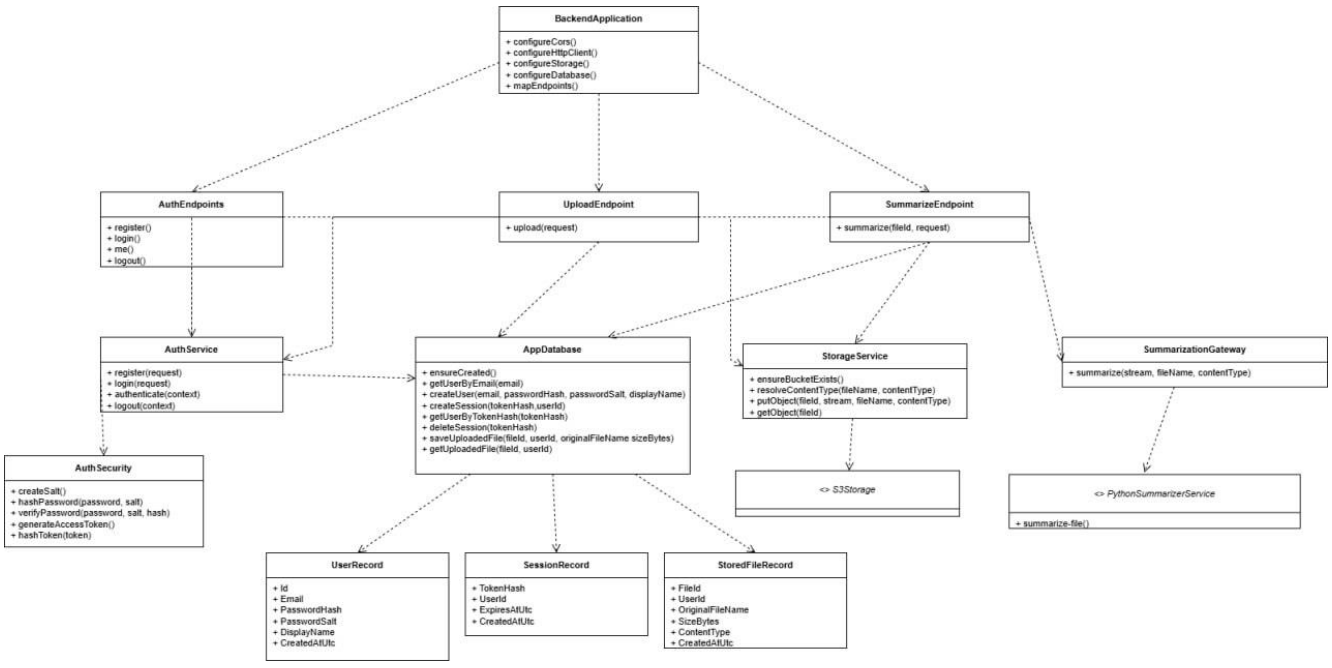

Рисунок 3.3 – Діаграма класів серверної частини розробленої системи

На рівні прикладного інтерфейсу виділено три основні групи кінцевих точок: AuthEndpoints, UploadEndpoint і SummarizeEndpoint. Перша група відповідає за реєстрацію, вхід, вихід і отримання даних поточного користувача, друга за завантаження документа, а третя за запуск процедури резюмування.

Логіка аутентифікації винесена в AuthService, який працює разом із допоміжним класом AuthSecurity. Ці компоненти забезпечують перевірку облікових даних, створення сесій, хешування паролів і токенів доступу. Робота зі структурованими службовими даними реалізована через AppDatabase, який взаємодіє із записами UserRecord, SessionRecord і StoredFileRecordi використовується для збереження користувачів, активних сесій і метаданих завантажених файлів, а не для збереження самих документів.

Робота з файловим сховищем реалізована через StorageService, який взаємодіє з S3-сумісним сховищем і відповідає за збереження та отримання документів за їх ідентифікатором.

Запуск алгоритмічного модуля резюмування виконується через SummarizationGateway, який у поточній реалізації передає запит до PythonSummarizerService. Це дозволяє відокремити серверну прикладну логіку від модуля резюмування.

Окремо в структурі виділено записи UserRecord, SessionRecord і StoredFileRecord, які представляють відповідно користувача, сесію та метадані завантаженого файла.

Основні класи серверної частини:

- AuthEndpoints – кінцеві точки аутентифікації;
- UploadEndpoint – кінцева точка завантаження документа;
- SummarizeEndpoint – кінцева точка запуску резюмування;
- AuthService – логіка роботи з користувачами та сесіями;
- AppDatabase – доступ до службових даних системи;
- StorageService – робота з файловим сховищем;
- SummarizationGateway – передавання запиту до модуля резюмування.

3.2.3 Опис реалізації клієнтської частини

Клієнтська частина системи реалізована як веб-застосунок на Angular [37]. Вона призначена для організації безпосередньої взаємодії користувача із системою резюмування. Через цей інтерфейс користувач може вибрати документ, завантажити його в систему, ініціювати побудову резюме та переглянути результат разом зі службовою статистикою. Діаграма класів клієнтської частини зображена на рисунку 3.4.

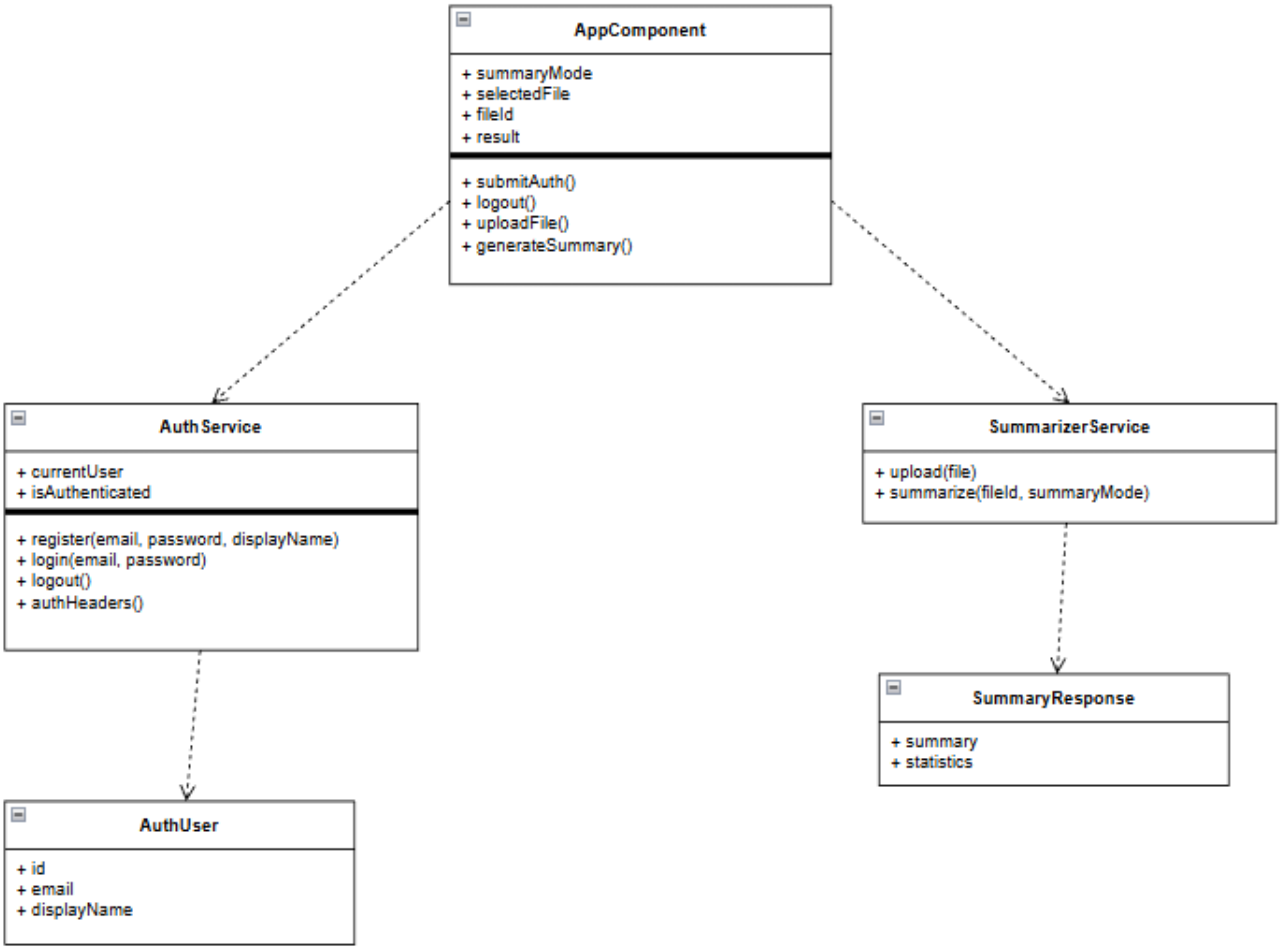

Рисунок 3.4 – Діаграма класів клієнтської частини

Центральним компонентом є AppComponent, який відповідає за взаємодію користувача з веб-інтерфейсом, зокрема за вибір режиму резюмування, завантаження документа, запуск побудови резюме та відображення отриманого результату. Для забезпечення аутентифікації використовується AuthService, який

забезпечує реєстрацію, вхід, вихід із системи та збереження інформації про поточного користувача у вигляді об’єкта AuthUser.

Взаємодія із серверною частиною при завантаженні документа та запуску резюмування здійснюється через SummarizerService, а результат обробки подається у вигляді структури SummaryResponse, що містить текст резюме та супровідні статистичні характеристики.

3.2.4 Опис реалізації збереження даних

Підсистема збереження даних у поточній реалізації орієнтована насамперед на роботу із завантаженими документами. Для цього використовується S3-сумісне об’єктне сховище [38], доступ до якого з серверної частини реалізується через бібліотеку AWSSDK.S3.

Під час локальної розробки та тестування використовується LocalStack, який дозволяє емулювати середовище об’єктного сховища без потреби в реальному зовнішньому хмарному сервісі.

Обране рішення має наступні переваги:

– об’єктне сховище дозволяє відокремити етап завантаження документа від етапу його подальшої обробки;
– реалізація створює основу для переходу від локального прототипу до більш масштабованого розгортання.

У серверній частині системи база даних використовується для збереження службових відомостей, пов’язаних з обліковими записами користувачів, активними сесіями та завантаженими документами [39].

На відміну від файлового сховища, де розміщуються самі файли, база даних призначена для роботи зі структурованими метаданими і не містить повного вмісту документів.

Структуру бази даних серверної частини системи наведено на рисунку 3.5.

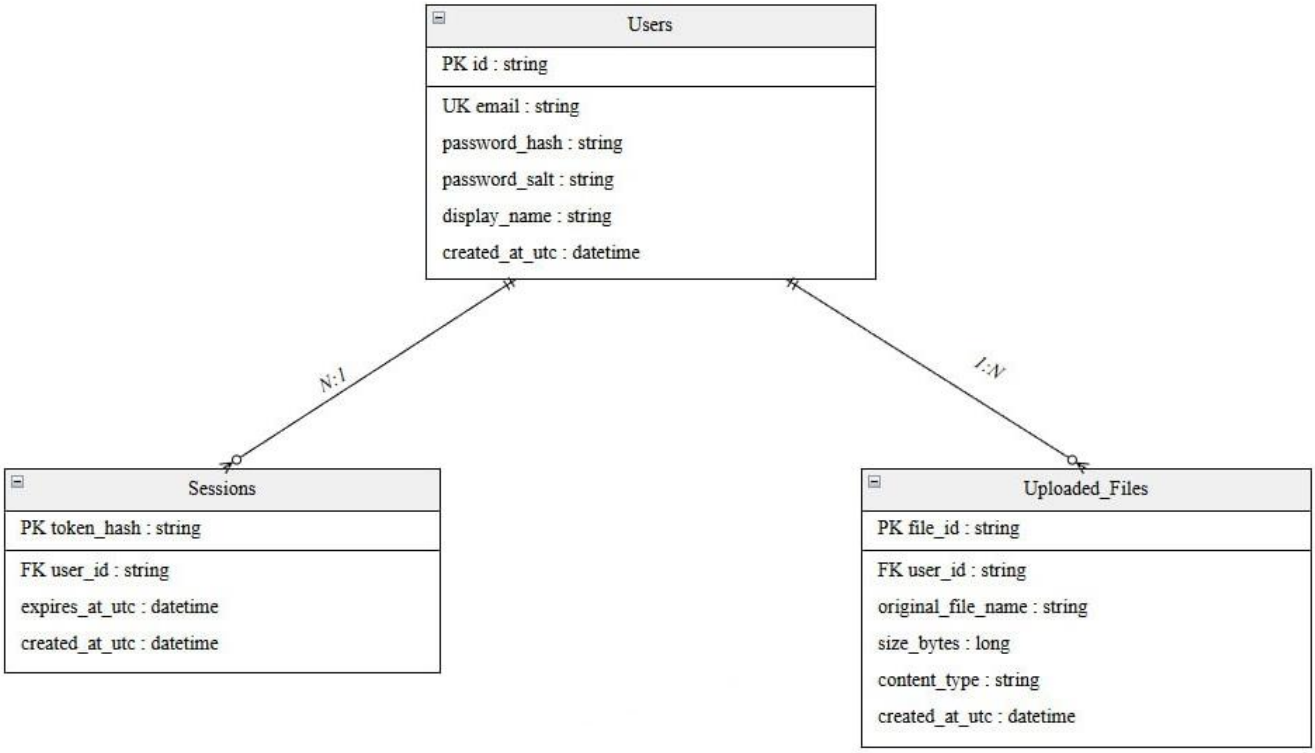

Рисунок 3.5 – Схема структури бази даних серверної частини системи

Таблиця users зберігає облікові записи користувачів і містить ідентифікатор, адресу електронної пошти, хеш пароля, сіль для хешування, відображуване ім’я та дату створення запису.

Таблиця sessions призначена для роботи з активними сесіями доступу. У ній зберігаються хеш токена, ідентифікатор користувача, час завершення дії сесії та дата її створення. Це дозволяє виконувати перевірку користувача за токеном доступу, не зберігаючи сам токен у відкритому вигляді.

Таблиця uploaded_files використовується для збереження відомостей про завантажені документи. У ній містяться ідентифікатор файла, посилання на користувача, оригінальна назва файла, розмір, тип вмісту та дата завантаження. Самі документи в базі даних не зберігаються, оскільки їх фізичне розміщення виконується у S3-сумісному об’єктному сховищі.

3.3 Деталізація взаємодії компонентів системи

У цьому підрозділі розглянуто детальніше, яким чином окремі компоненти програмної системи взаємодіють між собою під час виконання основних функцій. Основна увага приділяється послідовності обміну даними між клієнтською частиною, серверним рівнем, підсистемою збереження даних та модулем резюмування, а також ролі кожного з цих компонентів у загальному процесі обробки документа.

3.3.1 Процес побудови резюме

Діаграма послідовностей показує, як об’єкти або компоненти системи взаємодіють між собою в часі через обмін повідомленнями для виконання певного сценарію. На рисунку 3.6 зображена діаграма послідовностей взаємодії компонентів системи та користувача.

Робота системи починається з дій користувача в інтерфейсі. Користувач проходить автентифікацію, після чого завантажує документ, задає параметри обробки та ініціює запуск побудови резюме. Після цього інтерфейс формує запит і передає його до серверної частини.

Отримавши запит, сервер звертається до підсистеми аутентифікації та керування доступом для перевірки прав користувача. Якщо доступ дозволено, сервер виконує перевірку вхідних даних, зокрема формату документа та коректності переданих параметрів.

Після успішної перевірки сервер передає документ і параметри запуску до модуля резюмування.

У модулі резюмування виконується основне опрацювання документа. На цьому етапі система виконує структурно-семантичний поділ тексту, кластеризацію змістово близьких фрагментів, визначення ключових фактів, локальне резюмування окремих груп і побудову фінального глобального резюме.

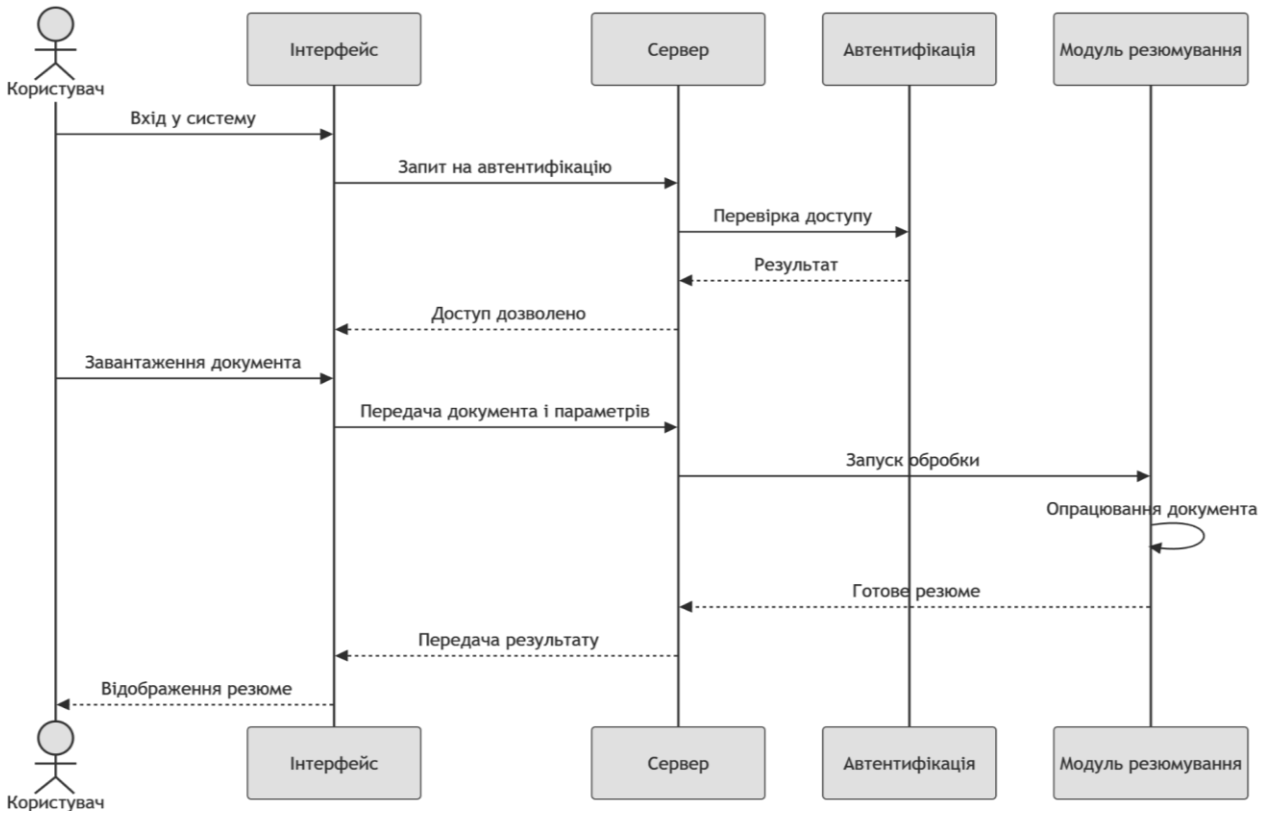

Рисунок 3.6 – Sequence-діаграма взаємодії компонентів програмної системи резюмування

Після завершення обробки модуль повертає результат серверній частині. На завершальному етапі інтерфейс відображає користувачу побудоване резюме. За потреби користувач може змінити параметри запуску або повторно обробити документ, після чого описана послідовність взаємодій виконується знову.

3.3.2 Діаграма діяльності процесу резюмування

У попередніх підрозділах було розглянуто основні компоненти програмної системи та їхнє призначення. Окремо було описано засоби роботи з документами, вибір режиму резюмування та формування підсумкового результату.

І для цілісного уявлення про функціонування системи важливо розглянути послідовність їхньої спільної роботи в межах одного користувацького сценарію.

У цьому сценарії поєднуються дії користувача, пов’язані із завантаженням документа та запуском резюмування, і внутрішні дії системи, що забезпечують збереження документа, його обробку та повернення готового результату.

Узагальнену схему роботи програмної системи резюмування наведено на рисунку 3.7.

Рисунок 3.7 – Узагальнена схема роботи програмної системи резюмування

Спочатку користувач виконує вхід у систему, після чого завантажує документ. Далі документ зберігається в системі, користувач обирає режим

резюмування та запускає побудову резюме. Після цього виконується обробка документа, формування підсумкового резюме та передавання результату користувачу для подальшого перегляду.

3.4 Основні технічні проблеми роботи системи та шляхи їх подолання

Побудова програмної системи резюмування документів великого обсягу пов'язана з низкою технічних труднощів. Частина з них виникла на рівні самого методу опрацювання тексту вибір способу поділу документа на блоки, принципи відбору найважливіших фрагментів, формування вхідного запиту для генеративної моделі. Інша група проблем стосується прикладного рівня системи: розмежування функцій між клієнтською та серверною частинами, аутентифікація, збереження даних.

Нижче розглянуто кожну з цих проблем, а також архітектурні рішення, що були прийняті для їх усунення в процесі розроблення системи:

- вибір підходу до сегментації документа: від поділу з перекриттям до структурно-семантичного поділу;
- відбір фрагментів, придатних для резюмування, на противагу тематично центральним;
- вибір архітектурного розміщення етапу фільтрації шуму;
- формування компактного проміжного представлення тексту для генеративної моделі;
- кластеризація та агрегація результатів: проблема подвійного резюмування;
- нерівномірний розподіл важливої інформації по документу;
- дублювання змісту в довгих документах;
- довжина вихідного резюме та калібрування обмежень;

– прикладні технічні проблеми системного рівня.

Першою проблемою стала сегментація довгих документів. На початковому етапі і в опублікованій статті [16] використовувався поділ із перекриттям: текст ділився на блоки фіксованої довжини, причому сусідні блоки перекривалися на задану кількість символів або токенів. Такий підхід був простим у реалізації, однак мав суттєві недоліки.

По-перше, він не враховував логічну структуру документа і часто розривав завершені смислові одиниці - думку, аргумент або пояснення – на межі двох блоків.

По-друге, через перекриття в сусідніх блоках дублювалася одна й та сама інформація, що призводило до надлишку даних без покращення якості.

По-третє, на наступний етап відбору часто потрапляли або неповні факти, або майже ідентичні фрагменти.

Для усунення цих недоліків було здійснено перехід до структурного поділу, при якому межі фрагментів визначалися структурними елементами тексту: заголовками, абзацами, списками та межами секцій.

Такий підхід дозволив формувати природніші фрагменти і зменшив штучне розривання змісту. Проте згодом стало очевидно, що структурний поділ теж не є достатнім, фізичні межі тексту не гарантують його смислової однорідності всередині блоку.

Наступним кроком стало запровадження структурно-семантичного поділу, який поєднує два критерії одночасно [40]. Спочатку документ розбивається за структурними межами, визначеними заголовками, абзацами, списками та секціями. Після цього великі або потенційно неоднорідні фрагменти додатково уточнюються за семантичними межами на основі змістової близькості сусідніх речень. Такий підхід дає змогу краще зберігати природні межі тексту та водночас зменшувати ймовірність того, що в одному блоці будуть поєднані різнорідні змістові частини.

Якщо сусідні речення або невеликі текстові фрагменти залишаються змістово пов’язаними, вони зберігаються в межах однієї змістової одиниці; там, де тема або риторична функція змінюється, встановлюється межа.

Завдяки цьому блоки, що передаються на подальшу обробку, краще відповідають локальним змістовим одиницям документа, хоча остаточне зменшення шуму досягається також на наступних етапах відбору та фільтрації.

У таблиці 3.1 наведено порівняльний аналіз трьох підходів до поділу документа на блоки, що застосовувались на різних етапах розроблення системи.

Таблиця 3.1 – Порівняння підходів до поділу документа на блоки

Підхід	Принцип	Переваги	Недоліки
Поділ з перекриттям	Фіксована кількість токенів, сусідні блоки перекриваються на задану кількість символів	Простота реалізації, передбачуваний розмір блоку	Ігнорує структуру, дублювання на межах, розривання смислових одиниць
Структурний поділ	Межі абзаців, заголовків, списків і секцій документа	Краще зберігає природні межі фрагментів	Залежить від форматування, нерівномірний розмір блоків, шум у структурних блоках
Структурно-семантичний поділ	Структурні межі + семантична схожість сусідніх фрагментів за векторними представленнями	Тематична однорідність блоків, краще контрольований розмір, менше дублювання	Вища обчислювальна складність, потребує векторних представлень

Наступною проблемою стала своєчасність фільтрації шуму в архітектурі опрацювання документа. На одному з попередніх етапів розроблення фільтрація

виконувалася занадто рано, ще до завершення сегментації, що створювало ризик передчасного відкидання змістовних фрагментів.

Це показало, що архітектурно правильніше розділяти сегментацію і фільтрацію на два незалежних етапи: сегментація повинна насамперед якісно розбивати документ на змістові блоки, а рішення про те, що є корисним і що є шумом, слід приймати пізніше – на окремому етапі оцінювання та відбору.

Третьою проблемою стало формування проміжного подання документа для генеративної моделі [41]. Надмірне стискання проміжного подання призводило до втрати локальної зв'язності усередині фрагментів і послаблення природної послідовності фактів.

Як виявилось, доцільніше після структурно-семантичного поділу документа виконувати кластеризацію змістово близьких фрагментів, у межах кожного кластера обирати репрезентативні фрагменти, далі виділяти з них короткі ключові речення як змістові опори, формувати локальні резюме окремих груп, а вже на завершальному етапі будувати глобальне резюме всього документа. Такий підхід краще зберігає змістову цілісність тексту і зменшує втрати важливої інформації.

Четвертою проблемою стала кластеризація фрагментів та агрегація локальних результатів. Кластеризація разом із подальшим вибором репрезентативних фрагментів розглядалась як спосіб ширшого охоплення довгого документа адже групування тематично близьких фрагментів дозволяє відбирати представницькі блоки з різних частин тексту, зменшуючи ризик того, що резюме буде побудовано лише на основі початкових розділів. Проте, виникло декілька проблем:

- кластери добре відображали теми документа, але не завжди вказували на фрагменти, найбільш важливі для резюмування;
- вибір представницького фрагмента всередині кластера виявився критичним: якщо ним ставав другорядний або технічний блок, ця помилка лише посилювалась на наступних етапах.

У поточній архітектурі ці труднощі враховуються шляхом поєднання кластеризації з подальшим вибором репрезентативних фрагментів та виділенням коротких ключових речень як змістових опор.

Проблемою також може бути дублювання змісту у довгих документах [42]. У довгих документах одна й та сама думка може повторюватися у вступі, поясненні та висновках. Якщо система не відсіює надмірно подібні фрагменти, генеративна модель отримує зміщене уявлення про важливість окремих тем і схильна до надмірного повторення тих самих тез у вихідному тексті.

Для розв'язання цієї проблеми у системі використовується смислове порівняння блоків на основі їхніх векторних представлень із подальшим усуненням надмірно подібних варіантів.

Восьмою проблемою стала довжина вихідного резюме та калібрування обмежень на генерацію [43].

Під час експериментів з'ясувалося, що навіть за наявності релевантних фрагментів система може генерувати занадто короткі тексти через надто жорсткі обмеження на довжину локальних і фінальних резюме. У такому випадку модель не мала достатнього простору, щоб охопити всі важливі аспекти документа: результуючий текст зводився до кількох речень і не відповідав очікуваному рівню деталізації.

Тому окремим напрямом налаштування системи стало коригування меж довжини виходу та ослаблення надмірно жорстких обмежень на проміжних етапах генерації. Підбір параметрів здійснювався на основі якісного аналізу вихідних текстів.

Також варто зазначити про технічні проблеми системного рівня. До них належать перевірка доступу до даних, збереження завантажених документів і службових метаданих, а також розмежування функцій між клієнтською та серверною частинами.

Для розв'язання цих проблем у системі реалізовано механізм аутентифікації на основі токенів доступу, реляційну базу даних для зберігання користувачів, сесій

і метаданих завантажених документів, а також чіткий поділ на програмний інтерфейс прикладного рівня та клієнтський застосунок.

Висновки до розділу 3

У третьому розділі побудовано архітектуру програмної системи, призначеної для практичного використання розробленого методу резюмування документів. Визначено склад основних компонентів системи, їхні функції та порядок взаємодії під час проходження документа від етапу завантаження до отримання підсумкового тексту.

Обрана структура системи передбачає поділ на клієнтський рівень, серверний рівень, підсистему аутентифікації, модуль резюмування та засоби збереження даних, що дозволяє розмежувати відповідальність між окремими частинами програмного рішення.

У межах розділу також описано програмну реалізацію окремих складових системи. Алгоритмічне ядро побудовано у вигляді окремого Python-модуля, де реалізовано послідовність обробки документа відповідно до запропонованого методу. Серверну частину виконано на ASP.NET Core, що дало змогу організувати централізовану обробку запитів, зв’язок із файловим сховищем, базою даних та зовнішнім модулем резюмування.

Клієнтську частину реалізовано як веб-застосунок на Angular, через який користувач отримує доступ до функцій завантаження документа, запуску побудови резюме та перегляду результатів.

У системі розмежовано зберігання самих документів та службових даних: файли передаються до S3-сумісного об’єктного сховища, тоді як відомості про користувачів, сесії та завантажені документи розміщуються в базі даних. Такий підхід спрощує керування доступом, дозволяє відокремити прикладну логіку від зберігання файлів і підвищує керованість системи.

4 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ

У цьому розділі розглядаються практичні аспекти реалізації розробленої системи резюмування документів, а також результати її експериментального дослідження. Основну увагу зосереджено на виборі засобів реалізації, побудові користувацького інтерфейсу та перевірці ефективності запропонованого методу на документах різної довжини. Також подано методику проведення експериментів, використані набори даних, критерії оцінювання та аналіз одержаних результатів.

4.1 Вибір засобів реалізації

Створення програмної системи для резюмування документів потребувало вибору такого набору інструментів, який дозволяє реалізувати всі ключові етапи запропонованого методу в межах одного програмного середовища. Пріоритет надавався таким засобам, які добре поєднуються між собою, підтримують експериментальну роботу з текстовими даними та дозволяють без значних труднощів інтегрувати кілька моделей у спільну систему.

При виборі засобів розробки за основу було взято такі критерії:

- підтримка роботи з методами обробки природної мови та моделями-трансформерами;
- наявність засобів для кластеризації та чисельної обробки даних;
- зручність інтеграції кількох моделей-трансформерів в одному програмному методі;
- регулярне оновлення програмних компонентів;
- придатність для проведення експериментів та оцінки якості результатів;
- можливість легкої модифікації при збільшенні розмірів розробленої системи;

- підтримка використання різних засобів виконання програмного коду, включаючи можливість запуску моделі-трансформера на відеокарті задля збільшення продуктивності;
- підтримка сучасних рішень в сфері сервісно-орієнтованого програмування;
- наявність засобів для створення зручного користувацького інтерфейсу;
- можливість інтеграції компонентів у єдину систему, при цьому зберігаючи їх відносно незалежними одна від однієї, гарантувавши правило “слабкої звязності;”
- зручність тестування.

В розділі 3 було подано загальну архітектуру, де ключовим модулем виступав модуль резюмування. Для його реалізації обрано мову Python. Вибір цієї мови зумовлений тим, що вона є одним з найпоширеніших інструментів при роботі з задачами машинного навчання та обробки природньої мови. В даній роботі Python використовується для реалізації алгоритму резюмування. Перевагою цієї мови є наявність великої кількості бібліотек, що дозволяють працювати з нейронними моделями, моделями-трансформерами, проводити математичні операції в межах одного програмного середовища.

До причин вибору Python належать:

- поширеність у задачах машинного навчання;
- наявність великої кількості спеціалізованих бібліотек для роботи з задачами обробки природньої мови;
- можливість швидко змінювати алгоритмічну частину розробленої системи.

Для роботи з нейронними мережами, а саме моделями-трансформерами було застосовано бібліотеку PyTorch. Ця бібліотека використовується як основа для виконання обчислень, побудовою векторних представлень тексту і генерацією резюме. Його особливістю є можливість використання відеокарти для обчислень, що пришвидшує роботу системи(адже процесор не заточений під виконання

великої кількості малих обчислень на відміну від відеокарти) і сумісності з іншими бібліотеками для обробки тексту.

Переваги:

– підтримка обчислень на відеокарті;
– зручна інтеграція з сучасними моделями-трансформерами;
– гнучкість при налаштуванні і використанні моделей.

Недоліки:

– потребує значних обчислювальних ресурсів при роботі з великиим моделями-трансформерами

Альтернативи які розглядались:

– Tensorflow – потужне середовище для роботи з нейронними мережами;
– JAX – менш поширений у системах такого типу.

Для розроблення програмного продукту PyTorch було обрано через його поширеність у задачах обробки природньої мови та зручність для проведення експериментів.

Transformers – це бібліотека для роботи з попередньо навченими трансформерними моделями в екосистемі HuggingFace. Вона надає готові засоби для завантаження, налаштування та використання сучасних генеративних і аналітичних моделей. У даній роботі ця бібліотека використовується для підключення моделі Qwen2.5-1.5B-Instruct, яка виконує локальне та фінальне узагальнення тексту.

Переваги:

– уніфікований інтерфейс для роботи з великою кількістю моделей;
– наявність готових засобів токенізації та генерації;
– спрощення інтеграції генеративних моделей у прикладну систему;
– сумісність із PyTorch.

Недоліки Transformers:

– велика кількість залежностей і значний обсяг середовища;
– при роботі з великими моделями висуває підвищені вимоги до ресурсів;

– складність налаштування може зростати зі збільшенням кількості моделей у системі.

Scikit-learn це бібліотека для класичних методів машинного навчання та аналізу даних. У даній роботі вона використовується для кластеризації текстових фрагментів. Це дозволяє групувати змістово близькі частини документа перед етапом локального узагальнення.

Переваги**:**

– наявність стабільних реалізацій класичних алгоритмів;

– зручність інтеграції з числовими поданнями тексту;

– простота налаштування параметрів;

– добра придатність до експериментальних досліджень.

Недоліки:

– не призначена для роботи з великими нейронними моделями;

– має обмеження при дуже великих обсягах даних;

– якість групування залежить від якості попереднього текстового подання.

Scikit-learn було обрано через простоту використання, стабільність і достатню придатність для експериментальної кластеризації текстових фрагментів.

ASP.NET Core це серверна платформа для створення веб-застосунків і прикладних сервісів. У межах даної роботи вона використовується для реалізації серверної частини системи, яка приймає документи від користувача, запускає процедуру резюмування та повертає результат через серверні кінцеві точки.

Таким чином, цей засіб забезпечує прикладний рівень взаємодії між інтерфейсом користувача та алгоритмічним модулем.

Переваги:

– придатність до побудови сучасних серверних застосунків;

– зручна організація прикладних кінцевих точок;

– зручна інтеграція із зовнішніми сервісами та файловими сховищами;

– чітке відокремлення серверної логіки від алгоритмічного модуля.

Недоліки:

– не призначений для реалізації самого алгоритму машинного навчання;

– вимагає окремої інтеграції з Python-модулем;

– збільшує загальну складність системи порівняно з локальним прототипом.

Альтернативи, що розглядалися:

– серверна частина на Python;

– Node.js для побудови веб-сервісу;

– інші серверні платформи загального призначення.

ASP.NET Core було обрано через його зручність для побудови серверного прикладного рівня, стабільність та добру придатність до веб-архітектури системи.

Angular це засіб побудови клієнтських веб-застосунків. У даній роботі він використовується для створення користувацького інтерфейсу, через який можна завантажити документ, запустити побудову резюме та переглянути результат. Він забезпечує практичний доступ до можливостей системи без потреби безпосередньо взаємодіяти з серверними запитами або алгоритмічним модулем.

Переваги:

– придатність до побудови сучасного веб-інтерфейсу;

– компонентна структура застосунку;

– зручна організація взаємодії із серверною частиною;

– добра підтримка масштабування інтерфейсу.

Недоліки:

– складніший для початкового освоєння, ніж простіші клієнтські засоби;

– не виконує алгоритмічної обробки документів, а лише забезпечує інтерфейс;

– вимагає окремої серверної частини для повноцінної роботи.

Альтернативи, що розглядалися:

– React;

– Vue.

Angular було обрано тому, що він дозволяє побудувати впорядкований клієнтський інтерфейс і добре поєднується із серверною частиною системи.

Для організації роботи із завантаженими документами у серверній частині було використано клієнт AWSSDK.S3, який забезпечує взаємодію з об’єктним

сховищем Amazon S3. Об’єктне сховище призначене для збереження файлів у вигляді окремих об’єктів із власними ідентифікаторами та службовими атрибутами. У межах даної роботи цей засіб використовується для збереження завантажених документів, їх подальшого отримання серверною частиною та передавання до модуля резюмування. Такий підхід дозволяє не зберігати файл в межах одного короткочасного запиту, а працювати з ним як з окремою сутністю системи.

Переваги AWSSDK.S3 / Amazon S3:

– зручне збереження файлів великого обсягу;

– доступ до документа за унікальним ідентифікатором;

– відокремлення файлового сховища від прикладної логіки;

– добра придатність до масштабування системи;

– спрощення повторного використання вже завантажених документів.

Недоліки:

– потребує окремого налаштування доступу та конфігурації сховища;

– ускладнює локальне розгортання порівняно зі звичайною файловою системою;

– додає ще один зовнішній компонент до загальної архітектури;

– може створювати додаткові затримки при роботі з великими файлами.

Альтернативи, що розглядалися:

– збереження документів у локальній файловій системі сервера;

– використання бази даних для збереження двійкових файлів;

– інші об’єктні сховища, сумісні з S3.

Для збереження службових даних системи було використано SQL Server. Це реляційна система керування базами даних, призначена для структурованого збереження інформації, виконання запитів, підтримки зв’язків між сутностями та забезпечення цілісності даних.

У межах даної роботи SQL Server використовується не для збереження самих документів, а для роботи з інформацією прикладного рівня, зокрема даними про користувачів, службові записи та інші сутності, що мають чітку структуру.

Переваги:

– надійне збереження структурованих даних;

– підтримка реляційної моделі та зв’язків між таблицями;

– зручність виконання запитів до службової інформації системи;

– підтримка транзакцій і контролю цілісності даних;

– добра інтеграція із серверною частиною на ASP.NET Core.

Недоліки SQL Server:

– менш придатний для збереження великих файлів, ніж спеціалізовані файлові сховища;

– потребує окремого адміністрування та налаштування;

– є важчим рішенням, ніж прості вбудовані бази даних;

– у невеликих системах може бути надлишковим за функціональністю.

Альтернативи, що розглядалися:

– PostgreSQL як інша повноцінна реляційна система;

– MySQL для простішого серверного застосування;

– SQLite для локального або спрощеного варіанта системи.

У даній роботі SQL Server було обрано через його надійність, реляційну модель збереження даних та зручну інтеграцію із серверною частиною застосунку.

4.2 Опис інтерфейсу розробленої системи

Розроблена програма містить декілька форм – форму логіну та реєстрації користувачів, форму завантаження документів та сторінку відображення результатів резюмування. На рисунку 4.1 зображена форма реєстрації користувача.

Форма реєстрації містить поля введення адреси електронної пошти, імені користувача та пароля, а також кнопку підтвердження створення облікового запису.

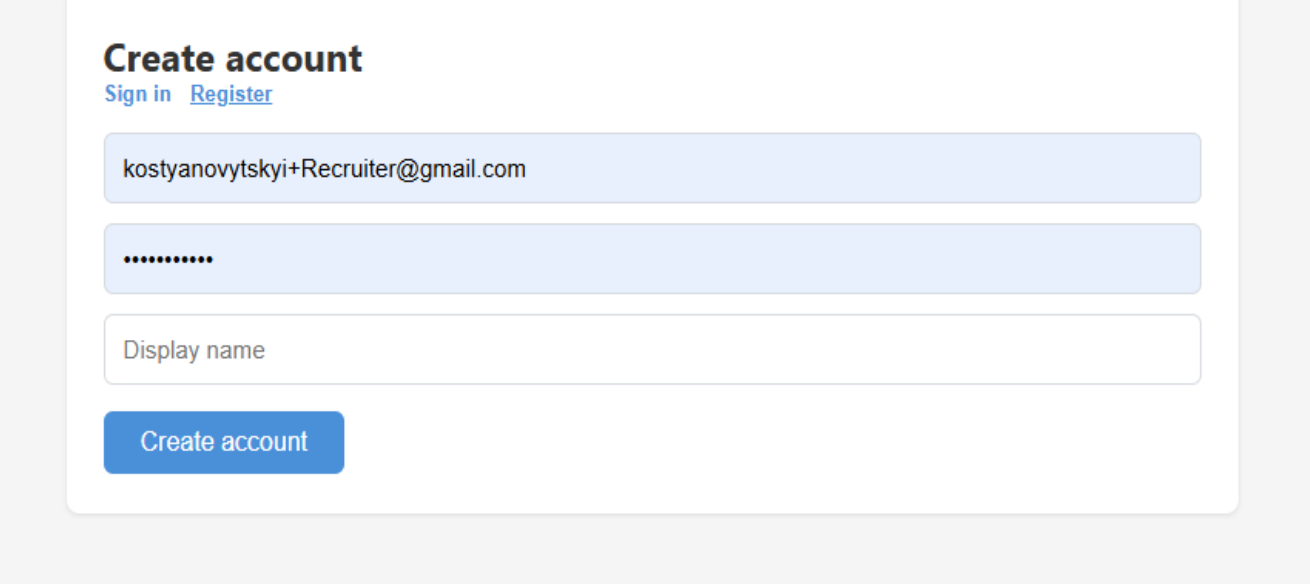

Рисунок 4.1 – Форма реєстрації користувача

Після створення облікового запису користувач переходить до форми входу. Саме через цей інтерфейс виконується аутентифікація та відкривається доступ до основної функціональності веб-застосунку.

Екран входу до системи наведено на рисунку 4.2.

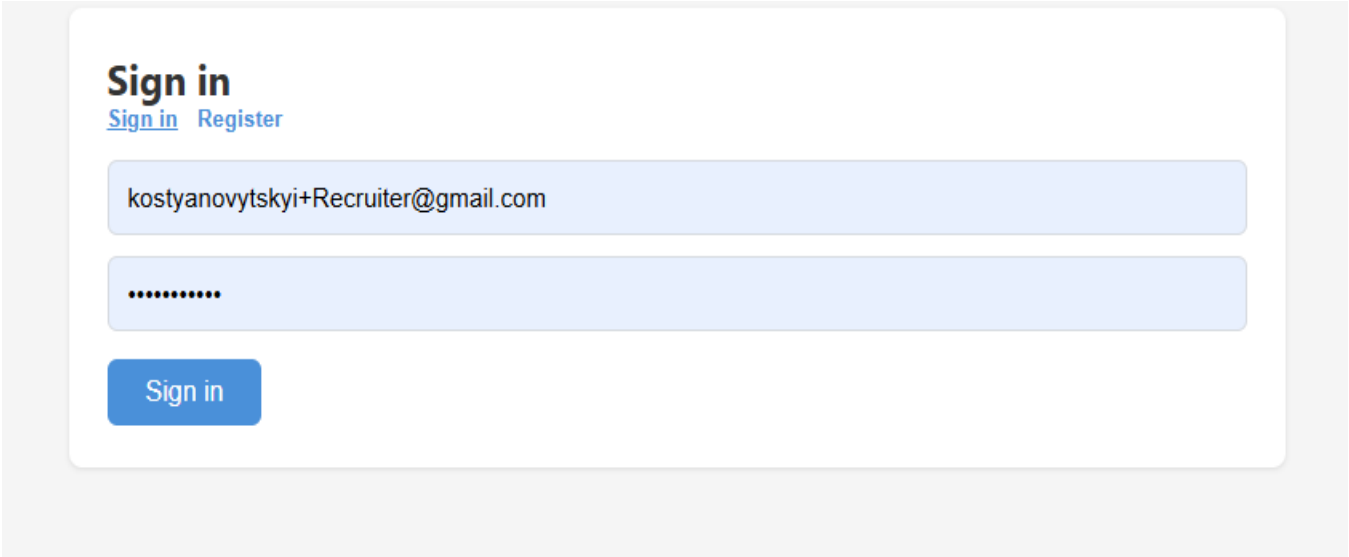

Рисунок 4.2 – Форма входу користувача

Форма входу містить поля введення адреси електронної пошти, пароля та кнопку підтвердження входу. У межах цього екрана виконуються дії, пов’язані з перевіркою облікових даних і початком роботи користувача в системі.

Після успішної аутентифікації відкривається основне робоче вікно системи. У цьому інтерфейсі зосереджено дії, пов’язані із завантаженням документа, вибором режиму резюмування та запуском побудови підсумкового тексту. Вигляд головного вікна системи після входу користувача наведено на рисунку 4.3.

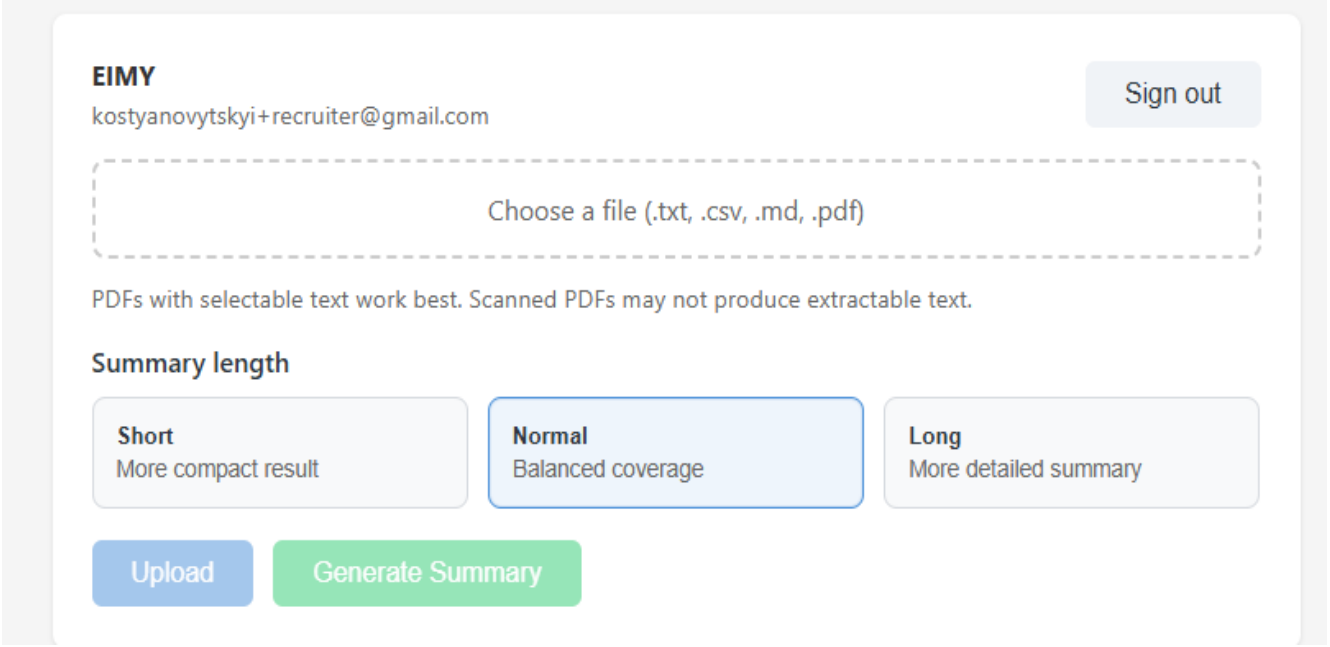

Рисунок 4.3 – Форма завантаження документа

Форма генерування резюме складається з таких компонентів: кнопка завантаження файлу (підтримуються формати .txt, .pdf, .docx), кнопка «Генерувати резюме», індикатор завантаження (spinner) під час обробки та блок виведення результату.

На рисунку 4.4 зображене згенероване резюме. Після того, як модель згенерувала резюме, клієнт отримує сповіщення від сервера, в якому подається як результат згенероване резюме.

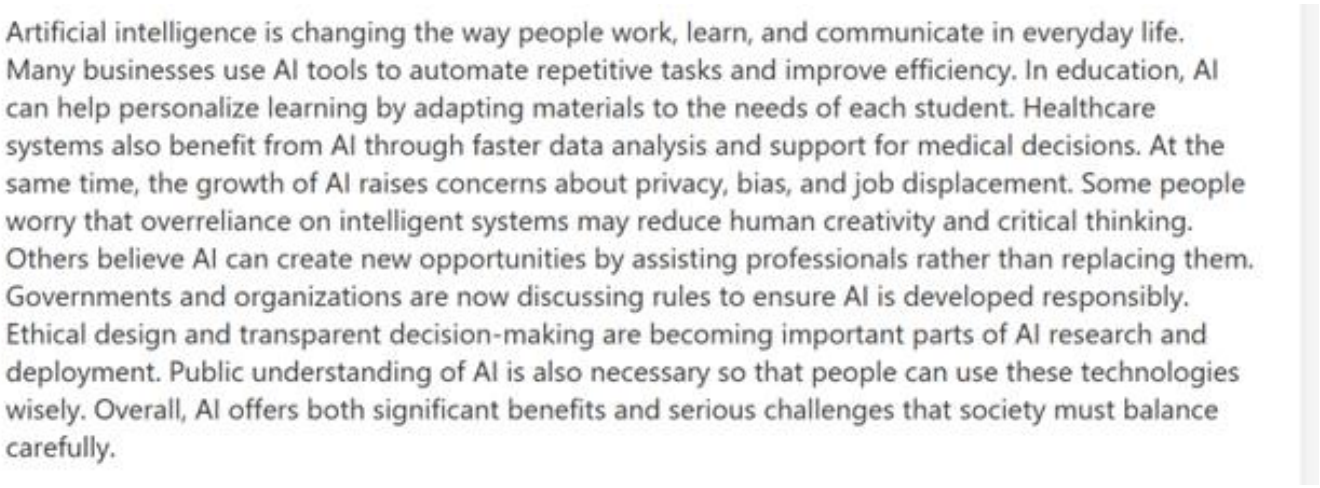

Рисунок 4.4 – Результат роботи системи

4.3 Проведення експерементальних досліджень

Для проведення експериментальних досліджень було використано бібліотеку Datasets, яка забезпечує завантаження, фільтрацію, підготовку та відтворюване формування вибірок документів. Її використання дозволило працювати зі стандартними наборами даних для задач резюмування, формувати фіксовані підвибірки документів, повторно запускати однакові серії експериментів та порівнювати різні варіанти системи на однакових даних.

Ця бібліотека суттєво спрощує завантаження та підготовку документів, зменшуючи кількість допоміжного коду. Також вона дозволяє формувати фіксовані вибірки, що забезпечує повторюваність результатів.

Серед альтернативних підходів розглядалися ручна підготовка документів, використання власних скриптів завантаження і фільтрації та робота лише з локальними файлами без окремої бібліотеки.

Проте такий підхід ускладнював би повторне відтворення вибірок і збільшував би обсяг допоміжної реалізації. Саме тому в роботі було обрано Datasets як засіб формування експериментальних підвибірок і організації дослідження.

4.3.1 Методика експериментального дослідження

Метою експериментального дослідження була перевірка ефективності розробленого методу резюмування довгих документів та його порівняння з прямим однорівневим резюмуванням генеративною моделлю. Для цього було організовано кілька серій експериментів, що відрізнялися довжиною документів та складністю вхідних текстів.

У дослідженні використовувалися документи з наборів ccdv/arxiv-summarization, NortheasternUniversity/big_patent, ccdv/govreport-summarization та kmfoda/booksum. Для багатодоменних експериментів використовувалися документи типів scientific, patent і government.

Для забезпечення коректності порівняння обидва варіанти методу запускалися на одних і тих самих фіксованих підвибірках, сформованих з однаковим значенням випадкового зерна. Це дозволило усунути вплив різниці між документами та зосередитися саме на порівнянні методів.

У межах дослідження було проведено три основні серії експериментів.

Перша серія стосувалася звичайних документів середньої довжини. Для неї було сформовано підвибірку з 75 документів, по 25 документів для кожного з трьох доменів: scientific, patent і government. На цій вибірці порівнювалися два варіанти: пряме резюмування просто модель Qwen та розроблений метод, який використовує модель Qwen.

Для другої серії було сформовано підвибірку з 30 документів, по 10 документів на домен, із довжиною тексту понад 15000 слів. Цей експеримент дозволяв оцінити, як обидва підходи поводяться в умовах, коли пряме резюмування стикається з більш вираженими обмеженнями довжини контексту.

Третя серія використовувалася для перевірки методу на довгих документах. Для цього було обрано підвибірку BookSum, у якій після фільтрації залишилося 30 документів із довжиною понад 50000 слово. Для третьої серії експериментів були також включені такі гібридні методи резюмування як кластеризація та підхід Map-Reduce де документ ділиться на частини і кожна частина резюмується окремо і потім весь текст резюмується знову разом.

Для оцінювання результатів використовувалися як лексичні, так і семантичні метрики. Основними метриками були ROUGE-1, ROUGE-2 та ROUGE-L, які характеризують ступінь лексичного перекриття між згенерованим і еталонним резюме. Додатково застосовувалася метрика BERTScore, яка дозволяє оцінювати семантичну близькість між текстами навіть за відсутності дослівного збігу. Крім того, у дослідженні враховувалися допоміжні показники: час виконання, середня довжина згенерованого резюме, показники зв'язності (coherence). Також аналізувався середній час побудови резюме.

Для експерименту спочатку формувалася фіксована підвибірка документів. Далі на ній окремо запускалися порівнювані варіанти системи. Після цього обчислювалися всі основні й допоміжні метрики, а результати агрегувалися як для всієї вибірки загалом, так і окремо за доменами. Умови експериментів наведені в таблиці 4.1.

Таблиця 4.1 – Основні серії експериментів

Набори даних	Кількість документів	Умова відбору
arXiv, BigPatent, GovReport	75	25 документів на домен
arXiv, BigPatent, GovReport	30	Понад 15000 слів
BookSum	30	Понад 50000 слів

4.3.2 Аналіз результатів експериментів

Результати експериментів показали, що ефективність запропонованого підходу залежить від довжини документа.

Для звичайних документів прямий метод і кластерний метод демонструють близькі результати, однак зі збільшенням довжини документа, тобто коли в вікно контексте вже не поміщається весь документ, переваги запропонованої схеми стають помітнішими.

На вибірці зі 75 звичайних документів розроблений метод отримав такі середні значення: ROUGE-1 = 0.3637, ROUGE-2 = 0.1047, ROUGE-L = 0.1892, BERTScore = 0.5913, середній час обробки 40.88 с на документ.

Прямий метод на тій самій вибірці показав ROUGE-1 = 0.3489, ROUGE-2 = 0.1299, ROUGE-L = 0.2082, BERTScore = 0.5970, середній час 61.68 с на документ. Результати на документах звичайної довжини наведені в таблиці 4.2.

Таблиця 4.2 – Результати на документах звичайної довжини

Метод	Rouge-1	Rouge-2	BERTScore	Час, с/док
Запропонований метод	**0.3637**	0.1047	0.5913	**40.88**
Пряме використання моделі	0.3489	0.1299	0.5970	61.88

На документах, довжина яких менше за вхідне вікно моделі, запропонований підхід уже показав вищий ROUGE-1, що свідчить про краще покриття змісту документа, а також виявився швидшим.

Водночас пряме використання моделі залишилось кращим за ROUGE-2, та BERTScore, тобто на середніх за довжиною документах він і далі краще зберігає локальні зв’язки та семантичну точність формулювань. Це означає, що на звичайних документах кластерний метод уже є конкурентним, але ще не дає безумовної переваги за всіма показниками.

Варто зазначити різницю по доменах. Для домену документів з державного сектору, запропонований метод виявився сильнішим за пряме використання моделі: ROUGE-1 = 0.4174 проти 0.2828, а BERTScore = 0.6019 проти 0.5813. Для научних і патентних доменів прямий підхід залишився дещо кращим.

Це може свідчити про те, що запропонований метод корисний там, де документ містить кілька віддалених змістових частин і де важливішим є широке охоплення, ніж відтворення локально компактного викладу.

На вибірці з 30 документів більше 15000 слів, запропонований метод показав ROUGE-1 = 0.3474, ROUGE-2 = 0.0993, ROUGE-L = 0.1777, BERTScore = 0.5966, середній час 56.24 с на документ. Прямий метод на тій самій вибірці досяг ROUGE-1 = 0.3645, ROUGE-2 = 0.1532, ROUGE-L = 0.2344, BERTScore = 0.5924, проте середній час його роботи склав 206.55 с на документ.

Тобто на довгих документах, але які ще поміщаються в контексне вікно моделі-трансформера пряме резюмування все ще утримує невелику перевагу за метриками якості, але ця перевага вже є значно меншою, ніж різниця в часі виконання.

Запропонований підхід працює приблизно у 3.7 раза швидше, при цьому його BERTScore трохи краще. Результати тестування з документами більше 15000 слів наведені в таблиці 4.3.

Таблиця 4.3 – Тестування з документами більше 15000 слів

Метод	ROUGE-1	ROUGE-2	ROUGE-L	BERTScore	Час, с/док
Запропонований метод	0.3474	0.0993	0.1777	**0.5966**	**56.24**
Прямий	0.3645	0.1532	0.2344	0.5924	206.55

У межах цього експерименту порівнювалися чотири варіанти побудови резюме – пряме використання генеративної моделі без попередньої багатоступеневої обробки, багатоступеневий підхід із послідовним локальним резюмуванням фрагментів і подальшим фінальним зведенням(Map-Reduce підхід), варіант із попередньою кластеризацією змістово близьких фрагментів та їх подальшим узагальненням, а також запропонований у роботі метод.

Тестування на наборі BookSum із 30 документів, довжина яких перевищувала 50000 слів, показало такі результати: на цій вибірці запропонований метод отримав ROUGE-1 = 0.2250, ROUGE-2 = 0.0431, ROUGE-L = 0.1071, BERTScore = 0.5216, а середній час обробки становив 104.11 с на документ.

Прямий підхід показав ROUGE-1 = 0.1548, ROUGE-2 = 0.0290, ROUGE-L = 0.0916, BERTScore = 0.4613, а середній час склав 198.51 с на документ.

Варіант із попередньою кластеризацією змістово близьких фрагментів досяг ROUGE-1 = 0.1971, ROUGE-2 = 0.0346, ROUGE-L = 0.1025, BERTScore = 0.4801 при середньому часі 44.11 с на документ. Варіант із послідовним багатокроковим локальним і фінальним узагальненням показав ROUGE-1 = 0.1582, ROUGE-2 = 0.0269, ROUGE-L = 0.0947, BERTScore = 0.4688, а середній час обробки становив 183.15 с на документ.

У даному випадку запропонований метод виявився найкращим за всіма основними метриками якості серед усіх порівнюваних варіантів. При цьому він поступався кластеризованому варіанту за швидкістю, однак забезпечував кращу якість підсумкового резюме.

У порівнянні з прямим підходом і багатокроковим локальним узагальненням запропонований метод одночасно показав і вищу якість, і менший час виконання. Це свідчить про доцільність використання запропонованого методу для резюмування довгих наративних документів, у яких важлива інформація розподілена по всьому тексту.

Результати тестування документів обсягом понад 50000 слів наведені в таблиці 4.4.

Таблиця 4.4 – Результати тестування документів з більше ніж 50000 слів

Метод	ROUGE-1	ROUGE-2	ROUGE-L	BERTScore	Час, с/док
Запропонований метод	**0.2250**	**0.0431**	**0.1071**	**0.5216**	104.11
Прямий	0.1548	0.0290	0.0916	0.4613	198.51
Map-reduce підхід	0.1582	0.0269	0.0947	0.4688	183.15
Кластеризація	0.1971	0.0346	0.1025	0.4801	44.11

Висновки до розділу 4

У четвертому розділі розглянуто практичну реалізацію розробленої системи резюмування документів та наведено результати її експериментального дослідження. У межах розділу обґрунтовано вибір засобів реалізації, описано інтерфейс створеної системи та показано, що програмне забезпечення забезпечує повний цикл роботи з документом: аутентифікацію користувача, завантаження файла, вибір режиму побудови резюме та відображення отриманого результату.

Було визначено методику проведення дослідження, підібрано набори документів різної довжини та виконано порівняння запропонованого підходу з кількома варіантами резюмування, зокрема з безпосереднім використанням генеративної моделі та іншими багатоступеневими методами обробки довгих документів.

Аналіз отриманих результатів показав, що розроблений багатоступеневий метод забезпечує конкурентоспроможну якість резюмування та водночас дозволяє зменшити середній час обробки документа.

Для частини експериментальних вибірок прямий підхід виявився дещо сильнішим за окремими метриками лексичного збігу, однак зі збільшенням довжини документа переваги запропонованого підходу ставали більш вираженими. На довгих документах він продемонстрував кращі результати за основними показниками якості.

Це свідчить про те, що багатоступенева схема є більш придатною для роботи з текстами великого обсягу, ніж безпосередня генерація резюме з повного документа.

ВИСНОВКИ

У ході виконання магістерської дисертації було опрацьовано основні етапи створення методу та програмної системи для автоматичного резюмування документів на основі моделей-трансформерів. Поставлена в роботі задача, пов’язана з покращенням якості узагальнення довгих текстів в умовах обмеженого контекстного вікна генеративних моделей, розв’язувалася шляхом побудови багатоступеневого гібридного підходу.

Запропонований підхід охоплює структурно-семантичний поділ документа, відбір змістово важливих фрагментів, кластеризацію тематично близьких частин тексту, виділення ключових фактів, побудову локальних резюме та формування підсумкового глобального резюме.

До основних результатів роботи належать: аналіз сучасних підходів до автоматичного резюмування довгих документів, розроблення власного гібридного методу, проєктування архітектури програмної системи, програмна реалізація системи із використанням Python, ASP.NET Core, Angular, S3-сумісного сховища та реляційної бази даних, проведення експериментального дослідження і порівняння результатів із базовими підходами.

У запропонованому методі документ подається як сукупність змістово та структурно пов’язаних блоків. Обробка включає сегментацію, виділення ключових тверджень, формування проміжного подання документа та побудову фінального резюме. Такий підхід зменшує навантаження на генеративну модель і забезпечує збереження важливих змістових частин документа під час роботи з довгими текстами.

Експериментальне дослідження охоплювало три серії випробувань. У першій використовувалися 75 документів із наборів ccdv/arxiv-summarization, NortheasternUniversity/big_patent і ccdv/govreport-summarization. У другій розглядалися 30 документів із тих самих доменів, але з довжиною понад 15000 слів. Третя серія проводилася на 30 документах набору kmfoda/booksum, довжина яких

перевищувала 50000 слів. У двохсеріях порівнювалися безпосереднє використання генеративної моделі, та запропонований метод, у третьому експерименті порівнювались також інші варіанти багатоступеневої обробки довгих документів.

Результати аналізу показали, що ефективність порівнюваних варіантів залежить від довжини документа. На документах середньої довжини порівнювані підходи демонстрували близькі результати. На вибірці документів обсягом понад 15000 слів безпосереднє використання моделі дало вищі значення метрик ROUGE, тоді як запропонований метод забезпечив значно менший середній час обробки одного документа – 56.24 с/док проти 206.55 с/док, а також трохи вищий BERTScore. На наборі BookSum із 30 документів, довжина яких перевищувала 50000 слів, запропонований метод виявився кращим серед розглянутих варіантів за основними метриками якості, хоча не був найшвидшим за часом виконання.

Отримані результати свідчать, що зі збільшенням довжини документа переваги запропонованого багатоступеневого методу стають більш вираженими. Якщо на документах середньої довжини безпосереднє резюмування може зберігати незначну перевагу за частиною метрик, то на довгих текстах запропонований метод забезпечує кращу якість підсумкового резюме та менший час виконання.

Практичне значення одержаних результатів полягає у створенні працездатної програмної системи для резюмування документів обсягу, що перевищує контекстне вікно моделі, у якій користувач може пройти повний цикл роботи – від аутентифікації та завантаження файла до отримання готового резюме.

Наукова новизна роботи полягає у розробленні багатоступеневого методу автоматичного резюмування документів великого обсягу, який відрізняється від прямих однорівневих підходів поєднанням структурно-семантичного поділу тексту, кластеризації змістово близьких фрагментів, виділення ключових фактів, локального резюмування та подальшого глобального узагальнення.

Подальший розвиток дослідження може бути пов’язаний з удосконаленням механізму відбору ключових тверджень, розширенням набору режимів резюмування, використанням потужніших або спеціалізованих генеративних моделей, а також із залученням засобів контролю фактичної достовірності.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Jin H., Zhang Y., Meng D., Wang J., Tan J. A Comprehensive Survey on Process-Oriented Automatic Text Summarization with Exploration of LLM-Based Methods. arXiv. 2024. arXiv:2403.02901.

2. Zhang H., Yu P. S., Zhang J. A Systematic Survey of Text Summarization: From Statistical Methods to Large Language Models. ACM Computing Surveys. 2024.

3. Luo Z. et al. Factual Consistency Evaluation of Summarization in the Era of Large Language Models. Expert Systems with Applications. 2024. Vol. 254.

4. Gana B., Allende-Cid H., Rüping S., Becerra-Rozas M., Zamora J. A Systematic Review of Long Document Summarization Methods: Evaluation Metrics and Approaches. Neurocomputing. 2025. Vol. 655. Art. 131287.

5. Bashir A. S., Bichi A. A., Mahmud U., Bello A. M. Long-Text Abstractive Summarization using Transformer Models: A Systematic Review. Journal of the Brazilian Computer Society. 2025. Vol. 31, No. 1. P. 1263–1278. DOI: 10.5753/jbcs.2025.5786

6. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser Ł., Polosukhin I. Attention Is All You Need. Advances in Neural Information Processing Systems. 2017. Vol. 30.

7. Raschka S. Understanding Encoder and Decoder LLMs. Sebastian Raschka: веб-сайт. URL:https://magazine.sebastianraschka.com/p/understanding-encoder-and-decoder (дата звернення: 21.10.2025).

8. Zhang J., Zhao Y., Saleh M., Liu P. J. PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. Proceedings of the 37th International Conference on Machine Learning. 2020. P. 11328–11339.

9. OpenAI. GPT-4 Technical Report. arXiv. 2023. arXiv:2303.08774.

10. Файл: ChatGPT-Logo.svg. Wikipedia: веб-сайт. URL: https://uk.wikipedia.org/wiki/Файл:ChatGPT-Logo.svg (дата звернення: 14.09.2025).

11. Qwen Team. Qwen2.5 Technical Report. arXiv. 2024. arXiv:2412.15115.

12. Файл: Qwen logo.svg. Wikipedia: веб-сайт. URL: https://uk.wikipedia.org/wiki/Файл:Qwen_logo.svg (дата звернення: 21.11.2025).

13. Gemma social share image. Google Cloud: веб-сайт. URL: https://storage.googleapis.com/gweb-uniblog-publish-prod/images/Gemma-social-share.width-1300.jpg (дата звернення: 09.12.2025).

14. Guo M., Ainslie J., Uthus D., Ontañón S., Ni J., Sung Y.-H., Yang Y. LongT5: Efficient Text-To-Text Transformer for Long Sequences. Findings of the Association for Computational Linguistics: NAACL 2022. 2022. P. 724–736.

15. Chen X., Chen Z., Cheng S. CoTHSSum: Structured long-document summarization via chain-of-thought reasoning and hierarchical segmentation. Journal of King Saud University – Computer and Information Sciences. 2025. Vol. 37. Art. 40. DOI: 10.1007/s44443-025-00041-2.

16. Шушура О. М., Новицький К. В., Іванов В. О. Автоматичне резюмування наукових документів на основі моделей-трансформерів. Вісник Херсонського національного технічного університету. 2026. № 1(96). С. 411–418. DOI: https://doi.org/10.35546/kntu2078-4481.2026.1.55

17. Sonowal H., Sadhu S. Structure-Aware Chunking for Abstractive Summarization of Long Legal Documents. Proceedings of the 1st Workshop on NLP for Empowering Justice (JUST-NLP 2025). Mumbai, 2025. P. 171–178. DOI: 10.18653/v1/2025.justnlp-main.19.

18. Wang Q., Wang R., Zhao K., Amor R., Liu B., Liu J., Zheng X., Huang Z. SKGSum: Structured Knowledge-Guided Document Summarization. Findings of the Association for Computational Linguistics: ACL 2024. Bangkok, 2024. P. 1857–1871. DOI: 10.18653/v1/2024.findings-acl.110.

19. Davoodijam E., Alambardar Meybodi M. Evaluation metrics on text summarization: comprehensive survey. Knowledge and Information Systems. 2024. Vol. 66, No. 12. P. 7717–7738. DOI: 10.1007/s10115-024-02217-0.

20. Papineni K., Roukos S., Ward T., Zhu W.-J. BLEU: a Method for Automatic Evaluation of Machine Translation. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics. Philadelphia, 2002. P. 311–318.

21. Graham Y. Re-evaluating Automatic Summarization with BLEU and 192 Shades of ROUGE. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Lisbon, 2015. P. 128–137.

22. Reiter E. A Structured Review of the Validity of BLEU. Computational Linguistics. 2018. Vol. 44, no. 3. P. 393–401.

23. Zhang T., Kishore V., Wu F., Weinberger K. Q., Artzi Y. BERTScore: Evaluating Text Generation with BERT. International Conference on Learning Representations. 2020.

24. Steen J., Markert K. How to Find Strong Summary Coherence Measures? A Toolbox and a Comparative Study for Summary Coherence Measure Evaluation. Proceedings of the 29th International Conference on Computational Linguistics. Gyeongju, Republic of Korea, 2022. P. 6035–6049.

25. Amari A., Ben Ammar M. A. Markov-Enhanced Clustering for Long Document Summarization: Tackling the “Lost in the Middle” Challenge with Large Language Models. *arXiv*. 2025. arXiv:2506.18036.

26. Cohan A., Goharian N. Scientific Article Summarization Using Citation-Context and Article’s Discourse Structure. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Lisbon, 2015. P. 390–400.

27. Cohan A., Dernoncourt F., Kim D. S., Bui T., Kim S., Chang W., Goharian N. A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. New Orleans, 2018. P. 615–621.

28. Reimers N., Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing. Hong Kong, 2019. P. 3982–3992.

29. Gokhan T., Price M. J., Lee M. Graphs in clusters: a hybrid approach to unsupervised extractive long document summarization using language models. Artificial Intelligence Review. 2024. Vol. 57, no. 7. Art. 189. DOI: 10.1007/s10462-024-10828-w.

30. MacQueen J. B. Some methods for classification and analysis of multivariate observations. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Berkeley: University of California Press, 1967. Vol. 1. P. 281–297.

31. Dai W., He Q. Automatic summarization model based on clustering algorithm. Scientific Reports. 2024. Vol. 14. Art. 15302. DOI: 10.1038/s41598-024-66306-4.

32. Qaroush A. M., Naser L., Mali M., Naji A. Semantic-aware hybrid graph-based extractive summarization for Arabic texts. Journal of King Saud University – Computer and Information Sciences. 2025. Vol. 37. Art. 349. DOI: 10.1007/s44443-025-00359-x.

33. Sharma E., Huang L., Hu Z., Wang L. An Entity-Driven Framework for Abstractive Summarization. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing. Hong Kong, 2019. P. 3280–3291.

34. Guntakandla A. R. Modular architecture: A scalable and efficient system design approach for enterprise applications. World Journal of Advanced Research and Reviews. 2025. Vol. 26, no. 1. P. 3114–3126.

35. Common web application architectures. Microsoft Learn: веб-сайт. URL: https://learn.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures (дата звернення: 16.10.2025).

36. Overview of ASP.NET Core Authentication. Microsoft Learn: веб-сайт. URL: https://learn.microsoft.com/en-us/aspnet/core/security/authentication/ (дата звернення: 17.01.2026).

37. Angular documentation. Angular: веб-сайт. URL: https://angular.dev/ (дата звернення: 03.02.2026).

38. What is Amazon S3? Amazon Web Services Documentation: веб-сайт. URL: https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html (дата звернення: 28.12.2025).

39. SQL Server I/O Fundamentals. Microsoft Learn: веб-сайт. URL: https://learn.microsoft.com/en-us/sql/relational-databases/sql-server-storage-guide?view=sql-server-ver17 (дата звернення: 11.01.2026).

40. Wang T., Yang C., Zou M. A study of extractive summarization of long documents incorporating local topic and hierarchical information. Scientific Reports. 2024. Vol. 14. Art. 10140.

41. Du X., Saxena R., Perez-Beltrachini L., Minervini P., Titov I. Enhancing Long Document Long Form Summarisation with Self-Planning. Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Short Papers). Mumbai, 2025. P. 317–332.

42. Gu N., Ash E., Hahnloser R. H. R. MemSum: Extractive Summarization of Long Documents Using Multi-Step Episodic Markov Decision Processes. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics. Dublin, 2022. P. 6507–6522. DOI: 10.18653/v1/2022.acl-long.450.

43. Cao S., Wang L. AWESOME: GPU Memory-constrained Long Document Summarization using Memory Mechanism and Global Salient Content. Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Vol. 1: Long Papers. Mexico City, 2024. P. 5925–5941. DOI: 10.18653/v1/2024.naacl-long.330.

ДОДАТОК А

АВТОМАТИЧНЕ РЕЗЮМУВАННЯ НАУКОВИХ ДОКУМЕНТІВ НА ОСНОВІ МОДЕЛЕЙ-ТРАНСФОРМЕРІВ

Стаття у фаховому технічному журналі

Вісник Херсонського національного технічного університету №1 2026

Аркушів 17

Київ 2026

УДК 004.912

О. М. ШУШУРА
доктор технічних наук, професор
професор кафедри цифрових технологій в енергетиці
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
ORCID: 0000-0003-3200-720X

К. В. НОВИЦЬКИЙ
магістр кафедри цифрових технологій в енергетиці
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
ORCID: 0009-0002-5000-1397

В.О. ІВАНОВ
магістр кафедри цифрових технологій в енергетиці
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
ORCID: 0009-0002-2498-5628

АВТОМАТИЧНЕ РЕЗЮМУВАННЯ НАУКОВИХ ДОКУМЕНТІВ НА ОСНОВІ МОДЕЛЕЙ-ТРАНСФОРМЕРІВ

Зростання обсягів наукової літератури актуалізує розробку ефективних методів автоматичної генерації стислих резюме. Традиційні підходи стикаються з обмеженням контекстного вікна, що робить неможливим безпосередню обробку документів довжиною понад кілька тисяч токенів.

Метою даної роботи є розробка гібридного методу автоматичного резюмування для узагальнення документів, які перевищують стандартні обмеження розміру контекстного вікна моделей-трансформерів.

Розроблений гібридний метод поєднує екстрактивні та абстрактивні методи резюмування для ефективної обробки документів довільної довжини. Для екстрактивної фази була використана модель Sentence-BERT, з метою отримати семантичні векторні представлення речень, що дозволило ідентифікувати найбільш важливі частини тексту. На відміну від статистичних методів, Sentence-BERT захоплює глибинний семантичний зміст незалежно від лексичного складу. Наступна фаза методу видаляє семантичні дублікати за допомогою косинусної подібності, що забезпечує компактність проміжного представлення. Метод ідентифікує як точні дублікати, так і перефразування, створюючи компактне резюме. Фаза абстрактивної генерації виконується з використанням моделі BART-large-CNN, що поєднує двонаправлене кодування та авторегресивну генерацію. Це забезпечує створення зв'язних резюме з власними формулюваннями моделі, здатність до перефразування та об'єднання інформації з різних частин документу.

Розроблено програмне забезпечення для реалізації методу згідно з SOLID принципами, забезпечуючи модульність та можливість розширення системи.

Проведено порівняльне дослідження розробленого методу з чотирма категоріями базових підходів і спеціалізованою моделлю яка має розширене вікно контексту LongT5. Оцінка на вибірці з наукових статей з arXiv показала, що запропонований метод краще показує себе аніж традиційні методи та працює на рівні з LongT5, використовуючи при цьому стандартну модель BART-large-CNN. Метод був застосований без додаткового перед-навчання, що знижує обчислювальні вимоги.

Ключові слова: *резюмування тексту, обробка природньої мови, моделі трансформери, машинне навчання, інформаційні системи.*

O. M. SHUSHURA
Doctor of Technical Sciences, Professor
Professor, Department of Digital Technologies in Energy
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
ORCID: 0000-0003-3200-720X

K. V. NOVYTSKYI
Master, Department of Digital Technologies in Energy
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
ORCID: 0009-0007-2508-4350

V. O. IVANOV
Master, Department of Digital Technologies in Energy
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
ORCID: 0009-0002-2498-5628

AUTOMATIC SUMMARISATION OF SCIENTIFIC DOCUMENTS BASED ON TRANSFORMER-MODELS

The rapid growth of scientific literature highlights the need for effective methods of automatic summary generation. Traditional approaches face limitations imposed by fixed context windows, which makes the direct processing of documents longer than several thousand tokens impractical.

The aim of this work is to develop a hybrid automatic summarization method for processing documents that exceed the standard context window limitations of transformer-based models.

The proposed hybrid method combines extractive and abstractive summarization techniques to efficiently handle documents of arbitrary length. In the extractive phase, the Sentence-BERT model is employed to obtain semantic vector representations of

sentences, enabling the identification of the most informative parts of the text. Unlike statistical methods, Sentence-BERT captures deep semantic meaning regardless of lexical variation. The subsequent phase removes semantic duplicates using cosine similarity, ensuring the compactness of the intermediate representation. The method identifies both exact duplicates and paraphrases, producing a concise intermediate summary. The abstractive generation phase is performed using the BART-large-CNN model, which combines bidirectional encoding with autoregressive generation. This enables the production of coherent summaries with model-generated phrasing, paraphrasing capabilities, and the integration of information from different parts of the document.

Software implementing the proposed method was developed in accordance with the SOLID principles, ensuring modularity and extensibility of the system.

A comparative study was conducted against four categories of baseline approaches as well as the specialized LongT5 model with an extended context window. Evaluation on a dataset of scientific articles from arXiv demonstrated that the proposed method outperforms traditional approaches and achieves performance comparable to LongT5, while relying on the standard BART-large-CNN model. The method was applied without additional pre-training, which significantly reduces computational requirements.

Keywords: text summarization, natural language processing, transformer models, machine learning, information systems.

Постановка проблеми

Резюмування текстів є однією з ключових задач обробки природньої мови, яке має широке застосування в багатьох сферах життя, таких як юриспруденція, наукова діяльність, тощо. Резюмування буває двох видів – екстрактивне і абстрактивне. Суть екстрактивного резюмування полягає в виборі найголовніших речень в тексті, без перефразування, в той час як абстрактивне резюмування створює короткий виклад, перефразовуючи і узагальнюючи зміст своїми словами. Через значні виклики при перефразуванні, домінуючим методом резюмування до винайдення моделей-трансформерів був екстрактивний [1]. Сучасні моделі-трансформери, такі як BART, PEGASUS, T5 демонструють дуже гарні результати

для коротких і середніх за обсягом текстів, проте мають обмеження розміру контекстного вікна (зазвичай 512-1024 токенів), що значно ускладнює роботу з документами, довшими ніж контекстне вікно [2]. Усічення документів призводить до неповної обробки та, відповідно, неточного резюмування. Екстрактивне ж резюмування, як от з використанням графових методів, не завжди може забезпечити необхідну якість узагальнення. Це зумовило появу значної кількості досліджень, у яких пропонуються нові підходи для резюмування довгих текстів.

Аналіз останніх досліджень і публікацій

Абстрактивне узагальнення часто реалізують за допомогою таких передтренованих моделей як BART, PEGASUS, T5. Модель BART має гарні результати завдяки архітектурі енкодер-декодер і попередньому навчанні [3]. Модель PEGASUS оптимізована спеціально для задач резюмування завдяки використанню gap-sentence generation під час попереднього навчання [4]. Модель T5(Text-to-text Transformer) розглядає резюмування як задачу перетворення тексту, досягаючи універсальності за рахунок єдиної архітектури.

Для обробки довгих документів було запропоновано декілька підходів. Модель Hierarchical Transformers [5] використовують багаторівневу архітектуру для обробки локальних та глобальних залежностей. Модель Longformer [6] застосовує механізм розрідженої уваги для масштабування до довших послідовностей. Модель LED (Longformer Encoder-Decoder) поєднує переваги обох підходів для задач резюмування. Однак ці моделі вимагають значних обчислювальних ресурсів та великих обсягів даних для навчання.

Гібридні підходи намагаються поєднати переваги екстрактивних та абстрактивних методів. Автори Zhang et al. [7] запропонували двоетапний підхід, де спочатку екстрактивна модель виділяє важливі фрагменти, а потім абстрактивна модель генерує фінальне узагальнення.

Нарешті, Sentence-BERT [8] запропонував ефективний спосіб отримання семантичних векторних представлень для речень, що широко використовується у задачах пошуку схожості та кластеризації. Застосування цього підходу для

екстрактивного резюмування показало перспективні результати, але неясним залишається можливість його найкращого поєднання з абстрактивними моделями для обробки довгих документів.

Таким чином, незважаючи на значний прогрес у розробці спеціалізованих архітектур для обробки довгих документів, питання ефективного поєднання екстрактивних та абстрактивних методів на основі стандартних моделей-трансформерів залишається недостатньо дослідженим. Зокрема, потенціал інтеграції семантичних векторних представлень Sentence-BERT з абстрактивними моделями типу BART для створення масштабованих рішень без потреби у значних обчислювальних ресурсах та розширених контекстних вікнах потребує детального вивчення. Це визначає актуальність подальших досліджень гібридного застосування моделей-трансформерів для задачі резюмування документів.

Формулювання мети дослідження

Метою даної роботи є розробка гібридного методу резюмування для узагальнення документів, які перевищують стандартні обмеження розміру контекстного вікна моделей-трансформерів.

Для досягнення поставленої мети потрібно вирішити наступні задачі:

- розробити гібридний метод резюмування довгих документів, що поєднує екстрактивні та абстрактивні підходи в рамках чотирьох-етапної архітектури;
- здійснити програмну реалізацію розробленого методу з дотриманням принципів модульності та розширюваності;
- провести порівняльне дослідження запропонованого методу з існуючими підходами на репрезентативній вибірці наукових документів.

Викладення основного матеріалу дослідження

На основі комбінації Map-Reduce підходу з семантичними векторними представленнями та трансформерними моделями було розроблено гібридний метод резюмування документів, узагальнений алгоритм якого наведено на рисунку 1.

Метод поєднує Map-Reduce підхід з семантичним відбором і абстрактивною генерацією для досягнення достатньої якості узагальнень, без потреби використання моделей-трансформерів з розширеним контекстним вікном.

Як видно на рисунку 1, на початку методу відбувається розбиття вхідного документу на фрагменти оптимального розміру, при цьому кожен наступний фрагмент включає останні 100 слів попереднього, що запобігає втраті семантичного зв’язку між реченнями на межі розбиття [9].

На другому етапі методу використовується модель Sentence-BERT для семантичного вибору найбільш важливих речень з кожного фрагменту.

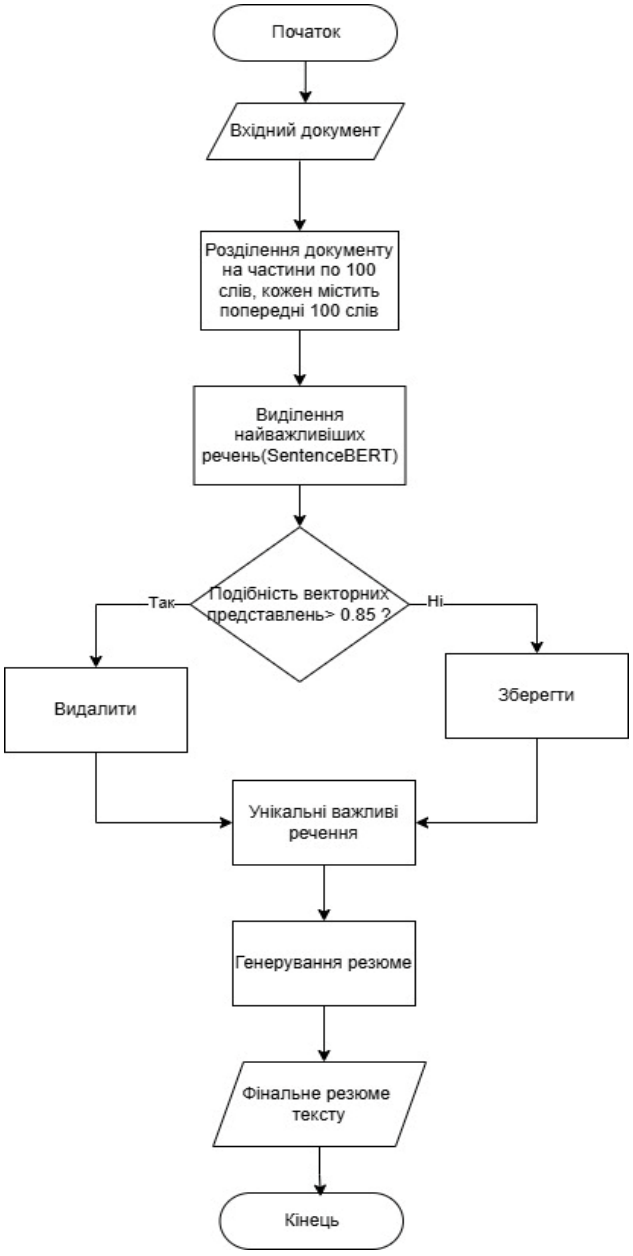

Рисунок 1 – Узагальнена схема методу гібридного резюмування документу

На відміну від статистичних методів (TF-IDF) або позиційних методів (Lead-k), Sentence-BERT є оновленою архітектурою BERT, яка спеціально адаптована для ефективної генерації векторних представлень речень. Оригінальна модель BERT вимагає передачі речень парами через мережу, що призводить до складності $O(n^2)$. Модель Sentence-BERT вирішує цю проблему через використання сіамської архітектури, що дозволяє кодувати кожне речення незалежно за один прохід, знижуючи складність до лінійної O(n). У даній роботі була використана модель all-MiniLM-L6-v2 з шістьма шарами, навчена методом контрастного навчання на понад мільярді пар речень. Алгоритм семантичної екстракції складається з декількох етапів. На першому етапі фрагмент документу розбивається на окремі речення за допомогою бібліотеки NLTK, що коректно обробляє абревіатури та інші специфічні конструкції наукових текстів. Далі кожне речення кодується моделлю all-MiniLM-L6-v2 у 384-вимірний вектор. Процес кодування включає токенізацію за допомогою WordPiece tokenizer(словник 30000 слів) та проходження через шість шарів трансформера з механізмом самоуваги, а також усереднення векторних подань для отримання підсумкового вектору речення. Для речення з m токенами, представленими векторами $h_1, h_2, \dots, h_m \in R^{768}$, отримуємо фінальне векторне представлення:

$$e = \frac{1}{m} * \sum_{i=1}^{m} h^i \quad (1)$$

де $e \in R^{384}$ після проекції з 768 на 384 виміри.

Потім обчислюється центроїд фрагменту – векторне представлення, яке задає загальну семантику тексту. Для фрагменту з n речень і векторами $e_1, e_2, \dots, e_n \in R^{384}$, центроїд $c \in R^{384}$ визначається як середнє арифметичне за формулою:

$$c = \frac{1}{n} * \sum_{i=1}^{n} e^i \quad (2)$$

Геометрично центроїд є центром усіх векторних представлень у просторі, базуючись на припущені, що найбільш репрезентативні речення про текст близькі до усередненого вектора усіх речень. Для кожного речення обчислюється косинусна подібність до центроїду за формулою:

$$similarity(e_i, c) = \frac{e_i * c}{(\,||e_i|| * ||c||)} \quad (3)$$

де $e_i * c$ – скалярний добуток, а $||e_i||$, $||c||$ – Евклідова норма.

Косинусна подібність нормалізована на інтервалі [-1, 1] і незалежна від довжини векторів, що означає що вона є оптимальною для високовимірних просторів. Відбираються декілька речень з найвищою подібністю, зберігаючи оригінальний порядок речень.

Третій етап методу видаляє семантичні дублікати на основі косинусної подібності векторних представлень. Оскільки перший етап створює фрагменти тексту, які містять частини інших для збереження цілісності тексту, а фаза виділення найважливіших частин обробляє кожен фрагмент незалежно, деякі речення можуть бути обрані декілька разів. Крім того, різні речення можуть висловлювати ідентичні ідеї в різних формулюваннях, що негативно впливає на якість фінального резюме. Для визначення семантичної схожості речень з векторами $e_1, e_2, \ldots, e_n \in R^{384}$ застосовується косинусна подібність векторних представлень:

$$similarity(e_i, e_j) = \frac{e_i * e_j}{(\,||e_i|| * ||e_j||)} \quad (4)$$

де $e_i * e_j$ – скалярний добуток, а $||e_i||$ - Евклідова норма.

Косинусна подібність нормалізована в інтервалі [-1, 1], де значення, наближені до 1, свідчать про високий рівень семантичної схожості між текстовими одиницями. Речення вважаються дублікатами за умови, що значення косинусної подібності перевищує поріг 0,85. Вибір даного порогового значення обумовлено результатами емпіричних спостережень: нижчі пороги призводять до надмірного видалення інформації, тоді як вищі не забезпечують фільтрацію дублікатів із незначними варіаціями формулювань.

Далі для набору з n речень $S = \{s_1, s_2, \ldots, s_n\}$ з векторами $E = \{e_1, e_2, \ldots, e_n\}$ ініціалізується порожня множина унікальних речень U. Для кожного речення s_i обчислюється попарна косинусна подібність з усіма реченнями U. Якщо максимальна подібність не перевищує задане значення, речення додається до

множини U, інакше відкидається. Таким чином, алгоритм має складність $O(n*m)$, де m – розмір фінальної множини U.

Наприкінці методу генерується фінальне резюме з використанням моделі BART-large-CNN(Bidirectional and Auto-Regressive Transformers), яка є архітектурою типу sequence-to-sequence на основі трансформерів, що поєднує двонаправлене кодування вхідного тексту і авторегресивну генерацію вихідного тексту. Довжина резюме регулюється параметрами, і зазвичай складає 80-120 слів. На основі розробленого методу було створене відповідне програмне забезпечення.

Програмний продукт для застосування методу розроблено в середовищі PyCharm на мові Python версії 3.11. Програма складається з файлу розробленої моделі та файлів інших методів, які використовувались для порівняння. На рисунку 2 зображена діаграма класів розробленого методу.

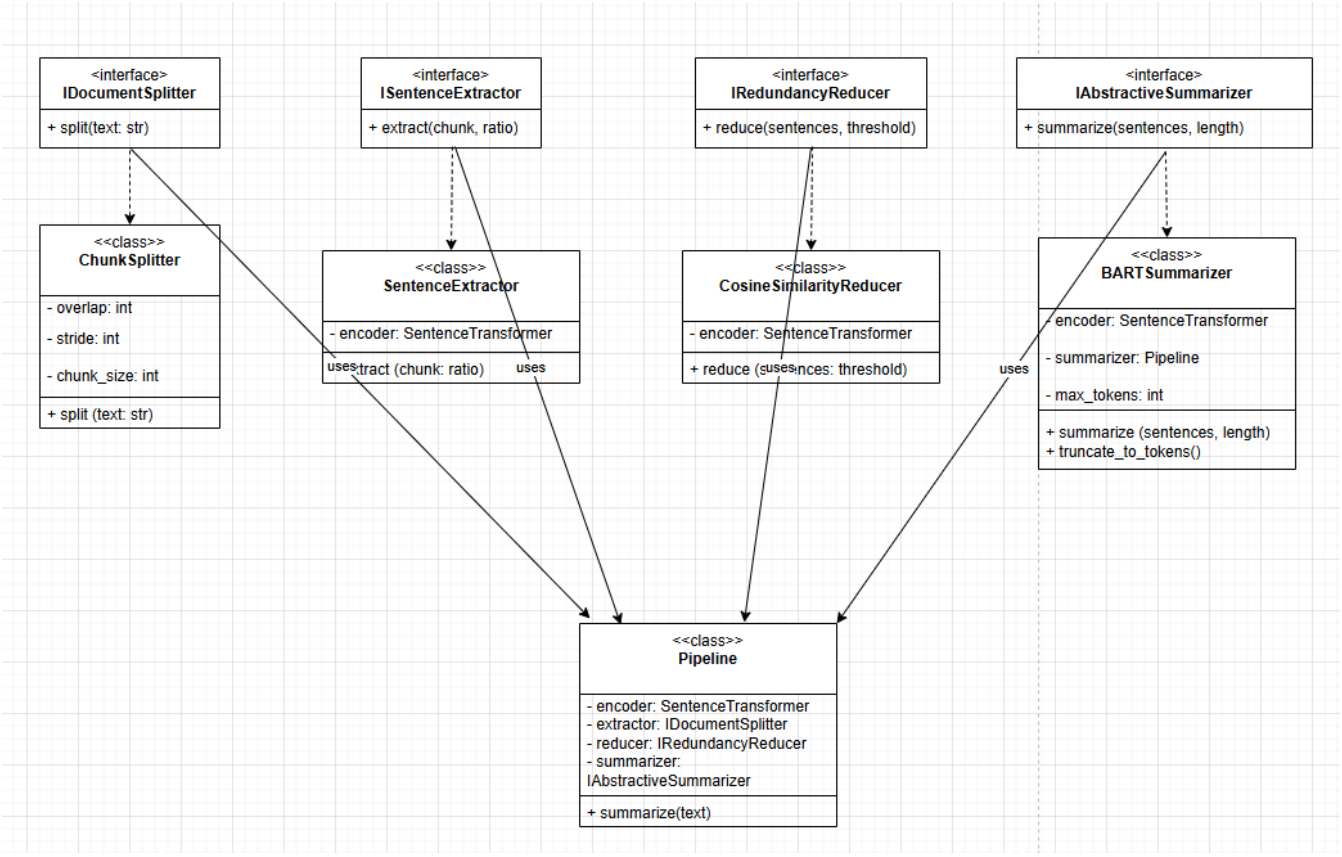

Рисунок 2 – Діаграма класів розробленого методу

Розроблена програма містить 4 інтерфейси, які визначають абстракції для кожної фази обробки документу. Інтерфейс IDocumentsSplitter відповідає за

розбиття вхідного тексту, ISentenceExtractor забезпечує виділення репрезентативних речень, IRedundancyReducer виконує видалення семантичних дублікатів, а IAbstractiveSummarizer генерує фінальне резюме. Кожен інтерфейс визначає один ключовий метод. Кожний інтерфейс має клас, який його реалізує. Клас ChunkSplitter розбиває документ з параметрами overlap і chunk_size. Наступний клас, SentenceExtractor використовує модель SentenceTransformer для створення векторних представлень і відбору речень на основі центроїду. Клас CosineSimilarityReducer застосовує косинусну подібність для видалення дублікатів, також використовуючи SentenceTransformer. Головний клас BARTSummarizer використовує модель BART для генерації фінального резюме через Pipeline клас.

Клас Pipeline виступає координатором всієї системи, яка агрегує всі компоненти через їх інтерфейси. Використовуючи розроблене програмне забезпечення, було виконано порівняльне дослідження методу на тестових даних.

Тестування проводилось на наборі наукових статей з arXiv. Були відібрані 5 документів різної довжини (від 3000 до 6000 слів) з галузей фізики і математики. Кожен документ містив короткий опис на початку документу який був використаний в якості порівняння для оцінки якості згенерованого резюме.

Оцінка якості проводилась з використанням двох груп метрик. Перша група – метрики ROUGE (ROUGE-1, ROUGE-2, ROUGE-L), які вимірюють лексичне перекриття і найдовшої спільної послідовності. Друга група метрик – BERTscore, які вимірюють семантичну подібність між згенерованим і еталонним резюме на основі концептуальних векторних представлень.

В таблиці 1 наведені результати порівнянь за метриками ROUGE. Були розглянуті як традиційні методи (TF-IDF), так і більш новітні, на основі машинного навчання. Метод на основі позиції речення демонструє високі результати завдяки специфічній структурі наукових статей, де найважливіша інформація традиційно концентрується на початку документа. Однак цей підхід має обмежену універсальність і не забезпечує ефективної роботи з документами альтернативної структури.

Таблиця 1 – Порівняння ROUGE-1 метрики для різних методів

Метод	Складність	ROUGE-1
Lead-k(позиція речення)	O(1)	0.33
TF-IDF + косинусна подібність	$O(n^2)$	0.27
Sentence-BERT + центроїд(розроблений)	O(n)	0.35
BERT cls token	$O(n^2)$	0.38

Метод TF-IDF показав найнижчі результати серед досліджуваних підходів. Це пояснюється тим, що він враховує лише частотність слів, ігноруючи їх семантичну значущість, що призводить до неефективної обробки текстів з великою кількістю спеціалізованої термінології.

Метод на основі BERTcls токенів продемонстрував найкращі показники якості, проте поступається використаному в дослідженні методу Sentence-BERT з вибором за центроїдом за швидкодією обробки. Ця різниця стає критичною при роботі з довгими документами, де час обробки є важливим фактором.

Застосування семантичних векторних представлень Sentence-BERT забезпечує оптимальний баланс між якістю відбору найважливіших речень та ефективністю обробки документів. На відміну від методів, що базуються на позиції речення чи частотності слів, цей підхід здатний вилучати інформативні речення з будь-якої частини документа, включаючи середину та кінець, а не лише з початку, як це характерно для традиційних методів. Така властивість робить його більш універсальним для різних типів документів та структур викладу інформації.

На рисунку 3 зображені результати порівняння за ROUGE-метриками. Були розглянуті наступні методи:

- розроблений метод: Map-Reduce з вектором семантичних представлень;
- LongT5: спеціалізована модель-трансформер, здатна оброблювати довгі документи(до 16000 токенів);
- Map-Reduce+BART: базовий Map-Reduce без семантичної екстракції;
- TextRank+BART: виділення найважливіших речень на основі графового алгоритму і генерація резюме через BART;
- T5-base: стандартна модель трансформер, яка приймає до 1024 токени.

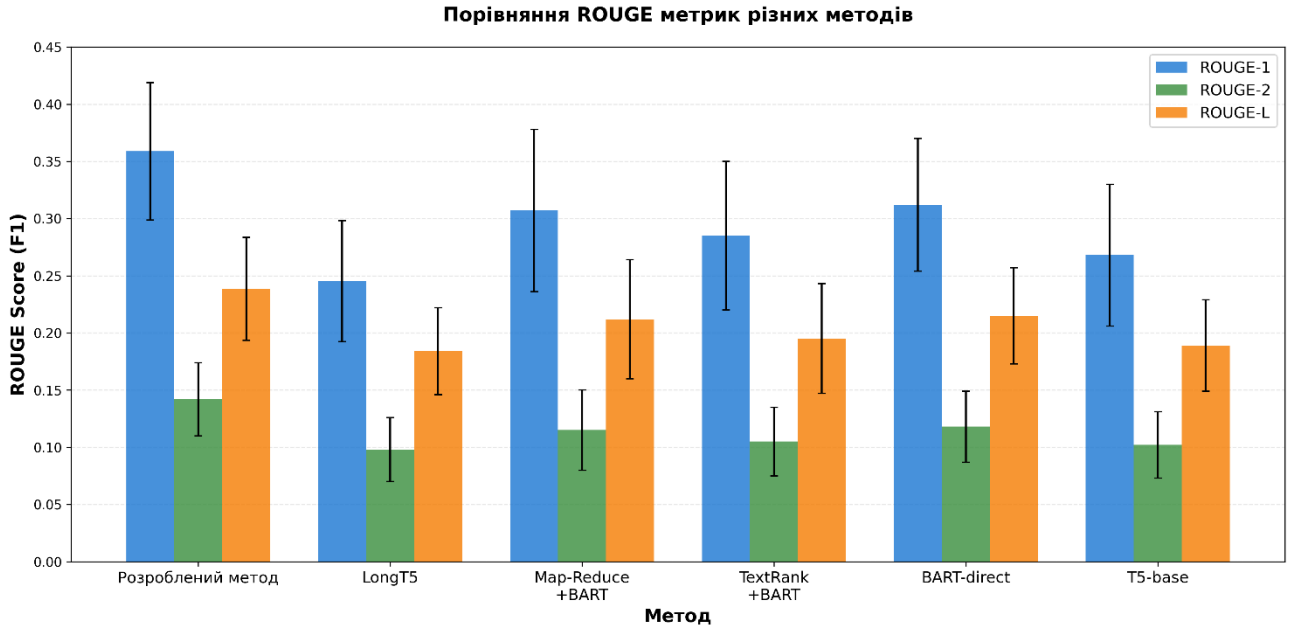

Рисунок 3 – Порівняння результатів ROUGE-метрик для різних методів

Як видно з діаграми на рисунку 3, за метрикою ROUGE-1 F1 наявна перевага запропонованого методу – він показав найвищий результат (0.5852±0.027). Це на 4.4% краще ніж спеціалізована модель-трансформер LongT5(0.5603±0.017) і на 16.1% краще за базовий Map-Reduce+BART.

За метрикою ROUGE-2, що вимірює збіг біграм, запропонований метод показав кращі результати також: 0.1420±0.032 проти 0.0980±0.028 у LongT5 (+44.9%). Це вказує на те, що згенеровані резюме не тільки містять правильні слова, але й зберігають типові словосполучення з оригінальних документів. За метрикою ROUGE-L F1 (найдовша спільна підпослідовність) результат запропонованого методу становить 0.2385±0.045 проти 0.1840±0.038 у LongT5, що свідчить про краще збереження структури та порядку інформації з оригінального тексту.

Результати порівняння за метрикою BERTscore представлені на рисунку 4.

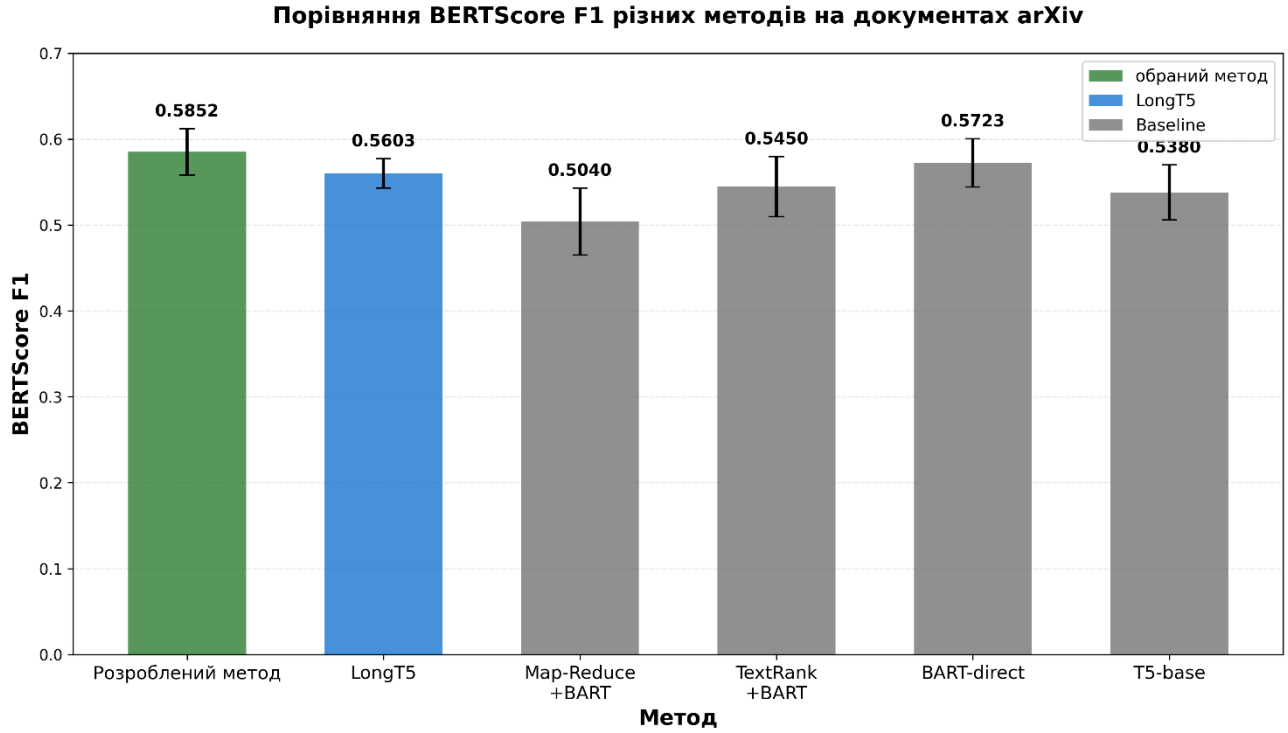

Рисунок 4 – Порівняння результатів BERTscore-метрик для різних методів

Як видно з діаграми, запропонований метод показав найкращий результат з усіх методів – 0.5852. Це на 4.4% краще за спеціалізовану модель-трансформер LongT5 і на 16% краще за схожий Map-Reduce метод.

Висновки

Розроблено гібридний метод резюмування довгих наукових документів, що поєднує розбиття документу на перекриваючі фрагменти, семантичну екстракцію з використанням Sentence-BERT, видалення семантичних дублікатів через косинусну подібність векторних представлень та абстрактивну генерацію за допомогою BART-large-CNN. Метод дозволяє обробляти документи довільної довжини без використання спеціалізованих моделей з розширеним контекстним вікном.

Здійснено програмну реалізацію методу з дотриманням SOLID принципів, що забезпечує модульність та можливість заміни окремих компонентів системи. З використанням розробленого програмного забезпечення проведено порівняльне дослідження методу на вибірці наукових статей arXiv. За метрикою ROUGE-1 F1

запропонований метод показав результат 0.5852±0.027, що на 4.4% краще за спеціалізовану модель LongT5 та на 16.1% краще за базовий Map-Reduce+BART. Результати підтверджують ефективність гібридного підходу для резюмування документів з використанням стандартних моделей-трансформерів без додаткового дотренування.

Отримані результати підтверджують доцільність подальших досліджень з поєднання семантичної екстракції на основі векторних представлень з абстрактивною генерацією для задачі резюмування наукових документів.

Список використаної літератури

1. See A., Liu P. J., Manning C. D. Get To The Point: Summarization with Pointer-Generator Networks. Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics. 2017. Vol. 1. P. 1073–1083. https://doi.org/10.18653/v1/P17-1099

2. Huang L., Wu L., Wang L. An Empirical Survey on Long Document Summarization: Datasets, Models, and Metrics. ACM Computing Surveys. 2022. Vol. 55, № 8. Article 157. https://doi.org/10.1145/3545176

3. Lewis M., Liu Y., Goyal N., Ghazvininejad M., Mohamed A., Levy O., Stoyanov V., Zettlemoyer L. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. 2020. P. 7871–7880. https://doi.org/10.18653/v1/2020.acl-main.703

4. Zhang J., Zhao Y., Saleh M., Liu P. J. PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. Proceedings of the 37th International Conference on Machine Learning. 2020. P. 11328–11339. https://doi.org/10.48550/arXiv.1912.08777

5. Liu Y., Lapata M. Hierarchical Transformers for Multi-Document Summarization. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. P. 5070–5081. https://doi.org/10.18653/v1/P19-1500

6. Beltagy I., Peters M. E., Cohan A. Longformer: The Long-Document Transformer. arXiv preprint arXiv:2004.05150. 2020. https://doi.org/10.48550/arXiv.2004.05150

7. Zhang X., Wei F., Zhou M. HIBERT: Document Level Pre-training of Hierarchical Bidirectional Transformers for Document Summarization. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019. P. 5059–5069. https://doi.org/10.18653/v1/P19-1499

8. Reimers N., Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing. 2019. P. 3982–3992. https://doi.org/10.48550/arXiv.1908.10084

9. Automatic text summarization of scientific articles using transformers: A brief review. Journal of Artificial Intelligence. 2024. Vol. 7, № 5. https://doi.org/10.32629/jai.v7i5.1331

References

1. See, A., Liu, P. J., & Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Vol. 1, pp. 1073–1083). Association for Computational Linguistics. https://doi.org/10.18653/v1/P17-1099

2. Huang, L., Wu, L., & Wang, L. (2022). An empirical survey on long document summarization: Datasets, models, and metrics. ACM Computing Surveys, 55(8), Article 157. https://doi.org/10.1145/3545176

3. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (pp. 7871–7880). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.703

4. Zhang, J., Zhao, Y., Saleh, M., & Liu, P. J. (2020). PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. In Proceedings of the 37th

International Conference on Machine Learning (pp. 11328–11339). PMLR. https://doi.org/10.48550/arXiv.1912.08777

5. Liu, Y., & Lapata, M. (2019). Hierarchical transformers for multi-document summarization. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (pp. 5070–5081). Association for Computational Linguistics. https://doi.org/10.18653/v1/P19-1500

6. Beltagy, I., Peters, M. E., & Cohan, A. (2020). Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150. https://doi.org/10.48550/arXiv.2004.05150

7. Zhang, X., Wei, F., & Zhou, M. (2019). HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (pp. 5059–5069). Association for Computational Linguistics. https://doi.org/10.18653/v1/P19-1499

8. Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-Networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (pp. 3982–3992). Association for Computational Linguistics. https://doi.org/10.48550/arXiv.1908.10084

9. Automatic text summarization of scientific articles using transformers: A brief review. (2024). Journal of Artificial Intelligence, 7(5). https://doi.org/10.32629/jai.v7i5.1331

ДОДАТОК Б

HYBRID APPROACH FOR AUTOMATIC SUMMATIZATION OF LONG SCIENTIFIC DOCUMENTS

Апробація

XXIII Міжнародна науково-практична конференція молодих вчених та студентів «Сучасні проблеми наукового забезпечення енергетики: безпека, стійкість, ІТ та екологічний моніторинг (до 40-річчя Чорнобильської катастрофи)»

Аркушів 4

Київ 2026

Application form
for participation in XXIII International scientific and practical conference
of young scientists and students
"Modern problems of scientific support of power engineering"
(April 17 – 22, 2026)

Report theme	
Hybrid approach for automatic summatization of long scientific documents	
Report UDC	004.912
Conference section	Information technologies and computer modeling

AUTHOR 1

Name Surname Patronymic	Novytskyi Kostya Vitaliovich
Place of study (organisation name)	
Igor Sikorsky Kyiv Polytechnic Institute	
Status	MS's programme of the 2 year
Email	kostyanovytskyi@gmail.com

AUTHOR 2

Name Surname Patronymic	
Place of study (organisation name)	
Status	of the year
Email	

ACADEMIC LEADER

Name Surname Patronymic	Shushura Oleksii Mykolaiovich
Position	prof.
Academic degree	doc.eng.sc.
Workplace (organisation name)	
Igor Sikorsky Kyiv Polytechnic Institute	
Email	
Google scholar	
https://scholar.google.com/citations?user=beUGko0AAAAJ&hl=en	

technical info

Hybrid approach for automatic summatization of long scientific documents.

NOVYTSKYI K.V., MS's programme
Academic leader - prof., doc.eng.sc. Shushura O.M.

Novytskyi K.V.

Shushura O.M.

UDC 004.912

[1] MS's programme 2nd year Novytskyi K.V.
[1] Prof., doc.eng.sc. Shushura O.M.
https://scholar.google.com/citations?user=beUGko0AAAAJ&hl=en

[1] Igor Sikorsky Kyiv Polytechnic Institute

HYBRID APPROACH FOR AUTOMATIC SUMMATIZATION OF LONG SCIENTIFIC DOCUMENTS

Statement of the problem and its relevance. The rapid growth of scientific literature necessitates efficient methods for automatic summary generation. Traditional transformer-based models demonstrate excellent results for short and medium-length texts, but face significant limitations due to fixed context windows (typically 512-1024 tokens), which complicates processing of documents longer than the context window [1]. Document truncation leads to incomplete processing and consequently inaccurate summarization. Moreover, purely extractive approaches, such as graph-based methods, cannot always provide the necessary quality of summarization [2].

Analysis of the latest research. Modern transformer models such as BART, PEGASUS, and T5 achieve good results through encoder-decoder architecture and pre-training [3,4]. For processing long documents, several approaches have been proposed. Hierarchical Transformers employ multi-level architecture for processing local and global dependencies. Longformer applies sparse attention mechanisms to scale to longer sequences [5]. However, these models require significant computational resources and large training datasets. Hybrid approaches attempt to combine the advantages of extractive and abstractive methods, but the potential for integrating semantic vector representations with abstractive models to create scalable solutions without requiring extended context windows remains insufficiently studied [6].

Goal formulation. The aim of this work is to develop a hybrid summarization method for processing documents that exceed the standard context window limitations of transformer models, combining Map-Reduce approach with semantic selection and abstractive generation.

Main part. A hybrid method was developed based on a four-stage architecture, combining extractive and abstractive summarization approaches. The first stage performs document splitting into overlapping fragments, with each subsequent fragment including the last 100 words of the previous one to prevent loss of semantic connections at fragment boundaries.

The extractive phase employs the Sentence-BERT model to obtain semantic vector representations of sentences. Unlike statistical methods (TF-IDF) or positional methods (Lead-k), Sentence-BERT captures deep semantic meaning independently of lexical variation. The all-MiniLM-L6-v2 model with six layers was utilized, trained using contrastive learning on over one billion sentence pairs. For each fragment, a centroid is computed as the arithmetic mean of all sentence vectors. The centroid for a fragment with n sentences and vectors $e_1, e_2, ..., e_n \in R^{384}$ is defined by the formula:

$$c = (1/n)\Sigma_{i=1}^{n} e_i \quad (1)$$

Sentences with highest cosine similarity to the centroid are selected, preserving the original order. To evaluate the effectiveness of different extractive approaches, a comparison was conducted on the extractive phase. Table 1 presents the comparative analysis of ROUGE-1 F1 scores for various sentence selection methods.

Table 1 – Comparison of ROUGE-1 F1 metric for different extractive methods

Method	ROUGE-1 F1
Lead-k (positional)	0.3319
TF-IDF (statistical)	0.2723
BERT CLS token	0.3760
Sentence-BERT + centroid	0.3460

The results demonstrate that while BERT CLS token approach achieves the highest score (0.3760), the Sentence-BERT with centroid-based selection (0.3460) provides an optimal balance between quality and computational efficiency, significantly outperforming traditional TF-IDF (0.2723) and positional Lead-k (0.3319) methods. The semantic approach captures important sentences regardless of their position in the document, unlike positional methods that rely primarily on document structure.

The third stage removes semantic duplicates based on cosine similarity of vector representations with a threshold of 0.85. Sentences are considered duplicates if their similarity exceeds this threshold. The algorithm has complexity O(n×m), where m is the size of the final unique sentence set.

The final abstractive generation phase utilizes the BART-large-CNN model (Bidirectional and Auto-Regressive Transformers), which combines bidirectional encoding of input text with autoregressive generation of output text. This enables creation of coherent summaries with model-generated phrasing, paraphrasing capabilities.

Software implementing the method was developed in accordance with SOLID principles, ensuring modularity and component replaceability. The architecture includes four interfaces and corresponding implementation classes coordinated through a central Pipeline class.

A comparative study of the complete method was conducted on a dataset of scientific articles from arXiv. Five documents of varying length (3000 to 6000 words) from physics and mathematics were selected. Quality assessment employed ROUGE metrics (ROUGE-1, ROUGE-2, ROUGE-L) and BERTScore. The proposed method demonstrated the highest ROUGE-1 F1 score (0.5852±0.027), which is 4.4% better than the specialized LongT5 model with extended context window (16384 tokens) and 16.1% better than baseline Map-Reduce+BART. For ROUGE-2 metric, the advantage was 44.9% compared to LongT5. BERTScore results confirmed method effectiveness, showing the best result among all studied approaches.

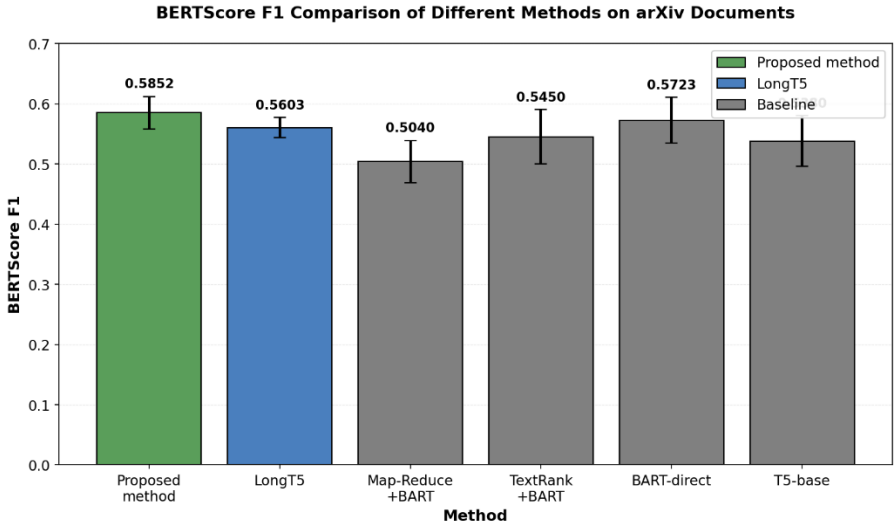

Figure 1 – BERTScore F1 comparison of different methods on arXiv documents

As demonstrated in Figure 1, the proposed hybrid method achieved the highest BERTScore F1 of 0.5852, outperforming the specialized LongT5 model (0.5603) by 4.4% and the baseline Map-Reduce+BART approach (0.5040) by 16.1%. The method also surpassed BART-direct (0.5723),

TextRank+BART (0.5450), and T5-base (0.5380) approaches. For ROUGE-2 metric, the proposed method demonstrated a 44.9% advantage over LongT5. These results confirm that the integration of semantic extraction using Sentence-BERT with BART-based abstractive generation creates an effective solution for long document summarization without requiring specialized models with extended context windows or additional pre-training. The practical advantages of the proposed approach extend beyond quality metrics to deployment considerations. Unlike specialized long-context models that require substantial GPU memory (LongT5 demands approximately 16GB for inference), the proposed method operates efficiently with standard hardware configurations, requiring only 6-8GB of GPU memory due to its chunk-based processing strategy. This reduced resource footprint enables deployment on widely available consumer-grade GPUs and facilitates integration into existing document processing pipelines without infrastructure upgrades. Furthermore, the modular architecture allows for incremental processing of extremely long documents (exceeding 20,000 tokens) that would be entirely impractical for monolithic transformer models, even those with extended context windows. These characteristics make the method particularly suitable for real-world applications in academic institutions and research organizations with limited computational resources.

Conclusions. A hybrid method for summarizing long scientific documents was developed, combining document splitting into overlapping fragments, semantic extraction using Sentence-BERT, semantic duplicate removal through cosine similarity of vector representations, and abstractive generation using BART-large-CNN. The method enables processing of documents of arbitrary length without specialized models with extended context windows. Software implementation adhering to SOLID principles ensures modularity and component replaceability. Comparative study results demonstrate that the proposed method achieves 4.4% improvement over the specialized LongT5 model while using the standard BART-large-CNN model without additional pre-training, significantly reducing computational requirements.An important consideration in evaluating summarization systems is their behavior across different document structures and scientific domains. The proposed method demonstrates robust performance on documents with varying organizational patterns, including traditional IMRaD structure (Introduction, Methods, Results, Discussion) common in experimental sciences, as well as more theoretical papers in mathematics and computer science that may follow alternative structures. The semantic extraction approach adapts naturally to these variations because it identifies important content based on meaning rather than position or formatting conventions.

References:

1. See, A., Liu, P. J., & Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. *In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics* (Vol. 1, pp. 1073–1083). https://doi.org/10.18653/v1/P17-1099

2. Huang, L., Wu, L., & Wang, L. (2022). An empirical survey on long document summarization: Datasets, models, and metrics. *ACM Computing Surveys*, 55(8), Article 157. https://doi.org/10.1145/3545176

3. Lewis, M., Liu, Y., Goyal, N., et al. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *In Proceedings of ACL 2020* (pp. 7871–7880). https://doi.org/10.18653/v1/2020.acl-main.703

4. Zhang, J., Zhao, Y., Saleh, M., & Liu, P. J. (2020). PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. *In Proceedings of ICML 2020* (pp. 11328–11339). https://doi.org/10.48550/arXiv.1912.08777

5. Beltagy, I., Peters, M. E., & Cohan, A. (2020). Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*. https://doi.org/10.48550/arXiv.2004.05150

6. Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-Networks. *In Proceedings of EMNLP 2019* (pp. 3982–3992). https://doi.org/10.48550/arXiv.1908.10084